UNIVERSIDAD DE ORIENTE
NÚCLEO DE SUCRE
ESCUELA DE CIENCIAS
DEPARTAMENTO DE INFORMÁTICA

SISTEMA DE INFORMACIÓN BAJO AMBIENTE WEB PARA EL INCENTIVO, PROMOCIÓN Y APOYO AL SECTOR TURISMO

ARNELLA JOSÉ GÓMEZ ARIAS
JHOENDRIS DEL CARMEN GONZÁLEZ RICARDI
LUIS JOSÉ TENA RUMBOS
MANUEL JOSÉ CONTASTI RIVERO
MARCOS JOSÉ MARÍN MATA
MIGUEL SANTIAGO CARMONA SALAZAR

TRABAJO DE GRADO PRESENTADO COMO REQUISITO PARCIAL PARA OPTAR AL TÍTULO DE LICENCIADO EN INFORMÁTICA

CUMANÁ, ENERO 2018

SISTEMA DE INFORMACIÓN BAJO AMBIENTE WEB PARA EL INCENTIVO, PROMOCIÓN Y APOYO AL SECTOR TURISMO

APROBADO POR:

_____
Prof. Eugenio Betancourt
Asesor Académico

_____
Profa. Dianelina Aguiar
Co-asesor Académico

_____
Profa. Lisbeth Fernández
Co-asesor Académico

_____
Prof. Leopoldo Acuña S.
Co-asesor Académico

_____
Jurado

_____
Jurado

ÍNDICE

DEDICATORIA ..... **VI**
AGRADECIMIENTO ..... **IX**
LISTA DE TABLAS ..... **X**
LISTA DE FIGURAS ..... **XI**
RESUMEN ..... **XVI**
INTRODUCCIÓN **1**
CAPÍTULO I. PRESENTACIÓN **6**
1.1. PLANTEAMIENTO DEL PROBLEMA 6
1.2. ALCANCE Y LIMITACIONES. 12
1.2.1. Alcance 12
1.2.2. Limitaciones 15
1.3. OBJETIVOS 16
1.3.1. Objetivo General 16
1.3.2. Objetivos Específicos 16
CAPÍTULO II. MARCO DE REFERENCIA **18**
2.1. MARCO TEÓRICO 18
2.1.1. Antecedentes de la investigación 18
2.1.1. Bases teóricas 21
2.1. METODOLOGÍA DEL ÁREA APLICADA 39
2.1.1. Metodología de la investigación 39
2.1.2. Metodología del área aplicada 41
CAPÍTULO III. DESARROLLO **59**
3.1. MODELO PRINCIPAL DE PROCESOS. 59
3.1.1. Inicio del Proyecto. 60
3.1.2. Planificación Inicial del Proyecto. 63
3.1.1.1. Estructura de Desglose de Productos. 65

3.1.1.2. Plan de Gestión de Alcance. 66
3.1.1.3. Límites del Proyecto. 68
3.1.1.4. Herramientas de Software para Gestionar el Proyecto. 69
3.1.1.5. Estructura de Costos Asociados. 69
3.1.1.6. Planificación de la Gestión de Tiempos. 70
3.1.1.7. Planes Suplementarios. 73
3.1.3. Análisis del Contexto de la Aplicación. 74
3.1.3.1. Modelado de Negocios. 76
3.1.3.2. Modelado de Objetivos. 76
3.1.3.3. Modelado de Procesos de Negocios. 80
3.1.3.4. Modelado de Objetos de Negocios. 84
3.1.3.5. Modelado de Reglas de Negocios. 84
3.1.3.6. Modelado de Actores. 86
3.1.4. Desarrollo Inicial de Requisitos. 92
3.1.4.1. Descubrimiento de Requisitos. 94
3.1.4.2. Análisis de Requisitos. 96
3.1.5. Diseño Inicial de la Aplicación. 97
3.1.5.1. Vista Funcional. 99
3.1.5.2. Vista Estructural. 103
3.1.5.3. Vista de Comportamiento. 113
3.1.5.4. Vista de Implementación. 115
3.1.5.5. Vista de Despliegue. 116
3.1.6. Planificación de la Iteración i. 117
3.1.7. Refinamiento de Requisitos del Incremento. 120
3.1.8. Diseño Detallado del Incremento. 122
3.1.9. Codificación del Incremento. 123
3.1.10. Pruebas de Incremento. 126
3.1.11. Verificación y Validación de Incrementos. 127

3.1.12. Entrega Final de la Aplicación. 129
3.1.13. Cierre del Proyecto. 131
3.1.14. Dirección, Seguimiento y Control del Proyecto. 133
3.1.14.1. Dirección del Proyecto. 133
3.1.14.2. Seguimiento y Control del Proyecto. 134
3.1.15. Gestión de Requisitos. 135
3.1.16. Verificación y Validación de Productos (Estática). 137
3.1.17. Gestión de Configuración de Software. 139
3.1.18. Gestión de Riesgos. 141
3.1.19. Procesos de Gestión. 145
3.1.20. Ejemplos de Ciclo de Versión. 148
3.1.20.1. Refinamiento de la arquitectura del software. 148
3.1.20.2. Diseño detallado de la primera versión. 148
3.1.20.3. Diseño de interfaz. 148
3.1.20.4. Diseño de base de datos. 155
3.1.20.5. Diseño detallado de componentes. 155
3.1.20.6. Programación e integración de las iteraciones de la aplicación. 155
3.1.20.7. Gestión de Pruebas. 155
3.2. Ciclo de versiones 161
CONCLUSIONES 206
RECOMENDACIONES 211
BIBLIOGRAFÍA 212
APÉNDICES 218
HOJAS DE METADATOS 739

DEDICATORIA

A Dios por haberme permitido llegar hasta este punto y haberme dado salud para lograr mis objetivos, por fortalecer mi corazón e iluminar mi mente, A mis padres por ser el pilar fundamental en todo lo que soy, en toda mi educación, tanto académica como en la vida. A mi esposa por su comprensión, motivación, y su amor incondicional. Y por supuesto a mis hijos todo esto es por y para ellos. A mis maestros aquellos que marcaron cada etapa de nuestro camino universitario y motivaron para la culminación de dicho camino. A mis compañeros y amigos de estudio, por la lucha, dedicación, y apoyo mutuo en nuestra formación profesional. En fin, a todos aquellos que participaron de alguna u otra manera en que todo este trabajo se hiciera posible. Gracias, mil gracias a todos.

Miguel Santiago Carmona Salazar

A Dios porque gracias a ti estoy cumpliendo este sueño, por protegerme durante todo el camino y darme la fuerza para superar todos los obstáculos, guiarme y llenándome siempre de serenidad. A mi madre Marianela Gómez por ser mi mayor fuente de inspiración, admiración, fortaleza, por apoyarme y por todos los sacrificios que hizo por mí. A mis hermanos Blanca Cesín y Javier Gómez que fueron pilar fundamental y siempre recordarme que si se podía lograr. A mi comadre y hermana que la vida me regalo Luzmary de Nóbrega, gracias por todos los pleitos que me formaste para que no abandonará mi título, te debo mucho. A mis hijos, por ustedes quise ser mejor persona, para ustedes este logro y a ti Leopoldo Acuña por ayudarme a aprender de mis errores, ser mi apoyo, mi mano derecha, darme tu compañía, compresión y tu amor, te amo.

Arnella José Gómez Arias

Mi carrera fue una difícil travesía, en donde me encontré conmigo misma y con

lo que soy capaz de hacer. El cariño y la motivación por parte de mi abuela fueron detonantes del esfuerzo realizado en cada tramo, el cual se ve plasmado en este trabajo y que está dedicado a ella, mi estrella guía, Gisela González por su apoyo incondicional; siempre presente en cada paso. A mis compañeros, a esos grandes amigos que la universidad me dio y muy especialmente a mi hermana “Jhorga” que a pesar de lo difícil de nuestra realidad siempre estuvieron a mi lado.

Jhoendris del Carmen González Ricardi

El trabajo que realice en este proyecto se lo dedico a Milagros Mata, mi madre, quien a pesar de lo difícil que se nos puso la vida nunca se rindió para darme lo mejor. A mis compañeros universitarios, a los cuales les agradezco los momentos y aprendizajes que logramos compartir. A la madre que la universidad me dio, Carmelys Rodríguez, siempre empeñada en dar lo mejor de sí para hacer mejores a sus estudiantes.

Marcos José Marín Mata

Gracias primeramente a mi madre Aracelis Rumbos, que siempre ha estado para mi sin importar que. A mis tías y tíos que con su cariño me han llenado la vida de buenos momentos en familia. A mis hermanos que a pesar de las distancias siempre he sentido su amor y su apoyo. A las personas que han influido en mí y me han tratado como a uno más de su familia (Dianelina Aguiar, Arnella Gómez y Leopoldo Acuña), a Eugenio Betancourt y Leopoldo por exigirme siempre más para dar lo mejor de mí, lo que me ayudó a superarme, creer en mí mismo y crecer profesionalmente. A todos, de corazón muchas gracias.

Luis José Tena Rumbos

A Dios por sobre todas las cosas por haberme permitido llegar hasta aquí, por darme siempre las fuerzas necesarias para salir adelante por sobre cualquier

adversidad. A mis dos madres Carmen Priscila Velásquez de Rivero y Maritza Rivero por sus consejos sus enseñanzas, por llenarme de amor, por estar allí siempre para mí por formarme con valores y principios que me han permitido ser una persona de bien, las amo. A mi novia Johana Lashley por ser siempre mi apoyo, por esa persona que siempre me motiva a seguir adelante, por regalarme una sonrisa en los momentos más difíciles de mi vida, por bridarme su ayuda siempre que la necesite y todos mis compañeros de la universidad por todos estos años en los cuales hemos compartido momentos que siempre llevare en mi mente y mi corazón.

Manuel José Contasti Rivero

AGRADECIMIENTO

A Dios todo poderoso por ser nuestro guía y sustento.

A la Universidad de Oriente, nuestro segundo hogar.

A la empresa Serman Import, CA por brindarnos sus instalaciones y equipos.

Al Ing. Leopoldo Acuña, por su admirable disposición y colaboración en la realización de este proyecto.

A todos los profesores de la Licenciatura en Informática, del Núcleo de Sucre de la Universidad de Oriente, con especial énfasis a los profesores Lisbeth Fernández, Dianelina Aguiar y Eugenio Betancourt por asumir la tarea de emprender los Cursos Especiales de Grado.

Gracias por todo.

LISTA DE TABLAS

Página

Tabla 1. Versión del sistema para el módulo Administrador. 71

Tabla 2. Versión del sistema para el módulo Prestador de Servicios. 72

Tabla 3. Versión del sistema para el módulo Turista. 73

Tabla 4. Especificación del actor Turista, roles y actividades que desempeña. 89

Tabla 5. Especificación del actor Prestador de Servicios, sus funciones y actividades que desempeña..... 90

Tabla 6. Especificación del actor Prestador de Servicios, sus funciones y actividades que desempeña..... 91

Tabla 7. Lista de Requisitos..... 95

Tabla 8. Clasificación de requisitos funcionales y no funcionales..... 96

Tabla 9. Riesgos identificados del proyecto Softour. 143

Tabla 10. Análisis de riesgos. 144

Tabla 11. Descripción Caso de uso: Ingresar Lugar por usuario Administrador 163

Tabla 12. Clases de equivalencia prueba Ingresar Lugar. 175

Tabla 13. Casos de prueba para Ingresar Lugar 176

Tabla 14. Descripción Caso de uso Modificar Evento por usuario prestador de servicio. 181

Tabla 15. Descripción Caso de uso Consultar Alojamiento por usuario Turista. 197

LISTA DE FIGURAS

Figura 1. Sistemas de Información Gerenciales. 21
Figura 2. Simbología de Casos de Uso. 25
Figura 3. Representación gráfica de una clase en UML. 26
Figura 4. Ejemplo de Diagrama de Clases. 28
Figura 5. Notación de Diagrama de Secuencias. 29
Figura 6. Notación MVC del Diagrama de Secuencias. 30
Figura 7. Arquitectura Cliente Servidor de tres (03) capas. 31
Figura 8. Modelo de 4 + 1 Vistas de Kruchten. 34
Figura 9. Componentes del Método Watch. 41
Figura 10. Jerarquía de productos del Método Watch. 42
Figura 11. Clasificación de Actores. 45
Figura 12. Roles de los desarrolladores requeridos por Blue Watch. 46
Figura 13. Cadena de Valor de los procesos del Método Watch. 47
Figura 14. Procesos del Método Watch. 47
Figura 15. Estructura del Modelo de Procesos. 50
Figura 16. Proceso de Instanciación del Método Watch. 52
Figura 17. Cadena de Valor de los procesos del Método Watch. 53
Figura 18. Relación entre procesos y productos del Método Watch. 54
Figura 19. Procesos del Ciclo de la Aplicación. 55
Figura 20. Procesos relacionados con el Ciclo de Versión. 56
Figura 21. Cadena de Valor del Ciclo de Versión. 57
Figura 22. Descripción de procesos de Diseño Detallado del Ciclo de Versión. 57
Figura 23. Procesos relacionados con el Ciclo de Incremento. 58
Figura 24. Descripción de Desarrollo del Ciclo de Incrementos. 58
Figura 25. Modelo principal de procesos del marco metodológico Blue Watch. 59
Figura 26. Diagrama de análisis de Inicio del Proyecto. 60
Figura 27. Diagrama de actores y estructura del equipo de desarrollo. 62
Figura 28. Diagrama de análisis de Planificación Inicial del Proyecto. 64

Figura 29. Diagrama de análisis del Contexto de la Aplicación. 75
Figura 30. Cadena de valor de los procesos de Softour. 81
Figura 31. Diagrama del proceso Agregar Lugar. 82
Figura 32. Diagrama de actividad del proceso Agregar Lugar. 83
Figura 33. Diagrama inicial de clases de objeto. 84
Figura 34. Diagrama del modelo de actores del método Blue Watch. 86
Figura 35. Diagrama de Actores. 92
Figura 36. Diagrama de desarrollo inicial de requisitos. 93
Figura 37. Diagrama de diseño inicial de la aplicación. 98
Figura 38. Casos de Uso general del perfil Administrador. 100
Figura 39. Caso de Uso Administrar Lugares. CRUD Lugares. 101
Figura 40. Casos de Uso reducido del perfil Prestador de Servicios. 102
Figura 41.Caso de Uso Administrar Lugares. CRUD Lugares. 102
Figura 42. Diagrama de Clases asociada al Caso de Uso Ingresar Lugar. 104
Figura 43. Diagrama de Clases asociada al Caso de Uso Consultar Lugar... 105
Figura 44. Diagrama de Clases asociada al Caso de Uso Modificar Lugar. .. 106
Figura 45. Diagrama de Clases asociada al Caso de Uso Eliminar Lugar. 107
Figura 46. Diagrama de Clases del Módulo Alojamiento del Sistema Softour. 108
Figura 47. Diagrama de Clases del Módulo Consultar Lugar del Sistema Softour. 109
Figura 48. Diagrama de Clases del Módulo Consultar Evento del Sistema Softour. 110
Figura 49. Diagrama de Clases del Módulo Consultar Gastronomía del Sistema Softour. 111
Figura 50. Diagrama de Clases del Módulo Consultar Experiencia del Sistema Softour. 112
Figura 51. Administrar patrones de búsqueda de eventos. 114
Figura 52. Diagrama de Secuencias Consultar Detalle de Alojamiento. 114
Figura 53. Diagrama de Secuencias Consultar Eventos Turista. 115
Figura 54. Diagrama de Componentes. 116
Figura 55. Diagrama de Despliegue. 117
Figura 56. Diagrama de planificación de la iteración i-ésima. 119

Figura 57. Diagrama de refinamiento de requisitos del incremento i-ésimo... 121
Figura 58. Diagrama de diseño detallado del incremento i-ésimo..... 122
Figura 59. Diagrama de codificación del incremento i-ésimo. 124
Figura 60. Diagrama de pruebas del incremento i-ésimo..... 127
Figura 61. Diagrama de verificación y validación de incrementos..... 128
Figura 62.Diagrama de entrega final de la aplicación. 130
Figura 63. Diagrama de cierre del proyecto. 132
Figura 64. Diagrama de subprocesos de gestión del proyecto..... 133
Figura 65. Diagrama de dirección del proyecto. 134
Figura 66. Diagrama de seguimiento y control del proyecto. 135
Figura 67. Diagrama de gestión de requisitos. 136
Figura 68. Diagrama de verificación y validación estática de productos. 138
Figura 69. Diagrama de gestión de la configuración del software. 140
Figura 70. Diagrama de gestión de riesgos..... 141
Figura 71. Diagrama de gestión del proyecto de desarrollo de software..... 146
Figura 72. Diagrama de gestión del proyecto de desarrollo de software..... 147
Figura 73. Interfaz perfil Administrador: Vista general Agregar Lugar..... 149
Figura 74. Interfaz perfil Administrador: Vista Agregar Lugar con Posicionamiento en el mapa geográfico satelital. 149
Figura 75. Interfaz perfil Administrador: Vista Agregar Lugar Subir Foto. 150
Figura 76. Interfaz perfil Administrador: Vista Agregar Lugar Subir Foto de Portada. 150
Figura 77. Interfaz perfil Prestador de Servicios: Vista general Editar Evento. 151
Figura 78. Interfaz perfil Prestador de Servicios: Vista Editar Evento para modificar datos de persona contacto. 151
Figura 79. Interfaz perfil Prestador de Servicios: Vista Editar Evento para añadir nueva foto de álbum..... 152
Figura 80. Interfaz perfil Prestador de Servicios: Vista Editar Evento para elegir nueva foto de portada. 152
Figura 81. Interfaz perfil Turista: Vista principal de Consultar Eventos. 153
Figura 82. Interfaz perfil Turista: Vista detalle de Consultar Evento..... 154
Figura 83. Caso de uso: ingresar lugar por usuario Administrador 162

Figura 84. Diagrama de clases: Ingresar Lugar por usuario Administrador. .. 164

Figura 85. Diagrama de Secuencia: Registrar Lugar por usuario Administrador 166

Figura 86.Colección de lugares: ingresar lugar por usuario Administrador. ... 167

Figura 87. Plantilla de interfaz 1: Ingresar Lugar por usuario administrador. . 169

Figura 88. Plantilla de interfaz 2: Ingresar Lugar por usuario administrador. . 170

Figura 89. Plantilla de interfaz 3: Ingresar Lugar por usuario administrador. . 170

Figura 90. Plantilla de interfaz 4: Ingresar Lugar por usuario administrador. . 171

Figura 91. Interfaz 1 de proceso de agregar lugar por usuario Administrador. 172

Figura 92. Interfaz 2 de proceso de agregar lugar por usuario Administrador. 172

Figura 93. Interfaz 3 de proceso de agregar lugar por usuario Administrador. 173

Figura 94. Interfaz 4 de proceso de agregar lugar por usuario Administrador. 173

Figura 95. Código desarrollado para el proceso de agregar lugar por usuario Administrador. 174

Figura 96. Caso de prueba A1 para Agregar Lugar. 177

Figura 97. Caso de prueba A1 para Agregar Lugar. 177

Figura 98. Caso de prueba A1 para Agregar Lugar. 178

Figura 99. Caso de prueba A1 para Agregar Lugar. 179

Figura 100. Caso de uso: Modificar Evento por prestador de servicio. 180

Figura 101. Diagrama de Clases: Modificar Evento por usuario Prestador de Servicios..... 182

Figura 102. Diagrama de secuencia: Modificar Evento por usuario Prestador de Servicio. 183

Figura 103. Imagen Colección de Evento: Modificar evento por usuario Prestador de Servicio. 184

Figura 104. Plantilla de interfaz 1: Modificar Evento por usuario prestador de servicio. 185

Figura 105. Plantilla de interfaz 2: Modificar Evento por usuario Prestador de Servicio. 186

Figura 106. Plantilla de interfaz 3: Modificar Evento por usuario Prestador de Servicio. 187

Figura 107.Plantilla de interfaz 3: Modificar Evento por usuario Prestador de Servicio. 188

Figura 108. Interfaz 1 de proceso de Modificar Evento por usuario Prestador de Servicio. 189

Figura 109. Interfaz 2 de proceso de Modificar Evento por usuario Prestador de Servicio. 190

Figura 110. Interfaz 3 de proceso de Modificar Evento por usuario Prestador de Servicio. 191

Figura 111. Interfaz 4 de proceso de Modificar Evento por usuario Prestador de Servicio. 191

Figura 112. Código desarrollado para el proceso Modificar Evento por usuario Prestador de Servicio. 192

Figura 113. Modificación de información de un evento por Prestador de Servicio. 194

Figura 114. Prueba de Modificar Evento por parte del usuario Prestador de Servicio. 194

Figura 115. Prueba finalizada de Modificar Evento por usuario Prestador de Servicio. 195

Figura 116.Caso de uso: Consultar Alojamiento por usuario Turista. 196

Figura 117. Diagrama de Clases: Consultar Experiencia por Usuario Turista. 198

Figura 118. Diagrama de secuencia: Consultar Experiencia por usuario Turista. 199

Figura 119. Plantilla de interfaz general: Consultar Experiencia por usuario Turista. 200

Figura 120. Plantilla de interfaz especifica: Consultar Experiencia por usuario Prestador de Servicio. 201

Figura 121.Interfaz general de proceso de Consultar Experiencia por usuario Turista 202

Figura 122. Interfaz especifica de proceso de Consultar Experiencia por usuario Turista 203

Figura 123. Código desarrollado para el proceso Consultar Experiencia por usuario Prestador de Servicio. 204

Figura 124. Prueba consultar experiencias por Turista 204

Figura 125. Prueba Consultar Experiencia por usuario Turista. 205

RESUMEN

Se desarrolló un sistema de información Web para el incentivo, promoción y apoyo al sector turismo, caso de estudio Softour, ofreciendo las funcionalidades para los perfiles de usuarios administrador, prestador de servicios y turista, con aspectos relevantes para la gestión y aprovechamiento de regiones y países en las que se deseen resaltar lugares, eventos, experiencias, alojamientos y gastronomías, brindando mecanismos de búsquedas avanzadas, una interfaz visual acorde a los requerimientos de los nuevos paradigmas de las aplicaciones turísticas, creando una asociación entre el prestador del servicio y el turista de ganar/ganar, manejando la retroalimentación de la calificación y el compartir las experiencias vividas, optimizando los tiempos de respuesta a través del manejo de colecciones de datos, programación versátil en .js. Se tomó como referencia la metodología WATCH (Montilva, Barrios y Rivero, 2008) en su instanciación del método *Blue WATCH* para la implementación de los procesos, productos y actores (Montilva y Barrios, 2014). Durante el ciclo de aplicación se realizó un Modelado de Negocios empleando notación UML, el Desarrollo de Requisitos, el Diseño Arquitectónico; en el ciclo de versión se ejecutaron actividades del ciclo anterior anexando el Diseño Detallado; finalmente, en el ciclo de incremento se realizó la construcción de las interfaces utilizando herramientas como Bootstrap, HTML, CSS, JQuery, Handlebars y JavaScript; la creación de la base de datos en MongoDB orientada hacia las colecciones de datos no estructuradas, conjuntamente con la librería de Mongoose para asociarla hacia la orientación a objetos del modelo; la codificación bajo el lenguaje de programación Node.js, Sublime y Postman; para el control de versiones Git y trabajo colaborativo en equipo GitHub. Cabe destacar que los ciclos de desarrollo del proyecto fueron controlados y administrados mediante los procesos de gestión y soporte compuestos por la gestión de proyecto, de riesgos, de requisitos, de la configuración, planificación y control de versiones y el plan de verificación y validación, en los cuales se resaltó el uso de las herramientas Trello, Scrum y Slack. El sistema obtenido ofrece a las organizaciones turísticas, los prestadores de servicios, los destinos turísticos y a los turistas, la intermediación, asociación y capacidad para incentivar, promocionar y apoyar al sector turismo, utilizando herramientas de punta, para conectarlas con el universo virtual y hacerlas más competitivas, adentrando en conceptos de innovación y calidad, sin olvidar la sencillez en el acceso, el uso y la comprensión de lo que se ofrece. Se consideró inicialmente un caso de prueba para el Estado Sucre, pero todas las funcionalidades descritas en la presente investigación pueden ser adaptadas a otras zonas turísticas ya que no existen limitaciones considerables y el sistema se desarrolló con una visión de uso universal, con la posibilidad de visualizar la aplicación en los idiomas español e inglés.

Palabras Claves: Turismo, Sistema de Información, Método Watch, Node.js, MongoDB, Lugares, Eventos, Experiencias, Alojamientos, Gastronomía.

INTRODUCCIÓN

El surgimiento de nuevas tecnologías ha provocado un desarrollo exponencial en la capacidad de comunicación de las civilizaciones y los consecuentes cambios provocados por su aplicación práctica, abren grandes posibilidades para el futuro y crean nuevas necesidades en la sociedad. Es el caso de las Tecnologías de la Información y la Comunicación (TIC), que emergen y evolucionan con un gran impacto en las diversas áreas de las organizaciones, se encargan del estudio, desarrollo, implementación, almacenamiento y distribución de la información mediante la utilización de hardware y software; ayudan a comunicarnos e intercambiar conocimientos al “eliminar” distancias geográficas y tiempo, potenciando las actividades humanas. Las organizaciones que ignoren estos cambios, dentro de sus estrategias, podrán perder importantes ventajas competitivas e inclusive desaparecer o volverse obsoletas (Valdes, 2002).

La necesidad de compartir, procesar y analizar volúmenes de información se incrementa cada día; y es a partir de esta necesidad que nacen los sistemas de información, los cuales, a través de aplicar conceptos, principios, técnicas de análisis y diseño, permiten hacer eficiente la toma de decisiones (Amaya, 2010).

Hoy en día, las organizaciones cuentan con un gran número de tecnologías para el manejo de la información de una manera confiable, económica y eficiente, como lo son las redes de computadoras y dispositivos asociados, configurados para compartir recursos con ayuda de interfaces especiales, protocolos y dispositivos de hardware. La Internet, como método de interconexión descentralizada de redes de computadoras implementado en un conjunto de protocolos denominado TCP/IP, que garantiza que redes físicas heterogéneas funcionen como una red lógica única, de alcance mundial, brinda el servicio *World Wide Web*, en donde un usuario visualiza sitios Web, que pueden contener texto, imágenes u otros contenidos multimedia, y navega a

través de ellas usando hiperenlaces (Carrillo, 2014).

Un sitio Web, por sí solo, no satisface los requerimientos de las organizaciones actuales, ya que menosprecia la utilización y el rendimiento de las innovaciones tecnológicas, por ello ha evolucionado a aplicaciones Web cada vez más complejas en términos de estructura, funcionalidad e interfaz. El alcance y la complejidad de estas aplicaciones varían extensamente y pueden ser desde servicios de escala reducida hasta aplicaciones empresariales distribuidas en la Web.

El crecimiento desmedido de las nuevas tecnologías estableció un efecto en cadena que no excluyó a nada, ni a nadie, actualmente encontramos un sinfín de procesos automatizados o negocios que dejaron de trabajar de forma física a trabajar de forma virtual, afectando de diferentes formas la economía; la mayoría de la empresas y negocios han pasado a una automatización de sus procesos, llevándolos a sistemas de información gerenciales, los cuales cumplen diferentes funciones dentro de las instituciones (De Pablos *et al.*, 2006).

Todo éste despertar tecnológico ha llevado a muchas empresas, organizaciones e instituciones venezolanas e internacionales del sector turismo a utilizar las ventajas que ofrecen los sistemas de información bajo ambiente Web para el mejoramiento de sus procesos, asignando cada vez mayor importancia a la inversión en el desarrollo de nuevos sistemas que incorporan el uso de estas tecnologías emergentes.

Es por ello que las tres grandes olas de transformación de la industria turística están relacionadas con el uso y la evolución de las tecnologías de la información, las cuales tienen lugar en la primera década del siglo XXI, se basan en la creación y desarrollo de modelos de negocio que con su capacidad de subvertir el orden establecido en el sector, consigue alterar sus paradigmas y con ellos el mapa de actores turísticos, locales, regionales y globales, con un

común denominador: el acceso de productos, prestadores de servicios y clientes a un, cada vez más abundante y libre, caudal de información sobre productos, destinos, servicios, costes y precios (Martínez, 2012).

La primera avalancha de cambio surge en los años 90 sobre las agencias de viajes online con la agregación de productos reservables y su mercadeo, ponen en manos de viajeros un poder que no tenían, mientras que las líneas aéreas de bajo coste proporcionaban su propio canal en Internet, con distribución y venta directa.

La segunda avalancha transformadora ocurre a partir del 2005 con la Web 2.0 o web social, como evolución de las relaciones continuas y multidireccionales de las plataformas sociales como Facebook, Twitter, Flickr o Youtube, brindando acceso a la información generado por las empresas y complementado ampliamente por las experiencias generadas por los usuarios, siendo cada usuario auditor y prescriptor de productos y servicios. Saliendo victoriosos los foros de opinión, los metabuscadores, las comunidades de viajeros y los creadores de contenido social de las plataformas mencionadas.

Finalmente se levanta una tercera gran ola en forma de teléfonos inteligentes y tabletas, que harán posible la conectividad permanente, el acceso a la información y a las personas en tiempo real, con la participación activa de Google, Apple, Facebook, Microsoft, entre otros, en la implementación de nuevas formas de buscar, diseñar, decidir, planificar, reservar y compartir el ocio y viajes de los turistas.

Es por ello que en el sector turismo, la globalización obliga a ser competitivos y a estudiar alternativas para su desarrollo local, regional, nacional e internacional, de allí que las ventajas comparativas y competitivas deben ser punto de partida de cualquier programa, plan o proyecto de desarrollo de software o sistemas de información relacionado con el sector.

Por lo expuesto anteriormente, se debe conocer y profundizar en los nuevos paradigmas de la calidad de la empresa turística actual, considerando los cambios que se producen en la sociedad y en el sector turismo, que redefinen y alteran las reglas de negocio. Por ello es importante entender las organizaciones turísticas, los prestadores de servicios, los destinos turísticos, la intermediación y la relación con los clientes, utilizando las nuevas herramientas de trabajo para la empresa turística, conectándolas con este nuevo universo virtual y haciéndolas más competitivas, adentrando en conceptos de innovación y calidad, sin olvidar la sencillez en el acceso, el uso y la comprensión de lo que se ofrece.

Por tal razón se busca desarrollar un sistema de información bajo ambiente web para el incentivo, promoción y apoyo al sector turismo, a fin de automatizar, agilizar y facilitar el registro de información y obtención de datos necesarios en todos y cada uno de los procesos involucrados a nivel del administrador, prestador de servicios y, muy especialmente, para el turista, mejorando así el tiempo de consulta y respuesta.

El presente trabajo de grado está estructurado en los siguientes tres capítulos:

Capítulo I. Presentación.

Está dividido en dos secciones: planteamiento del problema, donde se describe la problemática presentada en el área de estudio, determinadas mediante la investigación preliminar y obteniendo una visión básica del producto, describiendo los motivos por los cuales se realizó el trabajo y su importancia en la solución del problema; luego se desarrolla el alcance que describe lo que cubrió el presente trabajo, las limitaciones que se presentaron en su desarrollo, incluyendo los aspectos que por alguna razón no contempla el sistema.

Capítulo II. Marco de referencia

Consta de dos secciones: La primera es el marco teórico, donde se presentan

los fundamentos teóricos que soportan este trabajo, quien a su vez, está conformada por: antecedentes de la investigación y de las organizaciones, área de estudio y área de la investigación. La segunda sección es el marco metodológico, donde se describe la metodología utilizada en el trabajo y consta de dos partes: la metodología de la investigación, donde se indica el nivel y el tipo de investigación, además las técnicas e instrumentos de recolección de datos, y la metodología del área aplicada utilizada para desarrollar el trabajo.

Capítulo III. Desarrollo.

Presenta los resultados arrojados en la elaboración del trabajo de grado, resaltando los procesos de gestión, para la gestión del proyecto; los procesos de soportes, para la gestión del riesgo y gestión de la calidad; los procesos técnicos, que considera el modelo de negocios, la ingeniería de requisitos, el diseño de software, el aprovisionamiento de componentes, ensamblaje del sistema de software, pruebas del sistema de software, entrega del sistema de software, en todo ello inmerso los diagramas UML de casos de uso, de clases, de estados, de secuencias, la arquitectura de la aplicación, el diseño de las pantallas, la jerarquía de las pantallas, el modelo conceptual de la base de datos, el esquema físico de la base de datos, la arquitectura de los componentes o módulos del sistema, los procedimientos de la administración de la base de datos, la programación de los componentes, la integración de los diferentes componentes, el desarrollo de las diferentes pruebas, depuración de errores, la implementación de la aplicación, y la documentación. Todo esto según la metodología escogida.

A sí mismo, se presentan las conclusiones, recomendaciones, bibliografía y los apéndices que contemplan el contenido del presente trabajo.

CAPÍTULO I. PRESENTACIÓN

1.1. PLANTEAMIENTO DEL PROBLEMA

Según la Organización Mundial del Turismo, el turismo consiste en los viajes y estancias que realizan personas en lugares distintos a su entorno habitual, al menos durante una noche y como máximo 365 días, ya sea por ocio, aventuras, negocios u otros motivos (Sancho, 1998).

En 1942, Walter Hunziker y Kart Krapf definen el turismo como el conjunto de relaciones y fenómenos que se producen como consecuencia del desplazamiento y estancia temporal de personas fuera de su lugar de residencia, siempre que no esté motivado por razones lucrativas, se puede notar que contradice con la definición dada por la Organización Mundial del Turismo al decir que no tiene nada que esté motivado a razones lucrativas, pero esto no es del todo cierto, ya que inclusive quien viaje por razones de negocios necesita datos de interés de la zona a la cual tiene planeado viajar (Interviajes, s.f.).

El turismo es una actividad que cada individuo interpreta de acuerdo a sus intereses personales, lo cual se ve influido por diferentes factores como lo son vivencias, cultura, idioma, gustos, motivos, situación económica, por ende encasillar el turismo en una definición es algo que se torna complejo (Ferrari, Montero, Mondéjar y Vargas, 2013).

El turismo es considerado un sector con grandes posibilidades para generar desarrollo, riqueza, bienestar y empleo; convirtiéndose en una de las actividades más afectadas por la globalización: con acceso a la información a velocidades impresionantes y, por ende a los mercados, modificando los patrones de comportamientos de los potenciales turistas, dado que no solo se compite en costos, sino también en calidad, en la relación precio-valor, en responsabilidad social y presencia en los mercados internacionales. El turista

exige un servicio de calidad, respuestas oportunas, condiciones de seguridad que el destino turístico debe proporcionarle. Ya no basta con atractivos naturales o factores heredados para competir, se deben seleccionar segmentos específicos y especializarnos en él para vender en los mercados mundializados (Duque, 2005).

El impacto del turismo sobre los sistemas económicos es un tema que desde hace varios años interesa a los investigadores, analistas económicos y responsable de las políticas de desarrollo, tanto en los sectores públicos como privados. La multisectorialidad que caracteriza la actividad turística juega un papel fundamental tanto en las economías nacionales como en las regionales para activar, potenciar y estimular los sectores. Una correcta valoración de la incidencia de la actividad turística necesita instrumentos metodológicos que permitan una completa y extensa identificación de las múltiples interrelaciones existentes en el interior de un sistema económico (Ferrari *et al.*, 2013).

Las personas al momento de planear un viaje necesitan una gran cantidad de información que se podrían definir como elementos que componen la actividad turística, dicha información ayudará a decidir el lugar donde se alojará, las actividades que realizará, los eventos a los que asistirá, las experiencias en la que querrá participar, así como también el tipo de transporte que utilizará y los servicios asociados que le puedan ayudar a tomar la mejor decisión; en la mayoría de los casos esta información se encuentra dispersa y los turistas deben hacer un estudio y comparación de toda esa información, lo que se convertiría en unos de los principales problemas a resolver.

Según levantamientos previos de información, en Venezuela no se ha desarrollado un sistema de información del sector turismo que se considere sólido, que se imponga, que sea competitivo con los sistemas existentes en el mercado global. La mayoría de los sistemas nacionales conocidos se mantienen en un solo nicho, es decir, se centran en hospedaje, en regiones o

en actividades, y en su mayoría están patrocinados por agencias de viajes y hoteles locales, ya que no se globalizan. Estos aspectos, además de representar algunos problemas a considerar, se han transformado en oportunidades que hemos decidido aprovechar para desarrollar una aplicación que incentive, promocione y apoye los diferentes tipos de turismo, dando soporte y resaltando a cada uno de ellos.

El turismo en las áreas naturales protegidas debe considerar las normas de conservación de los recursos naturales y culturales, para garantizar se cumplan las expectativas económicas, de empleo e ingresos; las expectativas socioculturales, de participación comunitaria para la inclusión de diferentes géneros y estratos sociales; y expectativas ambientales que radica en el valor, el cuidado y la protección que se le da a los recursos naturales como playas, ríos, montañas, llanuras, bosques, selvas, desiertos y áreas rurales, entre otros, promocionando actividades sin deteriorar el entorno. También surgen propuestas en donde se incorporan plantas y animales, la flora y la fauna, para su contemplación fuera de sus hábitats naturales, como los casos de jardines botánicos y zoológicos (Díaz y Sastre, 2011).

El turismo religioso debe considerar los desplazamientos masivos de personas a los lugares de culto religioso, conformados a su vez por el patrimonio cultural-arquitectónico en recintos destinados al culto religioso, como por ejemplo la Catedral de Santiago de Compostela, la cual es uno de los tres grandes centros de peregrinación cristiana desde la época del medievo (Roma, Jerusalén y Santiago), estos viajes motivados por la fe han dado lugar a un turismo de peregrinación o religioso, creándose rutas de peregrinación como el Camino Francés, que es la ruta más conocida para llegar a Santiago (Millán, Morales y Pérez, 2011).

El Turismo religioso, presenta como motivación fundamental la fe. En la actualidad las ciudades Santas que históricamente han sido objeto de

peregrinaje, como Jerusalén, La Meca o Roma, siguen siendo importantes puntos de referencia del sector. El sector del turismo religioso está implantado con fuerza en Europa, donde se calcula que más de quince millones de personas realizan algún tipo de turismo religioso, y en América Latina, por lo que varias agencias turísticas se han especializado en este sector (Turismo Religioso, s.f.).

En el caso del turismo de negocios se debe considerar que se desarrolla con el objeto o fin de llevar a cabo un negocio o un acuerdo comercial, por lo general entre empresas. Es utilizado por empresarios, ejecutivos, comerciantes y otros profesionales para cerrar negocios, captar clientes o prestar servicios. Debe tenerse en cuenta que la estacionalidad es invertida a la vacacional, por lo que es un producto muy importante para el sector turismo. El cliente suele ser de alto poder adquisitivo, con interés en lo urbano y con necesidades de infraestructura muy concretas como la conexión a internet y las áreas donde pueda desarrollar sus negocios (Turismo de Negocios, s.f.).

El turismo idiomático ha adquirido una relevancia cada vez mayor en el ámbito de la economía turística, resaltando aquellos lugares para el aprendizaje de idiomas como el inglés, francés y español, entre otros. Por esta razón, se debe considerar una visión de las principales características de este tipo de turismo, tanto por el lado de la concentración espacial de la oferta como por el de la evaluación de la demanda, tanto potencial como actual, proponiendo medidas que permitan fomentar tan incipiente, pero lucrativo, segmento turístico (Montero, Fernández, Higueras y García, 2010).

El turismo cultural aborda las posibilidades que ofrece diferentes renglones relacionados con las costumbre, historia, acontecimientos, literatura, poesía, entre otros aspectos, al considerarlo las piezas de ese complejo rompecabezas denominado patrimonio cultural, de manera que sea posible organizar estrategias para el desarrollo de una oferta turística cultural que, en unos casos,

venga a sumarse a otras facetas ya trabajadas en el destino, complemente otras modalidades de oferta turística o abra un nuevo campo de acción para el territorio y la geografía implicada con esta visión de turismo cultural. Se apuntan las principales estrategias asociadas al uso del patrimonio cultural dentro de una dimensión económica y se esbozan sus posibilidades dentro de esa misma dimensión (Magadan y Rivas, 2010).

El turismo gastronómico se presenta como una nueva opción para adentrarse en el mundo culinario donde el turista podrá experimentar miles de los sabores que caracterizan a una región y el cual estará dirigido tanto a personas conocedoras como son los chefs, personas que trabajan en esta área, así como a todo tipo de personas dispuesta a tener una nueva experiencia de sabores. Los objetivos principales que persigue el turismo gastronómico se sintetizan en: conocer y experimentar nuevas comidas de otras regiones, dar a conocer las especialidades gastronómicas de las regiones a los visitantes, experimentar nuevos sabores y, conocer los diferentes ingredientes y formas de preparación de la gastronomía de la zona (Flavián y Fandos, 2011).

El turismo sustentable utiliza mecanismos para medir su desempeño, cualitativa y cuantitativamente, a través de sus operaciones, los ejemplos más resaltantes son: el turismo creativo, vinculado a la realización de actividades artísticas y creativas en el lugar de destino, como exposiciones de pintura y escultura, festivales de cine y teatro, conciertos musicales, espectáculos operísticos, entre otros; el turismo de compras, vinculado a las adquisiciones de carácter exclusivo o a buen precio, incluye artículos de lujo, arte, artesanía y artículos comerciales de uso común como calzado, electrónica, vestido, adornos, entre otros; el turismo urbano, desarrollado principalmente en ciudades Patrimonio de la Humanidad o en aquellas de alto desarrollo económico y social, los clientes son de nivel cultural y poder adquisitivo alto, es el tipo de turismo masivo más grande del mundo (Turismo Sustentable, s.f.).

El anterior análisis busca comprender a las organizaciones turísticas, los diferentes tipos de turismo, a su entorno y a su modelo de negocios, para relacionar los conceptos y los requerimientos turísticos con el uso de la tecnología de la información y comunicación, que permita a los prestadores de servicios, a los destinos turísticos y a los turistas, la intermediación y la relación, utilizando herramientas de trabajo para conectarlas con este universo virtual y hacerlas más competitivas, adentrando en conceptos de innovación y calidad, sin olvidar la sencillez en el acceso, el uso y la comprensión de lo que se ofrece.

En este escenario competitivo, hemos considerado como una estrategia clave de la innovación del destino a la asociación con el concepto de la experiencia que los turistas pueden llegar a vivir en él y que, posteriormente, comparten con otros turistas del mundo a través de nuestra aplicación Web. En este sentido, hemos analizado los sitios web con diferentes destinos, el perfil del cliente y sus características, y algunos ejemplos en este ámbito que ya aplican lo que se denomina la web experiencial. Los resultados obtenidos nos permitirán desarrollar una aplicación Web que considere las experiencias.

Por ello, la propuesta de desarrollo del sistema va más allá de una aplicación para patrocinar o publicitar el turismo de una región, se plantea elaborar una aplicación que pueda ser utilizada en cualquier región del mundo y que además suministre información valiosa para la toma de decisiones en lo que se refiera a mejorar la actividad turística en la región donde se implemente. Por ello se deben considerar los diferentes intereses relacionados con el turismo, todos aquellos aspectos posibles que puedan aportar los diferentes prestadores de servicios en los diferentes ámbitos donde se desenvuelven.

Es necesario involucrar en el desarrollo de la aplicación algunos elementos primordiales: como el alojamiento u hospedaje, en hoteles, posadas y cabañas, que además permitan disfrutar un merecido descanso; incentivando la

promoción de los diferentes eventos que tienen lugar en las localidades, pudiendo ser de tipo académico, social, cultural, político o cualquier otro; resaltando las vivencias de los turistas, el disfrute de diferentes aventuras, sitios o actividades, haciéndonos partícipes de sus experiencias; publicitando la gastronomía, presentando platos típicos nacionales e internacionales, según el lugar destino; informando sobre el transporte, el clima, la cultura, la religión; ya que todos estos elementos son valiosos para las personas que desean emprender un viaje.

Basándose en las ideas propuestas y las exigencias establecidas en las investigaciones realizadas, los cursantes de la primera cohorte de Cursos Especiales de Grado plantearon desarrollar un sistema de información bajo ambiente Web para el incentivo, promoción y apoyo al sector turismo, cuyo caso de estudio se ha denominado Softour.

La aplicación Softour es una propuesta para desempeñarse como un sistema de calidad en la actividad turística, que ofrecerá información de interés a sus usuarios Turistas, como recomendaciones o tendencias de las reservaciones más solicitadas, así como también para los usuarios Administrador y Prestador de Servicios que les brindará información relacionada con la inteligencia de negocios para apoyarlos en la toma de decisiones oportunas, que aporten un constante crecimiento e incentive al desarrollo.

1.2. ALCANCE Y LIMITACIONES.

1.2.1. Alcance

A nivel de investigación, se planteó el objetivo de identificar y comprender la secuencia de pasos proporcionados por la metodología de desarrollo de software WATCH (Montilva *et al*., 2008), desarrollando una serie de actividades, técnicas y notaciones propuestas que permitieron desarrollar los productos finales en cada una de las iteraciones realizadas dentro de cada proceso

técnico, de gestión y soporte. En los procesos de gestión y soporte se obtuvo el documento del plan del proyecto, la lista de chequeo de riesgos y el informe de validación y verificación; en la fase de análisis, compuesta por los procesos técnicos de modelado de negocio e ingeniería de requisitos, se obtuvo el modelado de negocios del sistema de información bajo ambiente web propuesto para el sector turismo, el listado de requisitos del cliente, listado de requisitos técnicos, planillas de Volére de requisitos, que ayudaron a especificar la lista de requerimientos funcionales y no funcionales que satisface dicha aplicación, se desarrolló el modelado del sistema en UML a través de los diagrama de casos de uso, las descripciones de las especificaciones de los casos de uso, la representación de las clases con sus atributos y métodos, el diagrama de clases persistentes, los diagramas de secuencias por cada caso de uso y diagramas de estado; en la fase de diseño de software, integrada por los procesos técnicos, diseño arquitectónico y diseño detallado, se elaboró el listado descriptivo de las metas de diseño, la estructura de la aplicación, el diseño de pantallas, el diseño jerárquico de pantallas, el modelado conceptual de la base de datos, el esquema físico de la base de datos, se definieron los componentes o módulos y se establecieron los procedimientos de administración de la base de datos; en la fase de desarrollo, en la que resalta el aprovisionamiento de componentes, se estableció la plataforma de desarrollo instalada, con manuales y procedimientos de instalación de software, los componentes adquiridos en su mayoría de carácter abierto, los componentes adaptados, que se basaron en su reutilización, los componentes desarrollados, según sus algoritmos y estándares de codificación, los componentes y módulos probados con las especificaciones de casos de pruebas como caja negra, prueba de hilos y depuración de errores; en la fase de despliegue de la aplicación, se llevaron a cabo los procesos técnicos de ensamblaje y pruebas del sistema de software, obteniendo la interfaz de usuario probada con la aplicación de técnicas y estrategias ejecutadas con las especificaciones de casos de prueba de interfaz gráfica y la depuración de los errores, la capa de

lógica de negocios de la aplicación probada a través de técnicas y estrategias de pruebas de integración de componentes o módulos utilizando especificaciones de casos de pruebas y la depuración de los errores, la base de datos probada a través de la aplicación de técnicas y estrategias de pruebas de bases de datos utilizando casos de pruebas y la depuración de los errores, la aplicación integrada y probada a través de la aplicación de técnicas y estrategias de pruebas de aplicaciones distribuidas utilizando especificaciones de casos de pruebas y la depuración de los errores, finalmente obteniendo la aplicación probada y depurada; en la fase de implementación, resalta el proceso de entrega del sistema de software, donde se desarrolló el plan de instalación, la plataforma de operación instalada, la base de datos actualizada, manuales de la aplicación, material de capacitación.

La meta principal fue el desarrollo de un sistema de información bajo ambiente web para incentivar, promocionar y apoyar al sector turismo enfocado inicialmente en el Estado Sucre, resaltando una serie de beneficios, como : Crear en minutos una página de registro de lugares, eventos, experiencias, alojamientos y gastronomía para los prestadores de servicios interesados en promocionarse y promocionar al estado, publicando los diferentes lugares turísticos de la zona, paseos guiados, cómodos alojamientos, platos gastronómicos locales y tradicionales, nacional e internacional, para el deguste del turista, eventos, conferencias, festivales, clases especializadas de alguna actividad deportiva, cultural, académica o de otra índole, con el acceso a diferentes guías y folletos que facilitarán la visita a la localidad para una mejor ubicación del turista; permitiendo a los turistas elaborar itinerarios de acuerdo a su perfil, con acceso en línea a través de las redes sociales con mayor popularidad en la actualidad, utilizando una tecnología de vanguardia, simplificada, que ayuda a poner en marcha diferentes tipos de emprendimiento, compatible con cualquier tipo de dispositivo.

El sistema contemplaría información de interés para los usuarios turistas

acompañados imágenes y descripciones asociadas, además de permitir una valoración y comentario sobre el elemento, obteniendo el sistema de esta forma información valiosa para su retroalimentación, con las valoraciones y las visitas de los elementos contemplados en el sistema, se tomarían datos para realizar estadísticas que vería el administrador del sistema, obteniendo así una herramienta que le permita hacerle mejoras a los elementos que publicita.

Se presentan tres módulos, el primer módulo es el del administrador que tiene acceso para gestionar los prestadores de servicios, lugares, eventos, experiencias, gastronomía y alojamientos; el segundo módulo es el del prestador de servicios que tiene acceso a todas las publicaciones relacionadas con su cuenta, pudiendo promover eventos, experiencias, comidas y alojamientos, además de administrar las modificaciones en cada una de sus publicaciones; el tercer módulo es el del turista, el cual cuenta con una gama de opciones para elaborar un itinerario acorde con sus necesidades, brindándole seguridad a la hora de realizar un viaje o actividad, tomando mejores decisiones.

1.2.2. Limitaciones

Existieron limitaciones importantes para la ejecución y el desarrollo del software. La adaptación de un grupo de 7 integrantes para desempeñar los diferentes roles en la aplicación del marco metodológico que describe un conjunto estructurado de actividades y procesos de manera cíclica, iterativa y controlada, para desarrollar el producto de software de una complejidad considerable y obtener un conjunto de documentación precisa, lo que requirió de planificación y ajustes en las responsabilidades de los diferentes integrantes. La asignación de herramientas tecnológicas novedosas, que eran desconocidas por los desarrolladores, dificultó lograr en menor tiempo la curva de aprendizaje, ya que se tuvo que desarrollar habilidades, experiencias y conocimientos relacionados con el uso de lenguajes, manejadores de base de datos no

convencionales, herramientas de apoyo tanto a nivel de liderazgo de proyecto, análisis, diseño, programación, pruebas e implementación.

El sistema se desarrolló de tal manera que se adapta fácilmente a la zona geográfica del Estado Sucre, aunque pudiera incorporarse a cualquier región de Venezuela y con algunas consideraciones legales se pudiera incorporar al mundo, de esta manera el mercado será más amplio y ventajoso para los diferentes entes interesados.

También es importante considerar las limitaciones del sistema desarrollado, ya que el mismo es limitado con respecto a la ejecución de los pagos en los itinerarios reservados por los turistas.

1.3. OBJETIVOS

1.3.1. Objetivo General

Desarrollar un Sistema de Información bajo ambiente Web para el incentivo, promoción y apoyo al sector turismo.

1.3.2. Objetivos Específicos

Determinar los elementos que definen los componentes de la aplicación.

Definir los requerimientos funcionales y no funcionales que contemplará el sistema.

Diseñar la estructura arquitectónica y detallada que integra el sistema.

Generar la codificación de la estructura diseñada.

Ensamblar los componentes que estarán definidos en la estructura funcional.

Probar el sistema para asegurar el cumplimiento de los requerimientos funcionales, no funcionales y la estandarización propuesta.

Clasificar los productos del desarrollo del software en productos técnicos, de gestión y de soporte.

Identificar los roles que se requieren para desarrollar la aplicación, estableciendo sus responsabilidades y la manera en que estos roles se organizan en el equipo de trabajo.

Describir los procesos técnicos, de gestión y de soporte que se requieren para desarrollar una aplicación.

Describir las actividades y flujos de trabajo de cada proceso.

CAPÍTULO II. MARCO DE REFERENCIA

2.1. MARCO TEÓRICO

2.1.1. Antecedentes de la investigación

Los sistemas web permiten a los usuarios el acceso a la información localizadas a grandes distancias cubriendo diferentes propósitos y situaciones, ofrecen un complejo arreglo de contenido y funcionalidad a una amplia población de usuarios finales, estos se integran cada vez más a las estrategias de los negocios para pequeñas y grandes empresas, satisfaciendo las necesidades de una variada comunidad de clientes, por este motivo las organizaciones educativas, comerciales, industriales, empresariales y de servicios hacen uso de estas aplicaciones, ya que a través de sus sitios Web muestran al mundo información institucional (Pressman, 2005).

Hurtado y Deam (2013) en su investigación plantean la aplicación de un Sistema de Información Gerencial para la Gestión Documental; tiene como objetivo gestionar adecuadamente los documentos generados por la ejecución de actividades, optimizar las búsquedas y el espacio físico empleado para el almacenamiento, cabe señalar que la finalidad de este sistema, como el de todo sistema gerencial, es brindar informes detallados que ayuden a la toma de decisiones, aportando seguridad de la información y gestión de la calidad para garantizar la integridad, disponibilidad y confiabilidad de la información. Se tomó como referencia esta investigación por el estudio que se realiza sobre los sistemas de información gerencial y la aplicación que se le puede dar en diferentes ámbitos.

Gonzales (2014) en la investigación que realizó para su tesis doctoral en la Universitat Ramon Llull, se dedicó a estudiar Impacto de la *Data Warehouse* e Inteligencia de Negocios en el Desempeño de las Empresas: Investigación Empírica en Perú, como País en Vías de Desarrollo, en esta se buscó estimar el

impacto de la *Data Warehouse* en el crecimiento y desarrollo en empresas de distintas índoles. Se hace referencia a esta investigación por las diferentes herramientas utilizadas por el autor para definir las variantes de las empresas al utilizar inteligencia de negocios.

La investigación de De Gregorio (2014) se basa en el estudio del impacto que producen las tecnologías de la información y la comunicación (TIC) en el sector turismo, se toma en cuenta esta investigación ya que es evidente que las TIC llegaron a mejorar un sinfín de ámbitos a nivel global y el turismo no escapa de esto. El turismo como negocio debe estar de la mano de cualquier innovación tecnológica que le ayude a su crecimiento, ya sea para obtener información y mejorar sus recursos o para promoción de sus características mediante medios que lleguen a un mercado más amplio.

La investigación de Valle (2009) plantea un plan de negocios para facilitar los servicios turísticos en Nicaragua, en el estudio determina diferentes factores que influyen en la disminución o aumento del turismo en la región, identifica los posibles problemas, clasifica a los turistas de acuerdo a su ubicación geográfica y determina planes para mejorar el servicio en Nicaragua. Se hace referencia a este trabajo por las diferentes áreas de abordar el turismo y establecer planes para su crecimiento, como lo es el marketing, elemento fundamental para que un negocio triunfe y que se puede complementar con el uso de la tecnología.

En el Núcleo de Sucre de la Universidad de Oriente, se desarrolló un sistema de información web para el control de los procesos que realiza la Comisión de Trabajos de Grado de la Licenciatura en Informática, el sistema está encargado de gestionar los procesos de la comisión ofreciendo funcionalidades que anteriormente eran llevadas de manera manual; sin embargo esta investigación fue utilizada como referencia por ser desarrollada bajo la metodología Watch, la cual posee diferentes variantes según el desarrollo del sistema y el tiempo que se plantea implementarlo. En la investigación de Gerardino se utilizó la

metodología Blue Watch, tomando de ella datos característicos para el desarrollo del proyecto (Gerardino, F. 2013).

Airbnb es una empresa y una plataforma de software dedicada a la oferta de alojamientos a particulares y turísticos. El nombre es un acrónimo de *airbed and breakfast* (colchón inflable y desayuno). Airbnb tiene una oferta de unas 2.000.000 de propiedades en 192 países y 33.000 ciudades. Desde su creación en noviembre de 2008 hasta junio de 2012 se realizaron 10 millones de reservas. Airbnb fue fundada en noviembre de 2008 por Brian Chesky, Joe Gebbia y Nathan Blecharczyk en San Francisco, California. La financiación original fue obtenida de la incubadora de empresas Y Combinator. Más tarde Greylock Partners, Sequoia Capital y Ashton Kutcher también invirtieron en la empresa. Airbnb abrió 6 oficinas internacionales a principio de 2012 en las siguientes ciudades: Paris, Milán, Barcelona, Copenhague, Moscú, y San Paulo (Airbnb, 2017).

En la región de Cerdeña, España, se realizó un análisis del impacto del turismo en el sistema económico de la región, a través de la estimación de un modelo CGE regional, que tuvo por objetivo el facilitar un cuadro de síntesis del sistema económico de la región de Cerdeña y del importante papel que en ella reviste la actividad turística. Se introdujeron tres escenarios fiscales que les permitió verificar y valorar, por un lado, los efectos que las maniobras de política económica producen sobre la actividad turística y sobre los principales agregados macro-económicos y por otro, los vínculos existentes entre el turismo y los principales sectores del sistema económico regional (Tiziana, Ferrari, y Secondi, 2010).

Se consideraron estudios realizados para la gestión del disfrute de la naturaleza en el sector turismo considerando las áreas naturales protegidas con especial apego a las normas de conservación de los recursos naturales y culturales. En la implementación de estas acciones resultó un producto que debe adecuarse a

expectativas económicas, como la generación de empleos directos e indirectos y los ingresos relacionados a ellos, además de los aspectos socioculturales de participación comunitaria, de las mujeres y hombres, y fueron fundamentales los aspectos ambientales en la puesta en valor de los recursos del medio rural (Díaz y Sastre, 2011).

2.1.1. Bases teóricas

Un sistema de información es un conjunto de elementos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio. Es una disposición de personas, actividades, datos, redes y tecnología, integrados entre sí con el propósito de apoyar y mejorar las operaciones cotidianas de una empresa, así como satisfacer las necesidades de información para la resolución de problemas y la toma de decisiones por parte de los directivos de una empresa (Laudon, K. 2004).

Figura 1. Sistemas de Información Gerenciales.
Fuente: Whitten *et al.,* 2005.

La ingeniería de software es la aplicación de un enfoque sistemático,

disciplinado y cuantificable al desarrollo, operación y mantenimiento del software; es decir la aplicación de la ingeniería al software (Whitten, Bentley y Barlow, 2005).

La ingeniería web es aquella que aplica sólidos principios científicos, de ingeniería y de administración, y enfoques disciplinados y sistemáticos para el desarrollo, despliegue y mantenimiento exitosos de sistemas y aplicaciones basados en Web de alta calidad (Pressman, R. 2005).

Los sistemas y aplicaciones basados en Web residen en una red y deben satisfacer las necesidades de una variada comunidad de clientes ofreciendo un complejo arreglo de contenido y funcionalidad. Engloban un espectro amplio de aplicaciones. En su forma más simples, estos sistemas son apenas un poco más que un conjunto de archivos de hipertexto ligados que presenta información mediante texto y algunas gráficas, logrando evolucionar hacia ambientes computacionales sofisticados que no sólo proporcionan características, funciones de cómputo y contenidos independientes al usuario final, sino que están integradas con bases de datos y aplicaciones de negocios (Pressman, R. 2005).

Los sistemas de procesamiento de transacciones son sistemas de información computarizada creados para procesar grandes cantidades de datos relacionadas con transacciones rutinarias de negocios, como las reservaciones y pagos (Cohen, D. 2000).

Sistemas de información administrativos o gerenciales son sistemas de información cuyo propósito es contribuir a la correcta interacción entre los usuarios y las computadoras. Debido a que requieren que los usuarios, el software (los programas de cómputo) y el hardware (las computadoras, impresoras, entre otros), funcionen de manera coordinada, los sistemas de información gerencial dan apoyo a un espectro de tareas organizacionales mucho más amplio que los sistemas de proceso de transacciones, como el

análisis y la toma de decisiones (Laudon, K. 2004).

La programación orientada a objetos (*Object-Oriented Programming*, OOP) es la base de unos de los cambios más importantes en la programación de computadoras. La OOP representa una nueva forma de organizar y construir los programas. La programación orientada a objeto (OO) parte de que los programas manipulan objetos que contienen valores de datos y la lógica de programa que opera sobre ellos (Oviedo, E. 2015).

UML (*Unified Modeling Language*) es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software. Cubre aspectos conceptuales como procesos de negocio y funciones del sistema, así como cosas concretas tales como las clases, los esquemas de bases de datos y los componentes de software reutilizables (Booch, Rumbaugh y Jacobson, 2006).

El UML está compuesto por diversos elementos gráficos que se forman para conformar diagramas. Debido a que el UML es un lenguaje, cuenta con reglas para combinar tales elementos lo cual permite la creación de dichos diagramas que lo conforman (Schmuller, J. 2000).

El diagrama de casos de uso muestra las relaciones existentes entre actores y casos de uso dentro de un sistema. Un caso de uso describe una interacción con los actores como secuencia de mensajes entre el sistema y uno o más actores, este es representado gráficamente por una elipse. El término actor incluye a los seres humanos, así como a otros sistemas informáticos y procesos. Este diagrama fue de suma importancia para modelar el comportamiento del sistema desarrollado (Rumbaugh, Booch y Jacobson, 2007).

Un caso de uso es una herramienta útil para los desarrolladores de sistemas, que les permite obtener los requerimientos desde el punto de vista del usuario.

Es una colección de situaciones respecto al uso de un sistema. Cada escenario describe una secuencia de eventos. Cada secuencia se inicia por una persona, otro sistema, una parte del hardware o por el paso del tiempo. A las entidades que inician secuencias se les conoce como actores. El resultado de la secuencia debe ser algo utilizable ya sea por el actor que la inició, o por otro actor. Gráficamente, los casos de uso se representan mediante elipses (Schmuller, J. 2000).

El caso de uso es un documento narrativo que describe la secuencia de eventos de un actor (agente externo) que utiliza para completar un proceso. Los casos de uso son historias o casos de utilización de un sistema; no son exactamente los requerimientos ni las especificaciones funcionales, sino que ejemplifican e incluyen tácticamente los requerimientos en las historias que narran. Un caso de uso es, en esencia, una interacción típica entre un usuario y un sistema de cómputo (Larman, C. 2003).

El actor es una entidad externa del sistema que de alguna manera participa en la historia del caso de uso. Por lo regular estimula el sistema con eventos de entrada o recibe algo de él. Los actores están representados por el papel que desempeñan en el caso (Rumbaugh *et al*., 2007).

La simbología utilizada para la elaboración de un caso de uso se presenta en la figura 2.

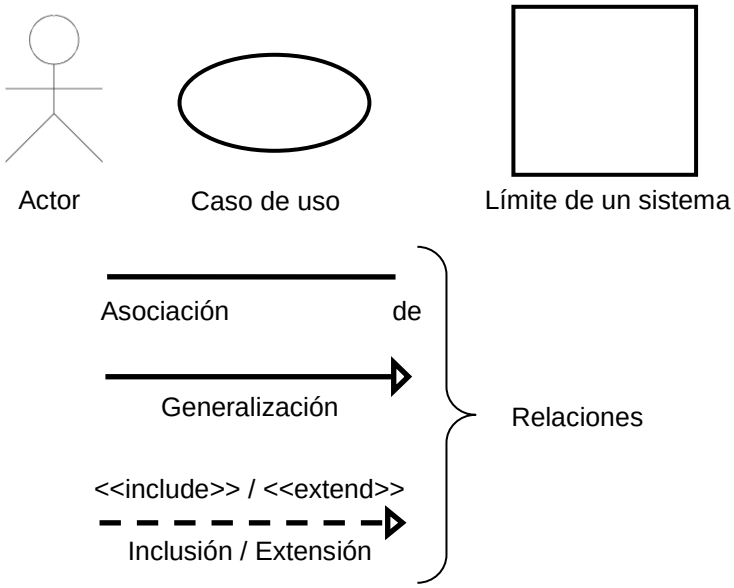


Figura 2. Simbología de Casos de Uso.
Fuente: Rumbaugh *et al*., 2007.

UML define cuatro tipos de relación en los diagramas de casos de usos: comunicación, inclusión, extensión y herencia. En la inclusión, una instancia del caso de uso origen incluye también el comportamiento descrito por el caso de uso destino. Las relaciones de inclusión se identifican como <<*include*>>. En la extensión, el caso de uso origen extiende el comportamiento del caso de uso destino. Se identifican como <<*extend*>>. En la herencia, el caso de uso origen hereda la especificación del caso de uso destino y posiblemente la modifica y/o amplía (Larman, C. 2003).

Las clases son declaraciones de objetos, también se podrían definir como abstracciones de objetos. Esto quiere decir que la definición de un objeto es la clase. Cuando programamos un objeto y definimos sus características y funcionalidades en realidad lo que estamos haciendo es programar una clase (Sommerville, I. 2005).

Las propiedades o atributos de las clases son las características de los objetos. Cuando definimos una propiedad normalmente especificamos su nombre y su

tipo. Nos podemos hacer a la idea de que las propiedades son algo así como variables donde almacenamos datos relacionados con los objetos. Los Métodos en las clases son las funcionalidades asociadas a los objetos. Cuando estamos programando las clases las llamamos métodos. Los métodos son como funciones que están asociadas a un objeto (Larman, C. 2003).

Una clase es una descripción generalizada o categorización de un grupo de cosas que tienen atributos y acciones similares. Las clases encapsulan los datos y las abstracciones de procedimientos requeridos para describir el contenido y el comportamiento de alguna entidad del mundo real. Una clase define los atributos y los métodos de una serie de objetos. Todos los objetos de esta clase (instancias de esa clase) tienen el mismo comportamiento y el mismo conjunto de atributos (cada objeto tiene el suyo propio). En ocasiones se utiliza el término “tipo” en lugar de clase, pero recuerde que no son lo mismo, y que el término tipo tiene un significado más general (Booch *et al*., 2006).

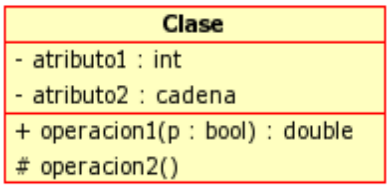


Figura 3. Representación gráfica de una clase en UML.
Fuente: Booch *et al*., 2006.

En UML, los atributos se muestran al menos con su nombre, y también pueden mostrar su tipo, valor inicial y otras propiedades. Los atributos también pueden ser mostrados visualmente: + Indica atributos *públicos,* # Indica atributos *protegidos,* - Indica atributos *privados.* Las operaciones (métodos) también se muestran al menos con su nombre, y pueden mostrar sus parámetros y valores de retorno. Las operaciones, se muestran visualmente: + Indica operaciones *públicas,* # Indica operaciones *protegidas,* - Indica operaciones *privadas* (Larman, C. 2003).

La generalización o herencia es uno de los conceptos fundamentales de la programación orientada a objetos, en la que una clase “recoge” todos los atributos y operaciones de la clase de la que es heredera, y puede alterar/modificar algunos de ellos, así como añadir más atributos y operaciones propias (Schmuller, J. 2000).

Una asociación representa una relación entre clases, y aporta la semántica común y la estructura de muchos tipos de “conexiones” entre objetos. Las asociaciones son los mecanismos que permite a los objetos comunicarse entre sí. Describe la conexión entre diferentes clases (la conexión entre los objetos reales se denomina conexión de objetos o *enlace*). Las asociaciones pueden tener un papel que especifica el propósito de la asociación y pueden ser unidireccionales o bidireccionales (indicando si los dos objetos participantes en la relación pueden intercambiar mensajes entre sí, o es únicamente uno de ellos el que recibe información del otro). Cada extremo de la asociación también tiene un valor de multiplicidad, que indica cuántos objetos de ese lado de la asociación están relacionados con un objeto del extremo contrario (Schmuller, J. 2000).

Las composiciones son asociaciones que representan acumulaciones muy fuertes. Esto significa que las composiciones también forman relaciones completas, pero dichas relaciones son tan fuertes que las partes no pueden existir por sí mismas. Únicamente existen como parte del conjunto, y si este es destruido las partes también lo son (Sommerville, J. 2005).

El diagrama de clases es una presentación gráfica de la vista estática de un sistema que muestra una colección de elementos estáticos del modelo, como las clases, que son una descripción generalizada o categorización de un grupo de cosas que tienen atributos y acciones similares, encapsulan los datos y las abstracciones de procedimientos requeridos para describir el contenido y el comportamiento de alguna entidad del mundo real, tipos y sus contenidos y

relaciones (Booch *et al*., 2006).

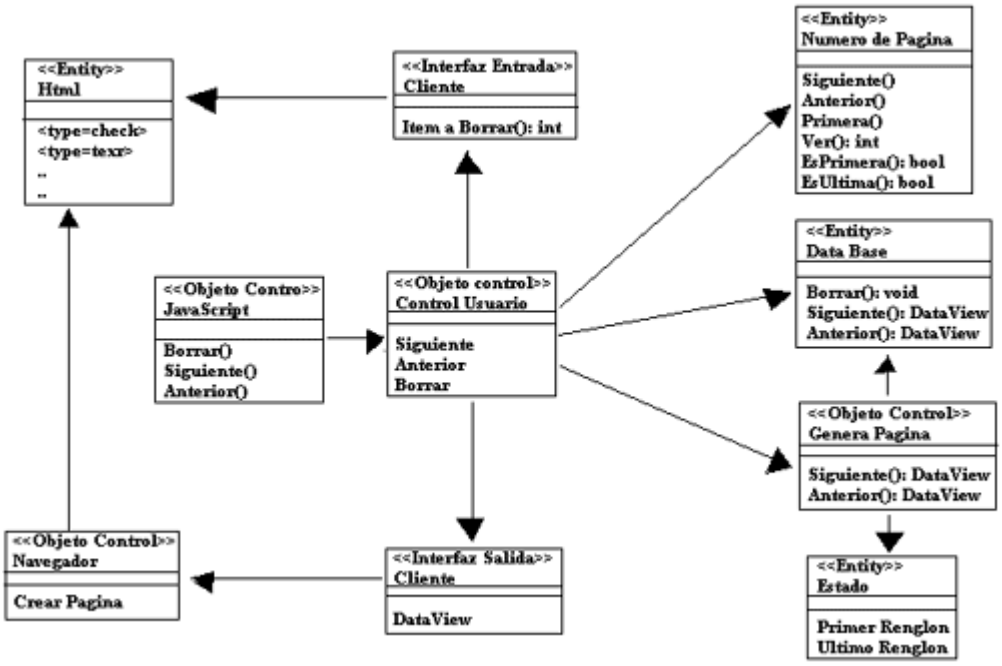


Figura 4. Ejemplo de Diagrama de Clases.
Fuente: Booch *et al*., 2006.

Los objetos son ejemplares de una clase cualquiera. Cuando creamos un ejemplar tenemos que especificar la clase a partir de la cual se creará. Esta acción de crear un objeto a partir de una clase se llama instanciar (que viene de una mala traducción de la palabra *instace* que en inglés significa ejemplar). Para crear un objeto se tiene que escribir una instrucción especial que puede ser distinta dependiendo el lenguaje de programación que se emplee, pero será algo parecido a esto: *Persona = new Persona()*. Con la palabra *new* especificamos que se tiene que crear una instancia de la clase que sigue a continuación (Schmuller, J. 2000).

Un diagrama de secuencia muestra las interacciones entre objetos, unidad de códigos compuestos de variables y métodos relacionados, organizados en una secuencia temporal. Este diagrama representa una interacción como un gráfico bidimensional. La dimensión vertical es el eje de tiempo que avanza hacia abajo

de la página. La dimensión horizontal muestra los roles de clasificador que representan objetos individuales en la colaboración (Booch *et al*., 2006).

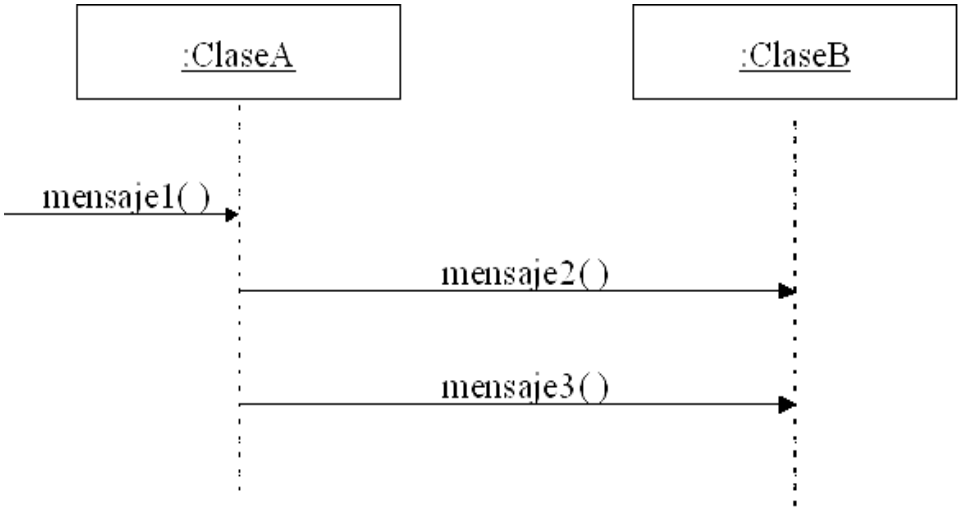


Figura 5. Notación de Diagrama de Secuencias.
Fuente: Booch *et al*., 2006.

El diagrama de secuencias sirve para modelar la interacción entre objetos en un sistema, contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes pasados entre los objetos. Un diagrama de secuencia muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como vectores horizontales. Los mensajes se dibujan cronológicamente desde la parte superior del diagrama a la parte inferior; la distribución horizontal de los objetos es arbitraria (Booch *et al*., 2006).

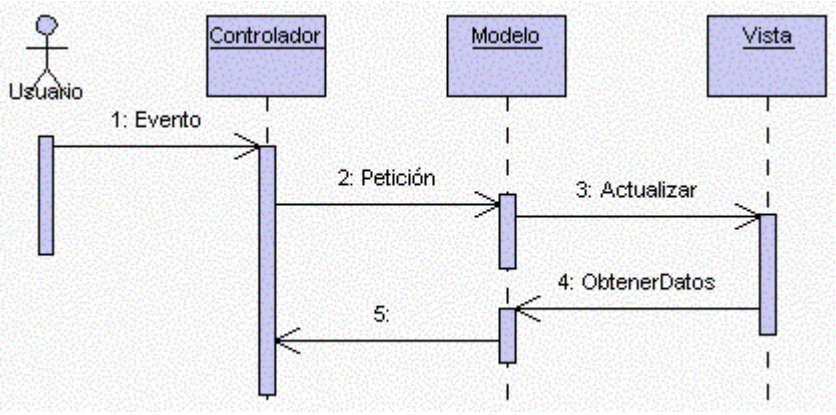


Figura 6. Notación MVC del Diagrama de Secuencias.
Fuente: Booch *et al*., 2006.

El diagrama de actividades se usa para mostrar cómo se construyen los diferentes flujos de trabajo o los procesos dentro de un sistema, cómo se inician, los variados caminos alternativos que se pueden tomar desde el inicio hasta el fin. También pueden ilustrar dónde puede ocurrir procesamiento en paralelo durante la ejecución de algunas actividades (Booch *et al*., 2006).

El diagrama de despliegue pertenece a los diagramas de UML que permiten modelar los aspectos físicos de un sistema. Este diagrama muestra la configuración de los nodos que participan en la ejecución y de los componentes que residen en ellos. Se utilizan para modelar la vista de despliegue estática de un sistema, esto implica poder modelar la topología del hardware y software sobre el que se ejecuta el sistema (Rumbaugh *et al*., 2007).

Un diagrama de componentes muestra la organización y las dependencias entre un conjunto de componentes. Se utilizan para modelar la vista de implementación estática de un sistema. Los diagramas de componentes se relacionan con los diagramas de clases en que un componente normalmente se corresponde con una o más clases, interfaces o colaboraciones (Jacobson, Booch y Rumbaugh, 2000).

A continuación, se presentan un conjunto de conceptos básicos los cuales

fueron necesarios para el desarrollo de este sistema de información Web.

Las aplicaciones Web, también llamadas soluciones Intranet, son sistemas capaces de encargarse del acceso a la información de una organización y sus procesos, así como de la distribución de esta información y de las necesidades de comunicación, para crear, editar, manipular o cambiar los datos de la Intranet (Powell, 2001).

La arquitectura cliente-servidor consiste básicamente en un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta. Aunque esta idea se puede aplicar a programas que se ejecutan sobre una sola computadora es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras. En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema (*Cliente-servidor,* s.f.)

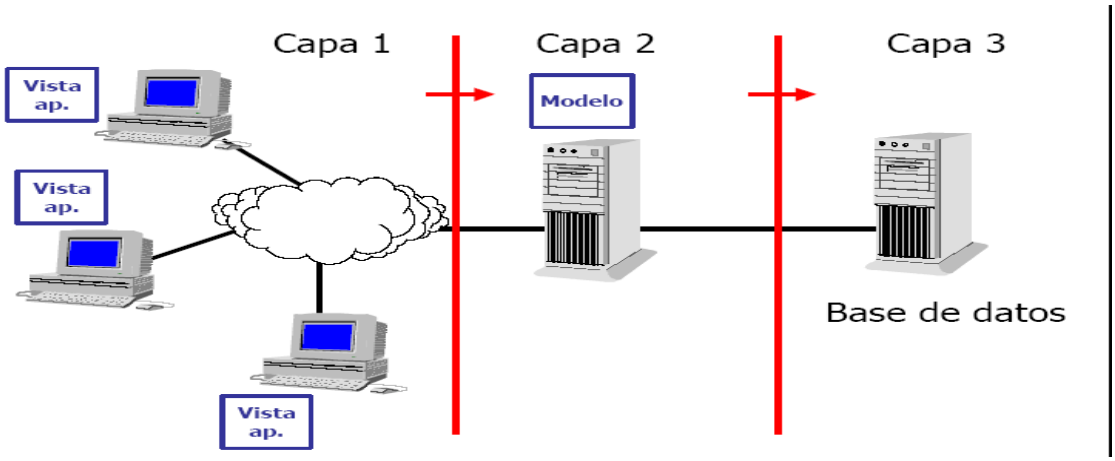


Figura 7. Arquitectura Cliente Servidor de tres (03) capas.
Fuente: Sánchez, 2004.

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas *Web* dinámicas. Técnicamente, *JavaScripts* es un lenguaje de

programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con *JavaScripts* se puede probar directamente en cualquier navegador sin necesidad de procesos intermedios. JavaScript es el pegamento de Internet. Gracias a este lenguaje son posibles aplicaciones como Google Docs, SkyDrive, GMail, Facebook o Twitter. En la actualidad todos los ordenadores, tablets y móviles del mundo incorporan al menos un intérprete de JavaScript. Además, su influencia se ha incrementado para alcanzar todo tipo de ámbitos: servidores, aplicaciones de escritorio, sistemas operativos, entre otros. Detrás de su aparente simplicidad se esconde un lenguaje de programación multi-paradigma y tremendamente versátil (Alarcón, J. 2012).

Ajax, acrónimo de *Asynchronous JavaScript And XML* (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (*Rich Internet Applications*). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones (Alarcón, J. 2012).

Otra concepción de las aplicaciones Web es la vista, que en su forma más simple las aplicaciones Web son apenas un poco más que un conjunto de archivos de hipertexto ligados que presentan información mediante textos y algunas gráficas. Sin embargo, a medida que el comercio electrónico y las aplicaciones B2B (en inglés *business to business*) adquieren mayor importancia, las aplicaciones Web evolucionan hacia ambientes computacionales sofisticados que no sólo proporcionan características, funciones de cómputo y contenidos independientes al usuario final, sino que están integrados con bases de datos corporativas y aplicaciones de negocios (Pressman, 2005).

Las aplicaciones Web son accedidas a través de una computadora con navegador Web ya que éste permite mostrar archivos (páginas y otros) y seguir enlaces según la elección del usuario, por lo cual una aplicación Web debe estar alojada en un servidor con acceso a Internet.

Un servidor es un programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten usando el protocolo HTTP o el protocolo HTTPS. Hypertext Transfer Protocol Secure (en español: *Protocolo seguro de transferencia de hipertexto*), más conocido por sus siglas HTTPS, es un protocolo de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de Hipertexto, es decir, es la versión segura de HTTP. El sistema HTTPS utiliza un cifrado basado en SSL/TLS para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información sensible que el protocolo HTTP. De este modo se consigue que la información sensible (usuario y claves de paso normalmente) no pueda ser usada por un atacante que haya conseguido interceptar la transferencia de datos de la conexión, ya que lo único que obtendrá será un flujo de datos cifrados que le resultará imposible de descifrar (Protocolo seguro de transferencia de hipertexto, s.f.).

Para la construcción de un sistema de información bajo ambiente Web se deben estructurar una serie de elementos tales como la arquitectura, este caso la tres (3) capas, y patrones de diseño como el modelo vista controlador (MVC). La arquitectura de software, tiene que ver con el diseño y la implementación de estructuras de software de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema, así como requerimientos no funcionales, como la confiabilidad, escalabilidad, portabilidad, y disponibilidad (Kruchten, 1995). Ver figura 8.

Dentro de las arquitecturas más comunes se tienen la monolítica, donde el software se estructura en grupos funcionales muy acoplados, el modelo 3 capas o niveles es una especialización de la arquitectura cliente – servidor donde la carga se divide en tres partes con un reparto claro de funciones: una capa para la presentación representada por la interfaz de usuario, otra para la aplicación de los métodos, colaboraciones y construcciones donde se encuentra el modelado el negocio y otra para el almacenamiento de las clases de persistencia, y la arquitectura utilizada en este trabajo la cual es el modelo 4 capas, en ésta se divide el modelado de negocio de la expuesta anteriormente en dos: la capa web donde se encuentra el servidor web y la capa de transacciones donde se mantiene la lógica de negocio.

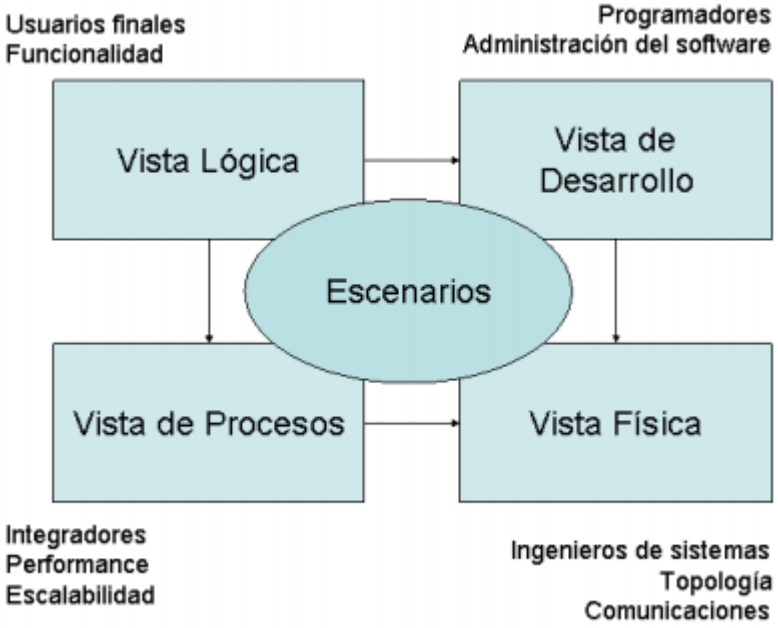


Figura 8. Modelo de 4 + 1 Vistas de Kruchten.
Fuente: Kruchten, 1995.

Las herramientas CASE (*Computer Aided Software Engineering*, Ingeniería de Software Asistida por Ordenador) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo

el coste de las mismas en términos de tiempo y de dinero. Estas herramientas nos pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software, en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras (Landeros, R. 2009).

Enterprise Architect es una herramienta CASE avanzada que combina el poder de la última especificación UML 2.1 con alto rendimiento, interfaz intuitiva, para traer modelado avanzado al escritorio, y para el equipo completo de desarrollo e implementación. Con un gran conjunto de características, equipa a su equipo entero, incluyendo analistas, evaluadores, administradores de proyectos, personal del control de calidad, equipo de desarrollo y más. Es una herramienta de diseño y análisis UML, cubriendo el desarrollo de software desde el paso de los requerimientos a través de las etapas del análisis, modelos de diseño, pruebas y mantenimiento, es multiusuario, basada en Windows, diseñada para ayudar a construir software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad. El Lenguaje Unificado de Modelado provee beneficios significativos para ayudar a construir modelos de sistemas de software rigurosos y donde es posible mantener la trazabilidad de manera consistente. Enterprise Architect soporta este proceso en un ambiente fácil de usar, rápido y flexible, provee trazabilidad completa desde el análisis de requerimientos hasta los artefactos de análisis y diseño, a través de la implementación y el despliegue. Combinados con la ubicación de recursos y tareas incorporados, los equipos de Administradores de Proyectos y Calidad están equipados con la información que ellos necesitan para ayudarles a entregar proyectos en tiempo (Sparx Systems, 2017).

La base de datos representa una extensa y organizada recopilación de información a la cual se tiene acceso a través de software y que persiste a través del tiempo. Las bases de datos generalmente funcionan en

computadoras que se dedican a forma exclusiva a este campo. Por las prestaciones requeridas, generalmente funcionan en computadoras multiprocesador con abundante memoria. Para el almacenamiento de los datos puede contar con sistemas de disco propio o almacenamiento de conexión directa (DAS), puede conectarse a una red de almacenamiento (SAN) o conectarse a un sistema de almacenamiento en red (NAS) (Sistema de gestión de bases de datos, s.f.).

Un sistema gestor de base de datos (SGBD) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos, además de proporcionar herramientas para añadir, borrar, modificar y analizar los datos. Los usuarios pueden acceder a la información usando herramientas específicas de consulta y de generación de informes, o bien mediante aplicaciones al efecto. Estos sistemas también proporcionan métodos para mantener la integridad de los datos, para administrar el acceso de usuarios a los datos y para recuperar la información si el sistema se corrompe. Permiten presentar la información de la base de datos en variados formatos. La mayoría incluyen un generador de informes. También pueden incluir un módulo gráfico que permita presentar la información con gráficos y tablas. Generalmente se accede a los datos mediante lenguajes de consulta, lenguajes de alto nivel que simplifican la tarea de construir las aplicaciones. También simplifican las consultas y la presentación de la información. Un SGBD permite controlar el acceso a los datos, asegurar su integridad, gestionar el acceso concurrente a ellos, recuperar los datos tras un fallo del sistema y hacer copias de seguridad. Las bases de datos y los sistemas para su gestión son esenciales para cualquier área de negocio, y deben ser gestionados con esmero (Sistema de gestión de bases de datos, s.f.).

MongoDB (de la palabra en inglés “hu**mongo**us” que significa enorme) es un sistema de base de datos NoSQL orientado a documentos, desarrollado bajo el concepto de código abierto. MongoDB es un sistema de base de

datos NoSQL orientado a documentos, desarrollado bajo el concepto de código abierto. En lugar de guardar los datos en tablas como se hace en las bases de datos relacionales, MongoDB guarda estructuras de datos en documentos similares a JSON con un esquema dinámico haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida (MongoDB, s.f.).

Para el desarrollo del sistema utilizamos la base de datos MongoDB, la cual es una base de datos orientada a documentos. Esto quiere decir que en lugar de guardar los datos en registros, guarda los datos en documentos. Estos documentos son almacenados en BSON, que es una representación binaria de JSON (Genbetadev, 2014).

Se utiliza Mongoose como una herramienta de modelado de objetos MongoDB diseñada para trabajar en un entorno asíncrono (Npmjs, 2010.). Mongoose proporciona una solución directa basada en esquemas para modelar los datos de su aplicación. Incluye casting de tipo incorporado, validación, creación de consultas, lógica de negocios y más, listo para usar (Mongoosejs, 2011).

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o front end como Cogito o StGIT. Sin embargo, Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena (Git, s.f.).

Se utiliza GitHub como una plataforma de desarrollo colaborativo de software para alojar proyectos utilizando el sistema de control de versiones Git, brindando herramientas muy útiles para el trabajo en equipo, dentro de un proyecto (Conociendo Github, 2012).

Bootstrap es el marco de HTML, CSS y JS más popular para desarrollar primeros proyectos receptivos y móviles en la web. Bootstrap hace que el desarrollo web front-end sea más rápido y más fácil. Está hecho para personas de todos los niveles de habilidad, dispositivos de todas las formas y proyectos de todos los tamaños, ya que escala de forma fácil y eficiente sus sitios web y aplicaciones con una única base de código, desde teléfonos a tabletas y escritorios con consultas de medios de CSS (Getbootstrap, s.f.).

jQuery es una biblioteca multiplataforma de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio (jQuery, s.f.).

Gentellela Admin es una plantilla de administrador de Bootstrap de uso gratuito. Esta plantilla utiliza los estilos predeterminados de Bootstrap 3 junto con una variedad de potentes complementos y herramientas de jQuery para crear un potente marco para crear paneles de administración o paneles de fondo. Utiliza varias bibliotecas para gráficos, calendarios, validación, interfaz de estilo de asistente, menú de navegación fuera de lienzo, formularios de texto, rango de fechas, área de carga, autocompletar formulario, deslizador de rango, barras de progreso, notificaciones y mucho más (Graygrids, 2015).

Sublime Text es un editor de texto y editor de código fuente está escrito en C++ y Python para los plugins. Desarrollado originalmente como una extensión de Vim, con el tiempo fue creando una identidad propia, por esto aún conserva un modo de edición tipo vi llamado *Vintage mode* (Sublime Text, s.f.).

Node.js es un tiempo de ejecución de JavaScript creado en el motor de

JavaScript V8 de Chrome. Node.js utiliza un modelo de E/S no bloqueado y controlado por eventos que lo hace liviano y eficiente. El ecosistema del paquete Node.js, npm , es el ecosistema más grande de bibliotecas de código abierto en el mundo (Nodejs, s.f.).

Node.js es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) basado en el lenguaje de programación ECMAScript, asíncrono, con I/O de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables. Node.js es similar en su propósito a Twisted o Tornado de Python, Perl Object Environment de Perl, libevent o libev de C, EventMachine de Ruby, vibe.d de D y Java EE de Java, existe Apache MINA, Netty, Akka, Vert x, Grizzly o Xsocket. Al contrario que la mayoría del código JavaScript, no se ejecuta en un navegador, sino en el servidor. Node.js implementa algunas especificaciones de CommonJS. Node.js incluye un entorno REPL para depuración interactiva (Node.js, s.f.).

2.1. METODOLOGÍA DEL ÁREA APLICADA

2.1.1. Metodología de la investigación

Se tomó como referencia para el desarrollo de este trabajo de grado el marco metodológico propuesto por Tamayo (2001).

Forma de investigación

La forma de investigación se enmarcó en el área de las investigaciones aplicadas, ya que confrontó principios y basamentos teóricos con la realidad, estudiando y aplicando la investigación a problemas concretos y reales, es decir, se estudiaron y aplicaron los conceptos y principios relativos a los sistemas de información Web, confrontándolos con las circunstancias y características propias en el desarrollo de un sistema de información bajo

ambiente web para el incentivo, promoción y apoyo al sector turismo.

Tipo de Investigación

El tipo de investigación fue descriptiva porque comprendió la descripción, registro, análisis e interpretación del estado de la naturaleza actual y la composición o proceso de los fenómenos que se realizan [22]. En esta investigación se realizaron estudios con el fin de detallar y especificar las actividades realizadas por los actores interesados, como lo son el administrador, los prestadores de servicios y los turistas, para llevar a cabo a feliz término los objetivos planteados.

Diseño de la Investigación

En función de los tipos de datos recolectados para llevar a cabo la investigación se considera el diseño de la investigación de campo, porque los datos se obtuvieron directamente de la realidad (Tamayo, 2001), es decir, tal y como se llevan a cabo los procesos relacionados con el sector turismo, para su incentivo, promoción y apoyo.

Técnicas de recolección de datos

Las técnicas en las cuales se apoyó la investigación fueron revisión de material bibliográfico (manuales, formularios, plantillas, componentes, código reutilizable, fotos, páginas web y archivos digitales), con los que se pudo definir, resolver, complementar y/o argumentar las bases y el desarrollo de la aplicación Web; la observación directa, que permitió participar en las diferentes aplicaciones web de servicios turísticos que se ofrecen en la red, modelando diferentes procesos de otras aplicaciones y determinando los propios, con lo que se pudo comprender el funcionamiento que se deseaba y obtener información confiable de cómo se efectúan los procesos requeridos; la aplicación de entrevistas no estructuradas a los clientes que propusieron el proyecto, lo que permitió reunir información relevante para el desarrollo de la aplicación (Tamayo, 2001).

2.1.2. Metodología del área aplicada

Para la realización del desarrollo de un sistema de información bajo ambiente web para el incentivo, promoción y apoyo al sector turismo se utilizó como proceso de desarrollo de software el método *WATCH*, el cual es un marco metodológico que describe los 22 procesos técnicos, gerenciales y de soporte que deben emplear los equipos de trabajo que tendrán a su cargo el desarrollo de aplicaciones de software empresarial (Montilva *et al*., 2008).

El método *WATCH* está compuesto por tres modelos que describen los tres elementos claves de todo método: el producto que se quiere elaborar, los actores que lo elaborarán y el proceso que los actores deben seguir para elaborar el producto.

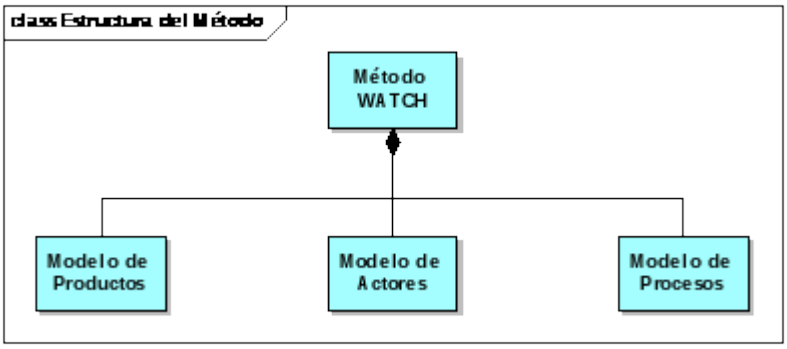


Figura 9. Componentes del Método Watch.
Fuente: Montilva *et al*., 2008.

Modelo de productos este modelo identifica y describe los tipos de productos que se deben generar durante el desarrollo de una aplicación empresarial. Estos tipos de productos se elaboran durante la ejecución de los procesos técnicos, de gestión o de soporte, que están descritos en el Modelo de Procesos del método.

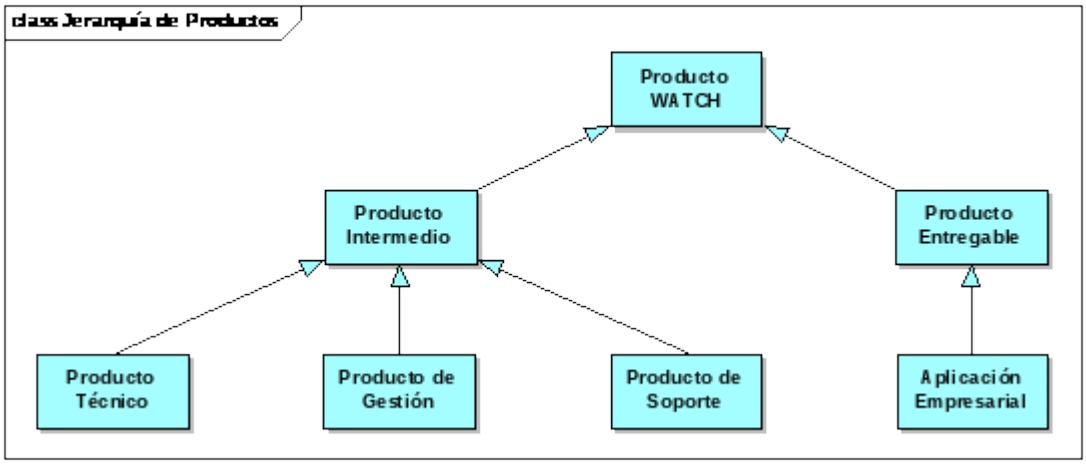


Figura 10. Jerarquía de productos del Método Watch.
Fuente: Montilva *et al*., 2008.

Los productos intermedios son todos aquellos documentos, modelos, listas, librerías de software, matrices, etc., que se elaboran durante la ejecución de los procesos técnicos, de soporte y de gestión y que son necesarios para desarrollar la aplicación. No son considerados productos finales o entregables, por cuanto no constituyen parte integrante de la aplicación. Los productos entregables o finales del proyecto son todos aquellos que conforman la aplicación empresarial propiamente dicha y que son entregados al cliente al final de un ciclo de desarrollo o de todo el proyecto. En este grupo se incluyen todas las versiones de la aplicación que se elaboran durante la vida del proyecto. Cada versión entregable está compuesta de programas, bases de datos y manuales.

Productos técnicos

Los productos técnicos son todos aquellos que se originan durante la ejecución de los procesos técnicos del desarrollo de la aplicación, dentro de los cuales se tienen: el modelado de negocios que tiene como objetivo asegurar que el equipo de desarrollo tenga un conocimiento adecuado del dominio de la aplicación, de manera tal que se facilite, en los procesos siguientes, definir apropiadamente los requisitos de la aplicación; el documento de requisitos es

un documento producido en el proceso de ingeniería de requisitos y tiene por objetivo identificar, describir, especificar y documentar cada uno de los requisitos funcionales y no funcionales que la aplicación empresarial debe satisfacer; el documento de diseño es un documento producido durante los procesos de diseño arquitectónico y diseño detallado y tiene por objetivo documentar los detalles del diseño de la arquitectura del sistema y de cada uno de los componentes que integran esta arquitectura; las especificaciones de pruebas son documentos que se elaboran durante la ejecución de los procesos de programación e integración, pruebas de la aplicación y entrega de la aplicación para realizar las pruebas de unidad, integración y sistemas que se requieren para verificar y validar dinámicamente la aplicación.

Productos de soporte

Los productos de soporte se originan durante la ejecución de los procesos de gestión de la configuración, gestión de riesgos y gestión de la calidad. Aquí se tienen: el plan de gestión de riesgo el cual es un documento de tipo gerencial que describe los objetivos plan, las actividades, recursos, responsabilidades, costos, tiempos que son necesarios para evaluar y responder a los riesgos del proyecto de manera organizada; el plan de gestión de la configuración es un documento de tipo gerencial que describe las actividades, recursos, tiempos y costos necesarios para controlar la configuración de una aplicación (el conjunto de productos que surgen durante su desarrollo); el plan de gestión de aseguramiento de la calidad es un documento gerencial, cuyo objetivo es definir un plan que permita conducir los procesos, actividades y tareas de aseguramiento de la calidad; el plan de gestión de verificación y validación es un documento que describe las actividades, recursos, tiempos, técnicas y procedimientos necesarios para verificar que cada uno de los productos intermedios y finales, del desarrollo de una aplicación empresarial, satisfacen los requisitos especificados en el documento de requisitos, y por otra parte, validar que la aplicación satisface las necesidades de información de sus

usuarios, es decir, llena las expectativas de los usuarios; el plan de gestión de pruebas es un documento que se deriva del plan de validación y verificación, tiene un carácter técnico-gerencial y describe, detalladamente, las actividades de verificación y validación dinámica (pruebas de software) que el grupo de pruebas debe realizar, con la finalidad de detectar los errores (faltas y fallas) en cada uno de los programas que haya sido elaborado por el grupo de programación e integración; el plan de gestión de auditorías es un documento en el que se establecen el cronograma de auditorías en base a los hitos especificados en el plan de proyecto, y establece las diferentes auditorías a realizar durante el ciclo de vida del proyecto; el informe de resultados es un documento en el que se describen los resultados obtenidos durante los procesos de verificación y validación, pruebas, auditorías y revisiones.

Productos de gestión

Los productos de gestión son elaborados durante la ejecución de los procesos de constitución, planificación, dirección, control y cierre del proyecto. Está constituido por los siguientes documentos: el enunciado del trabajo del proyecto es un documento de carácter preliminar que tiene por objetivo convencer a la alta gerencia de la empresa sobre la necesidad de desarrolla una nueva aplicación empresarial, indicando el porqué es necesaria la aplicación, que unidades organizacionales se verán beneficiadas y porque la empresa debe invertir en su desarrollo; el documento de inicio del proyecto es un documento de carácter gerencial y describe la importancia del proyecto, su justificación, sus objetivos, la relación de estos objetivos con los objetivos de negocio, los resultados esperados y la estimación preliminar de costos; el proceso de desarrollo es el resultado es la instanciación del método el cual no es más que una adaptación del modelo de procesos del método en la que se describe, con mayor precisión, los procesos específicos que se aplicaran al desarrollo de una aplicación particular; el plan integral de proyecto es un documento formal utilizado para gestionar la ejecución del proyecto y controlar su desarrollo, es el

documento de gestión más importante, pues, es usado para guiar los procesos de ejecución y control del proyecto; los informes de gestión o rendimiento del trabajo son utilizados para informar sobre el avance del proyecto o describir situaciones que puedan afectar el desarrollo normal del proyecto; los contratos son documentos legales que se establecen entre las empresas participantes en el proyecto, allí se define formalmente un acuerdo entre dos partes.

Modelo de actores

El modelo de actores tiene como objetivos: identificar los actores o interesados (*stakeholders*) que están involucrados en el desarrollo de aplicaciones empresariales; describir las modalidades de organización del equipo de trabajo que desarrollarán los diferentes componentes arquitectónicos de una aplicación empresarial; y definir los roles y responsabilidades de aquellos actores que integrarán el equipo de trabajo, tal como se muestra en la Figura 12.

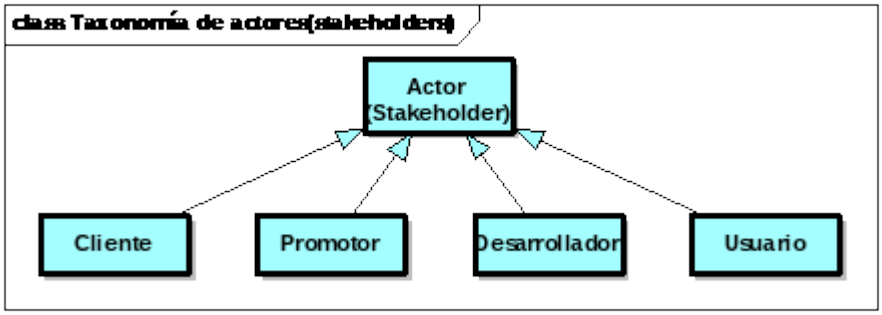


Figura 11. Clasificación de Actores.
Fuente: Montilva *et al*., 2008.

Cliente

El cliente es toda aquella persona, unidad organizativa o empresa que contrata o financia el proyecto de desarrollo de una aplicación empresarial. Su rol fundamental es proveer los recursos económicos que el equipo de desarrollo requiere para ejecutar el proyecto.

Promotor

Es toda aquella persona, unidad organizativa o empresa que tiene particular interés porque el proyecto se lleve a cabo, bien porque se beneficia directamente de los servicios que la aplicación le podrá proveer, o porque considera que la aplicación es necesaria para alcanzar objetivos de su empresa.

Desarrollador

Es aquella persona o grupo de personas que participan activamente en el desarrollo de la aplicación ejecutando procesos técnicos, de gestión y de soporte.

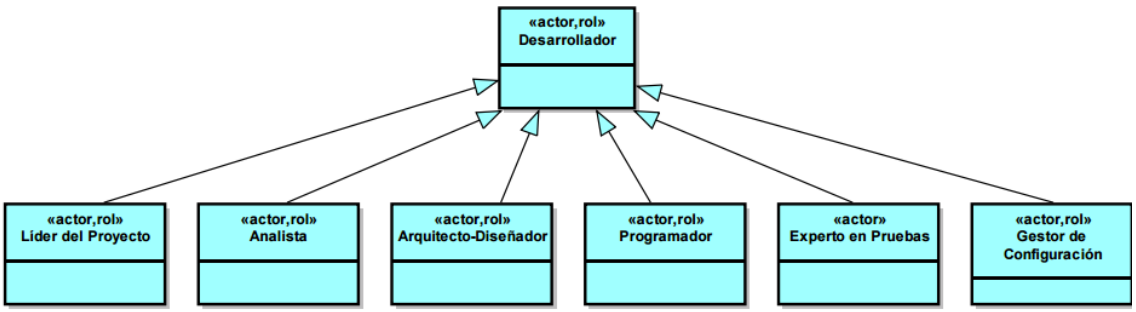


Figura 12. Roles de los desarrolladores requeridos por Blue Watch.
Fuente: Montilva y Barrios, 2014.

Usuario

Es aquella persona, grupo de personas, unidad u organización que hace uso de la aplicación empresarial para satisfacer necesidades de información y/o automatización de procesos.

Modelo de procesos

El objetivo de este modelo es describir los procesos técnicos, de gestión y de

soporte que los equipos de trabajo deben emplear para desarrollar una aplicación empresarial. Estos procesos se organizan en la forma de una cadena de valor. Ver Figura 13.

Dichos procesos se clasifican, como se muestra en la Figura 14, según su naturaleza con respecto al proceso de desarrollo de software, en tres grupos: procesos técnicos, procesos de gestión y procesos de soporte.

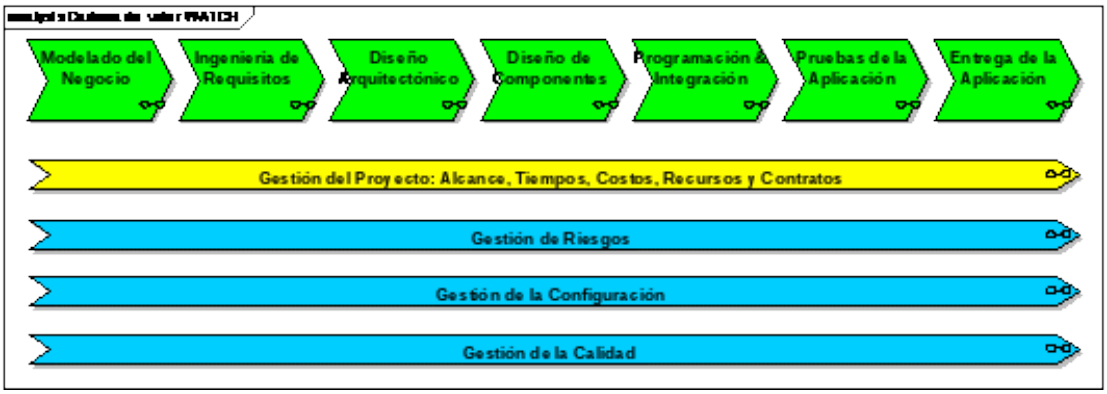


Figura 13. Cadena de Valor de los procesos del Método Watch.
Fuente: Montilva *et al*., 2008.

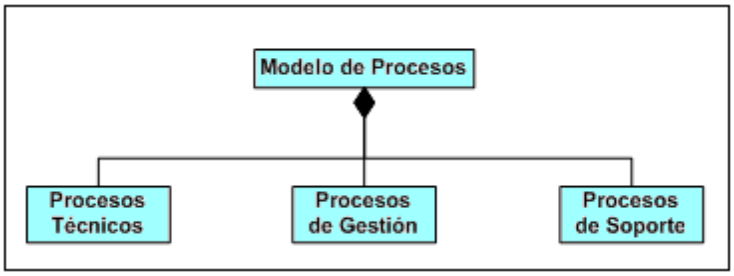


Figura 14. Procesos del Método Watch.
Fuente: Montilva *et al*., 2008.

Procesos técnicos

Los procesos técnicos se encargan de organizar las actividades tecnológicas que caracterizan el desarrollo de una aplicación empresarial cualquiera e incluye los siguientes procesos: modelado de negocio, agrupa a las actividades

encargadas de caracterizar y entender el dominio de la aplicación, es decir, el sistema de negocios para el cual se desarrolla la aplicación; ingeniería de requisitos, incluye todas las actividades necesarias para identificar, analizar, especificar, validar y gestionar los requisitos que se le imponen a la aplicación; diseño arquitectónico, congrega las actividades necesarias para especificar, diseñar y documentar la arquitectura de software que debe tener la aplicación; diseño de componentes, organiza todas las actividades de diseño detallado de los componentes arquitectónicos relacionados con la interfaz gráfica de la aplicación, sus componentes de software, su base de datos y su interacción con otras aplicaciones; programación e integración, agrupa las actividades de diseño detallado, codificación y prueba unitaria de cada uno de los componentes de software que integran la arquitectura de la aplicación, así como las actividades de integración y prueba de la integración de esto componentes; pruebas de la aplicación; ordena las actividades de pruebas de la aplicación como un todo, incluyendo las pruebas funcionales, no funcionales y de aceptación de la aplicación; entrega de la aplicación, estructura el conjunto de actividades que preceden a la puesta en producción de la aplicación, incluye la capacitación de usuarios, la instalación de la aplicación en su plataforma de producción u operación, las pruebas de instalación y la entrega final del producto.

Procesos de gestión

El grupo de procesos de gestión apoya la ejecución de todos los procesos técnicos y está relacionado con la gestión del proyecto. Se encarga de administrar el alcance, los tiempos, los costos, los recursos humanos y demás recursos que se requieran para desarrollar la aplicación. Este grupo incluye los siguientes procesos: constitución del proyecto, establece las actividades necesarias para promover, justificar, aprobar e iniciar el proyecto; planificación del proyecto, incluye las actividades encargadas de la planificación del alcance, tiempos, recursos humanos, otros recursos y servicios que requiera el

desarrollo de la aplicación; dirección del proyecto, agrupa las actividades de conformación del equipo de trabajo, capacitación del personal que integra estos equipos, administración de contratos con terceros, coordinación de la ejecución de las actividades del proyecto y administración de los recursos asignados al proyecto, entre otros; control del proyecto, contiene las actividades necesarias para supervisar y controlar el alcance tiempos, costos, recursos humanos y demás recursos que han sido asignados al proyecto; cierre del proyecto, organiza las actividades que se requieren para cerrar administrativamente y técnicamente el proyecto, una vez que concluya el desarrollo completo de la aplicación.

Procesos de soporte

El grupo de procesos de soporte complementan los procesos de gestión y, al que estos últimos, apoyan la ejecución de todos los procesos técnicos. Este grupo se relaciona con la calidad, los riesgos y la configuración de la aplicación. Incluye los siguientes procesos: gestión de riesgos, agrupa las actividades necesarias para identificar, analizar, planificar respuestas, monitorear y controlar todos aquellos riesgos o eventos que puedan afectar negativamente el proyecto; gestión de la configuración, organiza las actividades encargadas del control de los cambios que puedan surgir en la configuración de la aplicación, es decir, en los diferentes ítems o productos que la integran y que se desarrollan a lo largo del proyecto; gestión de la calidad, contempla las actividades necesarias para garantizar la calidad de la aplicación y todos los productos que la integran, así como la calidad del proceso usado para producir estos productos, dicho procesos está relacionado con las actividades de aseguramiento de la calidad de software y la verificación y validación del software.

El orden en que los procesos del método se ejecutan está inspirado en la metáfora del reloj; metáfora en la cual el proceso de desarrollo de software es

visto como un reloj, cuyo motor son los procesos de gestión y soporte y cuyos diales constituyen los procesos técnicos ya descritos. Esta metáfora determina la estructura del modelo de procesos. Ver Figura 15.

De acuerdo a la estructura del modelo, el proceso de desarrollo de software se inicia con la constitución y planificación del proyecto, la cual es parte de los procesos de gestión. Una vez planificado el proyecto, se da inicio a sus procesos técnicos mediante la ejecución del Modelado del Negocio. Se continua, luego, con los procesos de Ingeniería de Requisitos, Diseño Arquitectónico, Diseño Detallado, Programación e Integración y Pruebas de la Aplicación, en el orden indicado por las agujas del reloj; finalizando con la Entrega de la Aplicación.

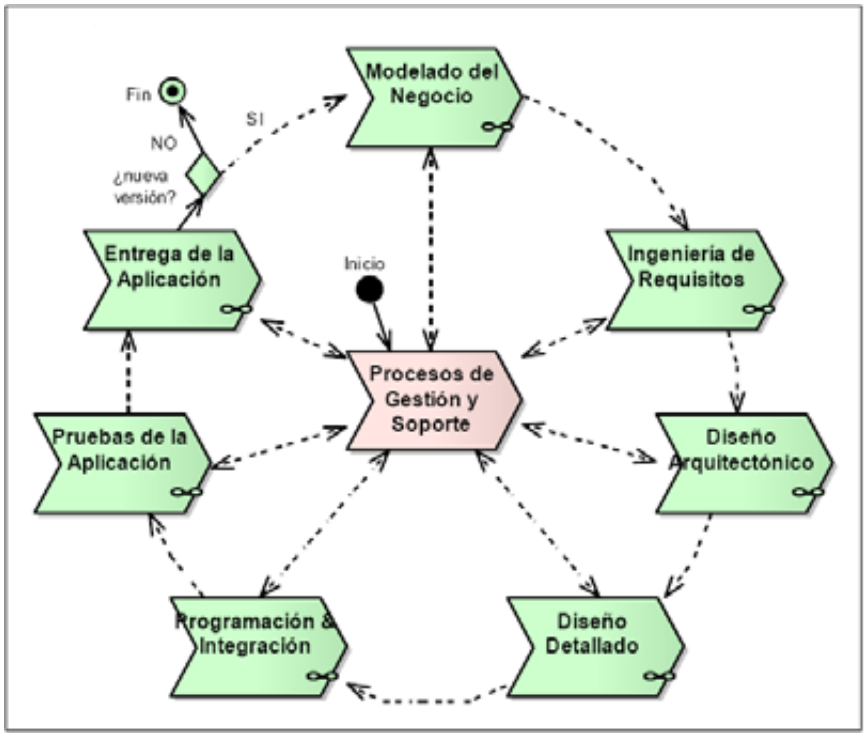


Figura 15. Estructura del Modelo de Procesos.
Fuente: Montilva *et al*., 2008.

Como puede observarse, en la Figura 16, el orden de ejecución es cíclico, es decir, la aplicación se desarrolla mediante la entrega de una o más versiones de la aplicación. Cada ciclo de desarrollo produce una nueva versión operativa de la aplicación. Una versión es un producto operativo, esto es, ejecutable y que provee ciertos servicios a sus usuarios.

Cada nueva versión le agrega, a la anterior, nuevos servicios o funciones. Los ciclos de desarrollo se repiten hasta completar al conjunto total de servicios o funciones que demandan sus usuarios y que están indicados en la arquitectura de la aplicación.

El proyecto culmina cuando se entrega la última versión prevista de la aplicación. Las versiones definen el carácter versionado o cíclico del método. Cada versión, a su vez, está compuesta de uno o más incrementos de software. Un incremento es una pieza de software que ejecuta un conjunto de funciones de la versión y que es usada, por los usuarios, para validar las funciones implementadas por el incremento, familiarizarse con la interfaz gráfica de la aplicación; y/o usarla para apoyar la ejecución de procesos de negocio. Los incrementos definen el carácter incremental del método. Uno de los procesos de soporte, denominado Verificación y Validación (V&V), se encarga de evaluar cada producto de los procesos técnicos, a fin de determinar si el proceso continúa hacia el siguiente proceso ó debe retornarse a un proceso anterior para corregir defectos en los productos. El carácter iterativo del método es determinado, en parte, por el proceso V&V.

Los métodos de desarrollo de software son modelos que guían a los equipos de trabajo en la definición del proceso más adecuado para llevar a cabo el desarrollo de un proyecto particular. Este proceso de adecuación se denomina instanciación y consiste el adaptar el conjunto de procesos y actividades prescritas por el método, a las características particulares de la aplicación empresarial que se quiere desarrollar. El proceso de instanciación especifica

que para utilizar apropiadamente el método *WATCH*, deben ejecutarse previamente cuatro subprocesos complementarios; éstos describen las actividades que se deben hacer para adaptar los modelos de producto, de actores y de procesos, a un proyecto particular (Montilva *et al*., 2008).

La instanciación del modelo de producto consistió en la selección de los productos concretos que se producirán durante todo el proceso de desarrollo de la aplicación.

El modelo de proceso resultante de la adaptación del modelo de procesos del método, especifica las actividades que debe seguir el equipo de trabajo para desarrollar la aplicación empresarial prevista, y servirá de base para la planificación del proyecto.

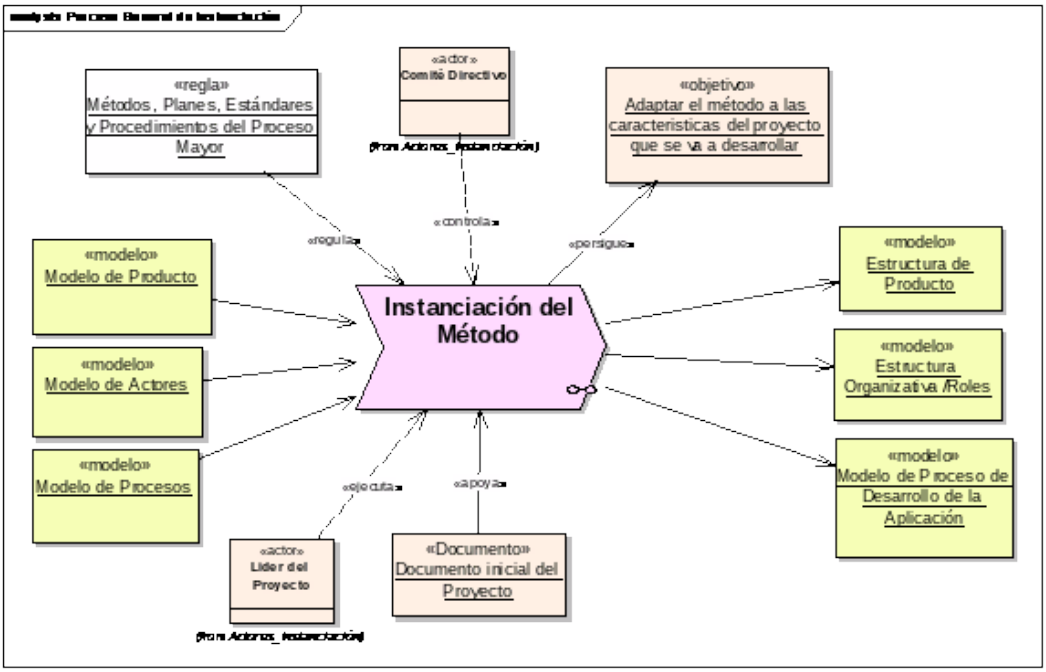


Figura 16. Proceso de Instanciación del Método Watch.
Fuente: Montilva *et al*., 2008.

Dicha instanciación dio como resultado la aplicación de *Blue WATCH* el cual es un marco de trabajo metodológico para el desarrollo de aplicaciones

empresariales de mediana complejidad y/o tamaño.

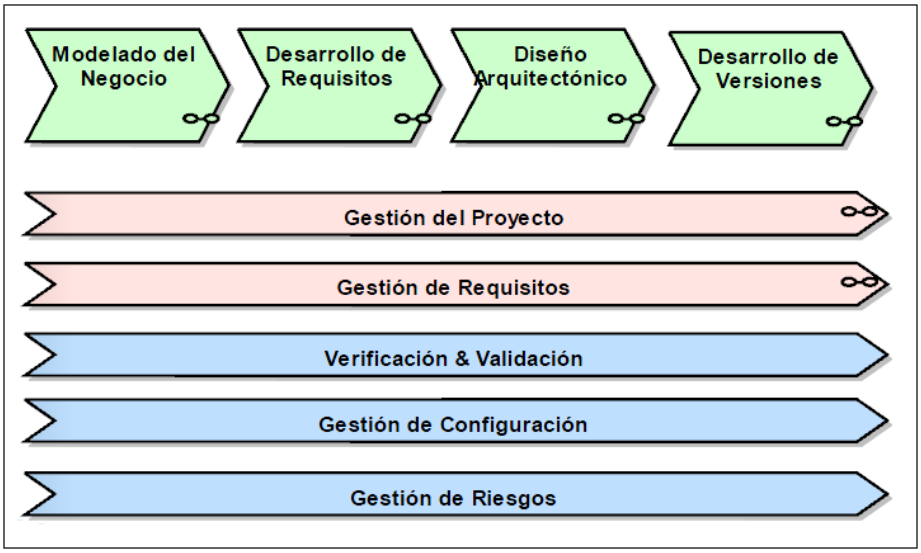


Figura 17. Cadena de Valor de los procesos del Método Watch.
Fuente: Montilva *et al*., 2008.

Este proceso ayudo al desarrollo del sistema de información bajo ambiente web para el incentivo, promoción y apoyo al sector turismo mediante la obtención de productos en el ciclo de aplicación, ciclo de versión y ciclo de incremento, tal como lo muestra la Figura 18.

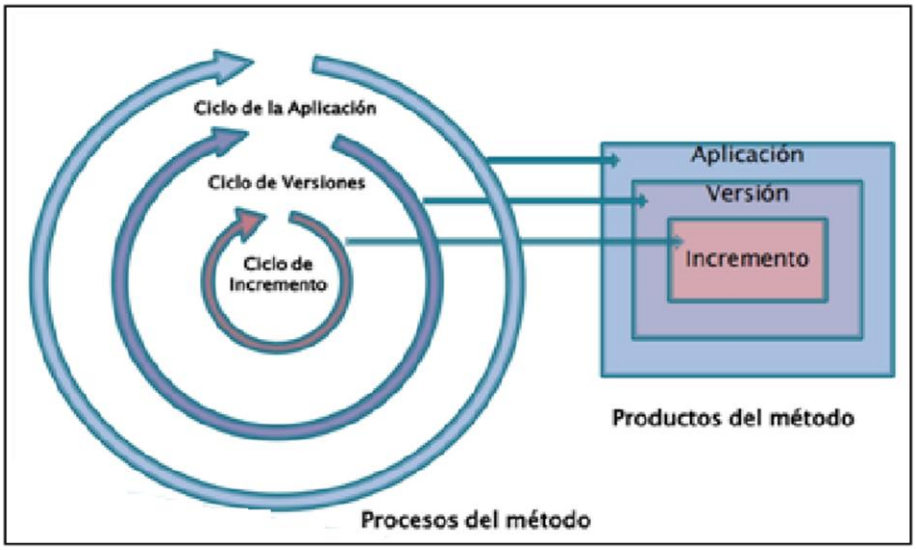


Figura 18. Relación entre procesos y productos del Método Watch.
Fuente: Montilva *et al*., 2008.

De igual forma cada ciclo posee un conjunto de procesos en los cuales se generan un conjunto de productos. En el ciclo de incremento se produce una pieza de código ejecutable denominada incremento, en el ciclo de versión se produce una versión completa y operativa de la aplicación incorporando un conjunto de incrementos y mediante los productos evolutivos se obtiene el ciclo de aplicación que se genera mediante la agregación progresiva de versiones. El tiempo estimado para generar una versión está entre uno (1) y tres (3) meses, de la misma forma para obtener un ciclo de incremento se tienen entre una (1) y tres (3) semanas (Montilva *et al*., 2014).

Los procesos de la cadena de valor presentados en la Figura 17 se ejecutan cíclicamente de la manera señalada en la Figura 19. Nótese que los procesos técnicos se ubican en el anillo externo del ciclo, siguiendo la forma del dial de un reloj; mientras que los procesos de gestión y soporte, están ubicados en el centro (Montilva *et al*., 2014).

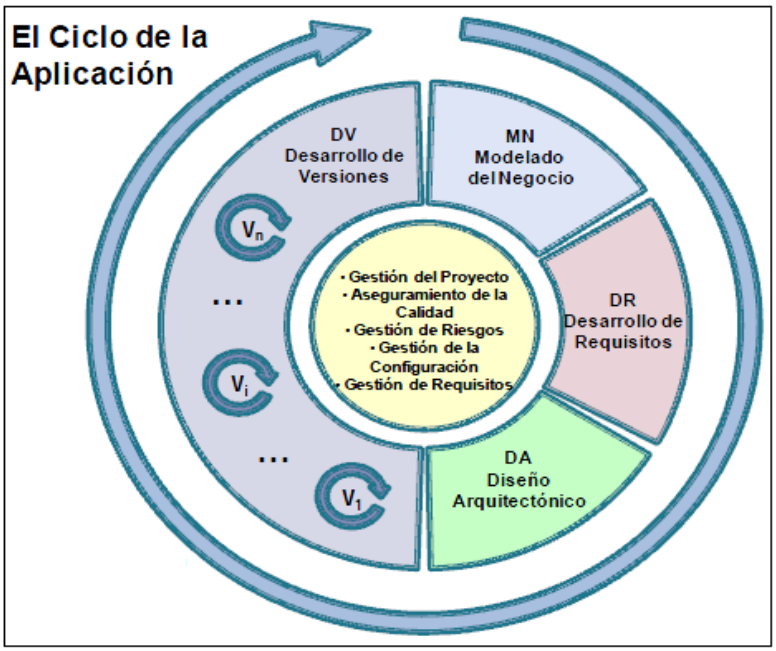


Figura 19. Procesos del Ciclo de la Aplicación.
Fuente: Montilva *et al*., 2008.

El proceso de desarrollo de versiones es iterativo y consta de un conjunto de ciclos de versiones que se van desarrollando secuencialmente. El número de versiones depende de las características de cada proyecto y aplicación. Cada ciclo de versión ejecuta secuencialmente el conjunto de procesos técnicos que se señalan en la Figura 20. Los cuatro primeros procesos refinan los productos elaborados previamente durante el ciclo de la aplicación.

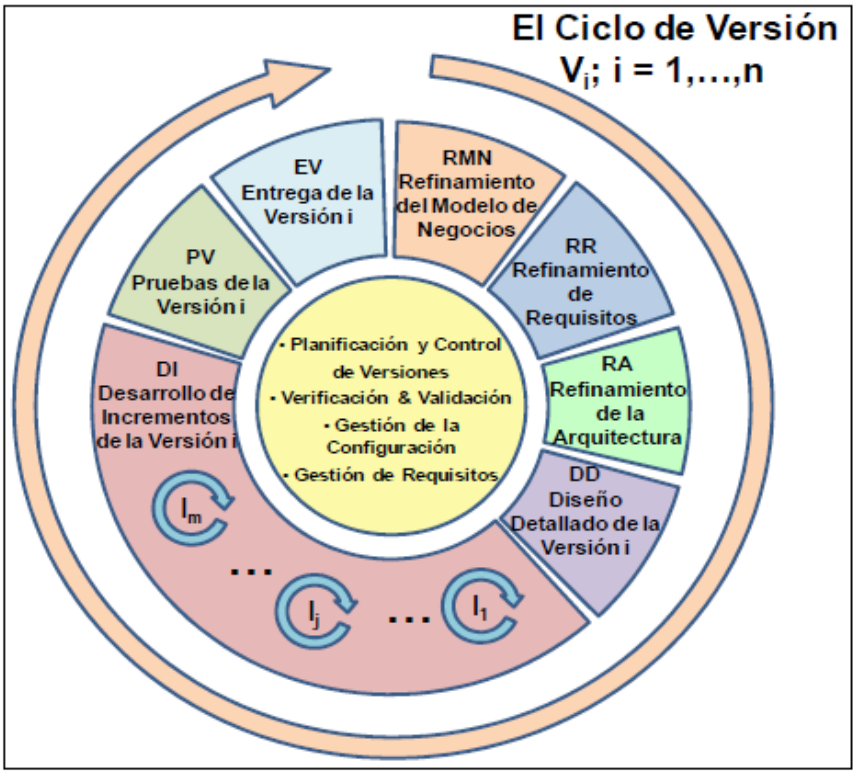


Figura 20. Procesos relacionados con el Ciclo de Versión.
Fuente: Montilva *et al*., 2008.

Así como se tiene la cadena de valor representada con los procesos pertenecientes al ciclo de aplicación, se tienen también la referente al ciclo de versión, ver Figura 20. Allí se puede evidenciar que el desarrollo de versiones está compuesto por los procesos representados en la Figura 21, los cuales a su vez están apoyados con las actividades de planificación, control, verificación y validación pertenecientes a los procesos de gestión y soporte respectivamente.

El proceso de desarrollo de incrementos, que se realiza en cada ciclo de versión, es también iterativo. Consta de un conjunto de ciclos de incrementos que se pueden desarrollar secuencialmente o en paralelo, dependiendo de la cantidad de programadores disponibles en el proyecto, ver Figura 22. Mientras que la Figura 23, señala el orden en el cual se ejecutan los procesos técnicos encargados de elaborar un incremento.

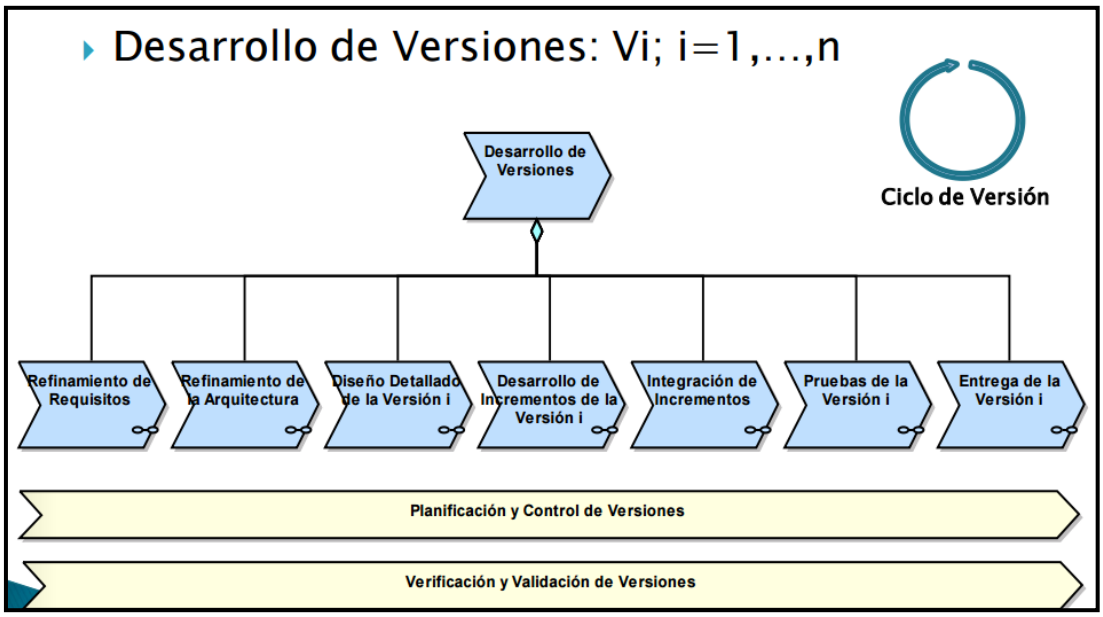


Figura 21. Cadena de Valor del Ciclo de Versión.
Fuente: Montilva *et al*., 2008.

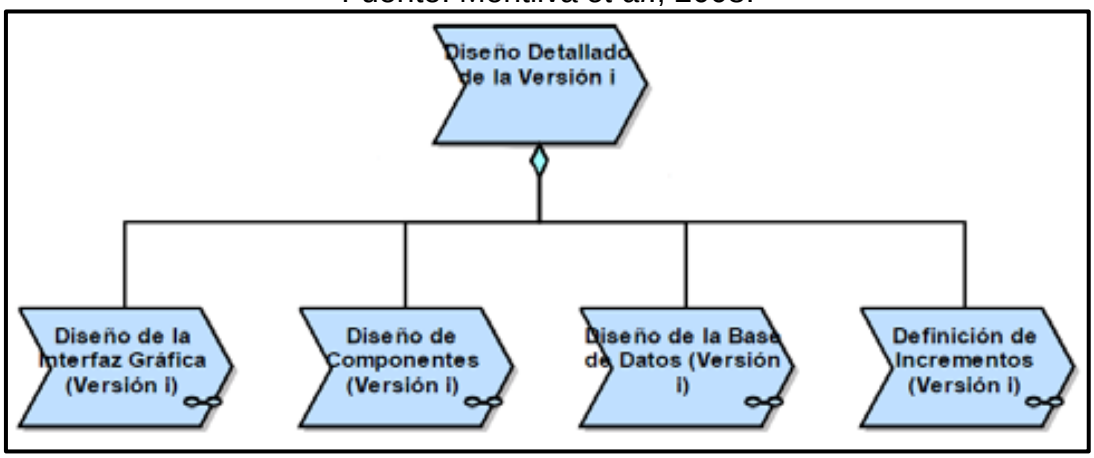


Figura 22. Descripción de procesos de Diseño Detallado del Ciclo de Versión.
Fuente: Montilva *et al*., 2008.

Al igual que en el ciclo de versión, el ciclo de incremento especifica mediante la cadena de valor los procesos técnicos mostrados en la Figura 21, de gestión y soporte como la planificación, control, verificación y validación de los incrementos, ver Figura 23 y 24.

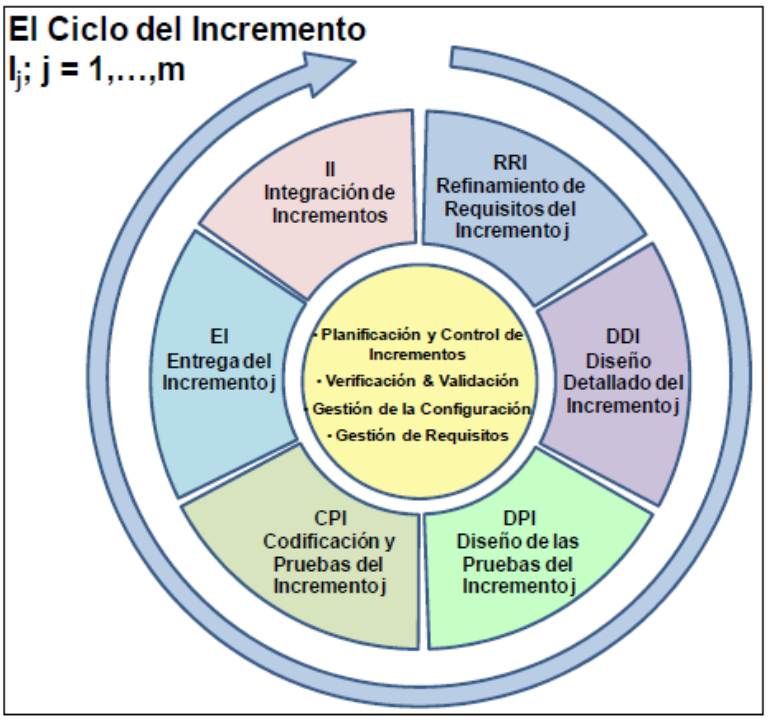


Figura 23. Procesos relacionados con el Ciclo de Incremento.
Fuente: Montilva *et al*., 2008.

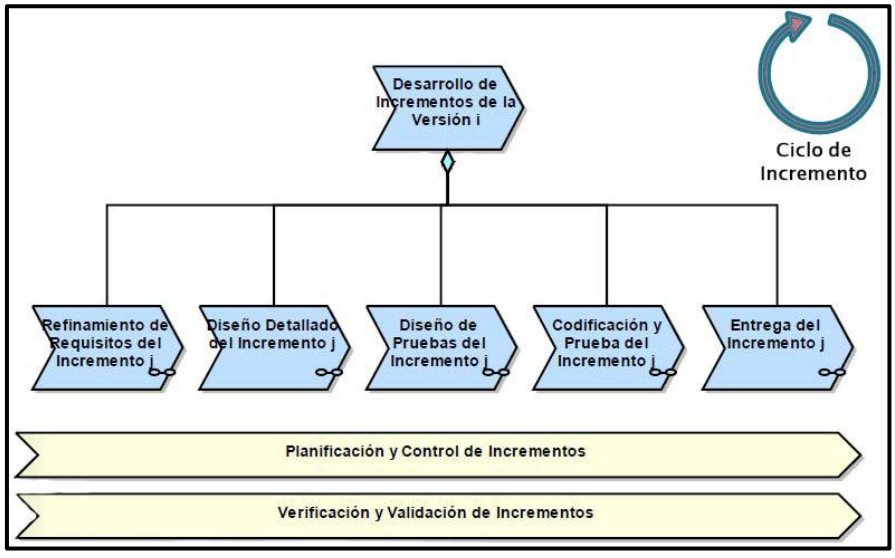


Figura 24. Descripción de Desarrollo del Ciclo de Incrementos.
Fuente: Montilva *et al*., 2008.

CAPÍTULO III. DESARROLLO

3.1. MODELO PRINCIPAL DE PROCESOS.

El presente capítulo consistió en el desarrollo de la aplicación, la cual está soportada por la secuencia de ejecución de las fases de la metodología *Blue Watch* siguiendo el Modelo Principal de Procesos que describe el orden o secuencia en que los procesos técnicos, de gestión y de soporte del desarrollo de la aplicación se ejecutaron, siguiendo el orden descrito a continuación.

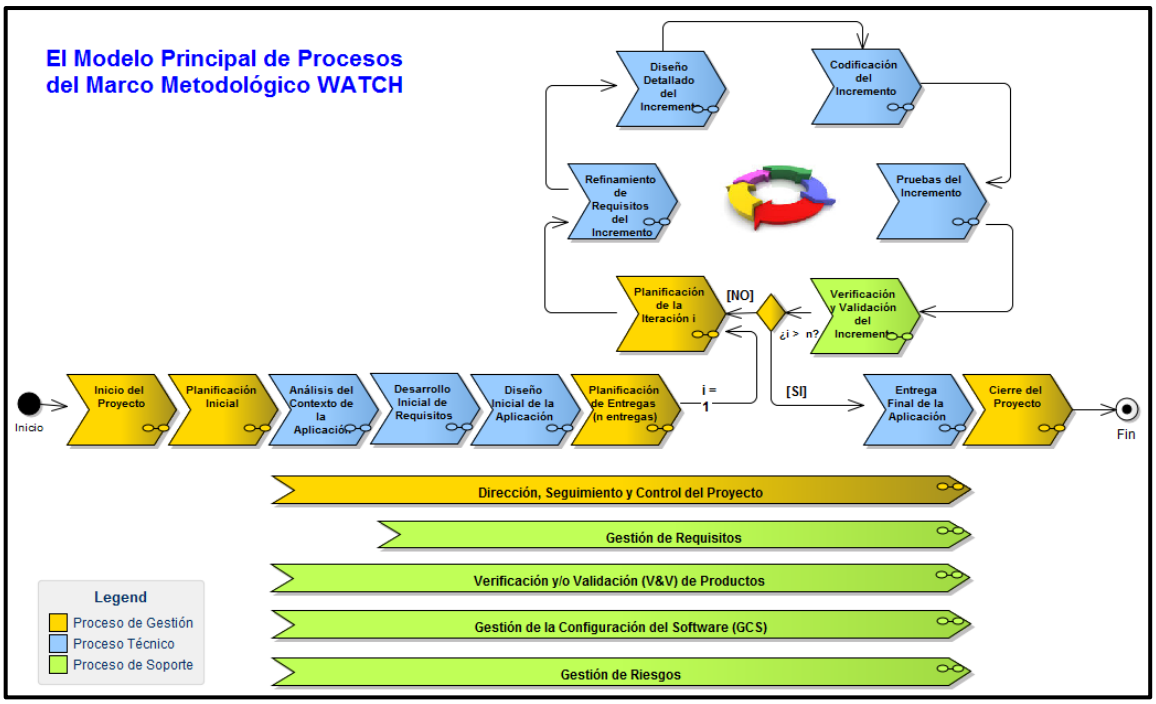


Figura 25. Modelo principal de procesos del marco metodológico Blue Watch. Fuente: Montilva y Barrios, 2014.

El desarrollo se inicia con dos procesos de gestión: Inicio del Proyecto y Planificación Inicial del Proyecto, los cuales obtuvieron como resultado, respectivamente, los informes de Visión del Producto y Plan Inicial del Proyecto.

3.1.1. Inicio del Proyecto.

El inicio del proyecto comprendió un conjunto de actividades de gestión previas o preliminares, que permitieron activar el desarrollo del proyecto de la nueva aplicación web denominada Softour. Lo cual se logró al dar inicio a las Alternativas de Grado en la Licenciatura en Informática, en donde un grupo de profesores coordinaron y asumieron el papel inicial de Gerentes de Desarrollo y Mantenimiento de Software, para controlar y aprobar dicho proceso.

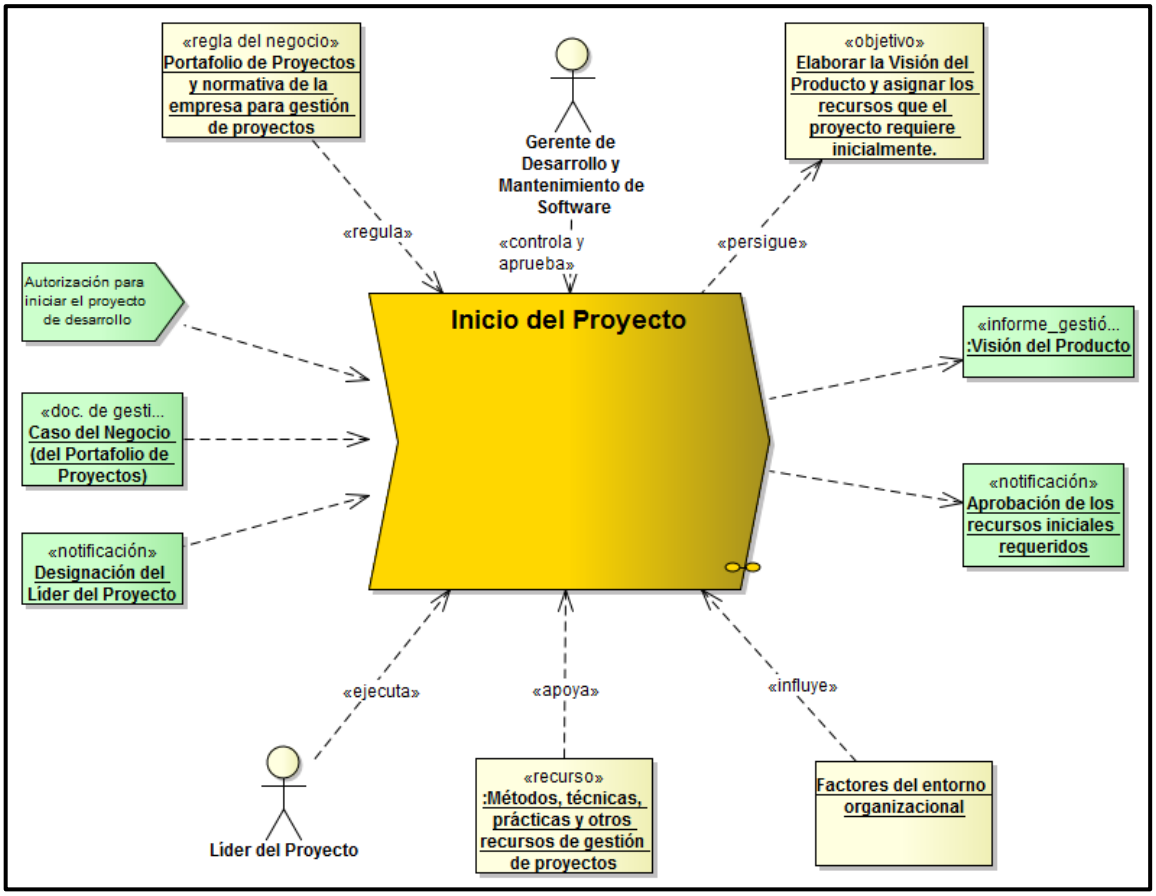


Figura 26. Diagrama de análisis de Inicio del Proyecto.
Fuente: Montilva y Barrios, 2014.

Considerando los conocimientos que se emprenderían en las alternativas de grado y en base al portafolio de proyectos que se había diseñado y que estaban disponibles para su desarrollo, se seleccionó uno de las temáticas relacionadas

con el incentivo, la promoción y el apoyo al sector turismo.

Se establecieron las normativas para el desarrollo de las Alternativas de Grado, resaltando la reglamentación de la misma para la gestión y regulación de los proyectos de trabajo de grado, fundamentadas como reglas de negocio.

Las Alternativas de Grado estuvieron conformadas por tres asignaturas denominadas:

Organización y Dirección de la Innovación Tecnológica, en donde se desarrollaron las siguientes competencias: diseñar modelos gerenciales en el contexto organizacional basándose en la investigación científica, desarrollo e innovación tecnológica; relacionar las teorías organizacionales con la creatividad empresarial y gestión del conocimiento; planificar la investigación científica, desarrollo e innovación tecnológica (I+D+I) en el contexto organizacional; proponer tendencias administrativas ajustadas a casos de estudio organizacionales.

Sistema de Información Gerencial, en donde se abarcaron las siguientes competencias: desarrollar sistemas de información gerenciales facilitando el proceso de toma de decisiones dentro de una organización; diseñar modelos gerenciales en el contexto organizacional basándose en la investigación científica, desarrollo e innovación tecnológica de las tecnologías informáticas y computacionales; precisar las relaciones entre la gerencia de una organización y los sistemas de información gerencial; seleccionar las tecnologías de la información en el contexto gerencial, manteniendo a la organización adaptada al entorno; estructurar sistemas de información gerenciales apegados a un caso práctico.

Inteligencia de Negocios, en donde se desarrollaron las siguientes competencias: diseñar herramientas de extracción de información útil en el proceso de tomas de decisiones estratégicas ajustadas al enfoque de la

Inteligencia de Negocios; precisar, con carácter crítico, los indicadores de gestión que permitan la evaluación continua del desempeño en una organización; relacionar los basamentos de la inteligencia de negocios con casos de estudio; proponer herramientas de extracción de la información en el contexto de sistemas organizacionales.

Luego se designó como Líder del Proyecto a la Br. Arnella José Gómez Arias, lo que permitió se constituyera un Comité Directivo del Proyecto y a la Estructura del Equipo de Desarrollo, para dar inicio a la ejecución del mismo.

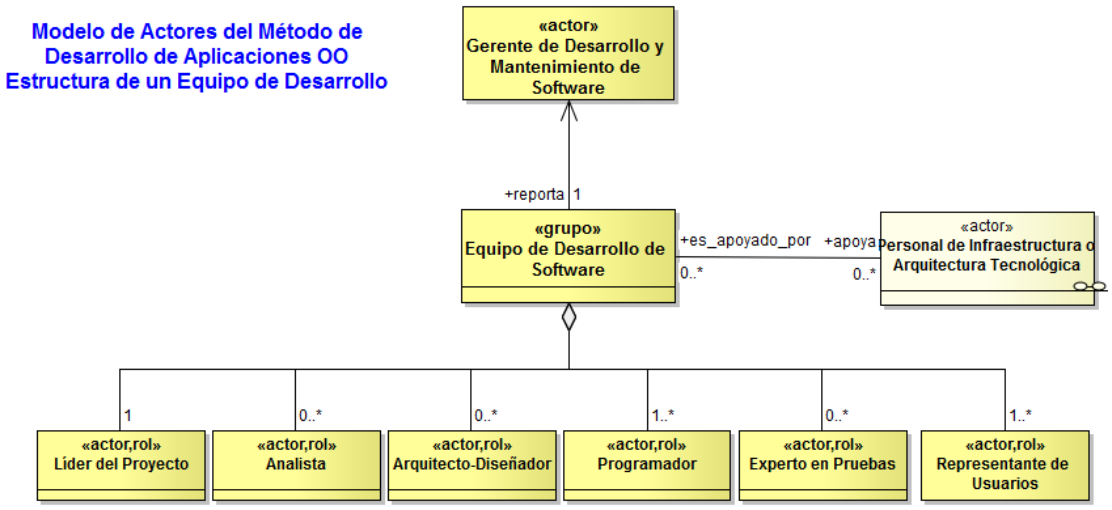


Figura 27. Diagrama de actores y estructura del equipo de desarrollo.
Fuente: Montilva y Barrios, 2014.

Se utilizaron métodos, técnicas, prácticas y herramientas como recursos para apoyar el inicio del proyecto, entre las cuales se encuentran: entrevistas no estructuradas, reuniones grupales, estructuras de grupos, técnicas de dinámica de grupos, investigación documentada de la problemática, herramientas ofimáticas, herramientas de modelado, herramientas de gestión de proyectos, formatos preestablecidos, técnicas de elaboración de documentos técnicos, métodos para la evaluación y selección del software requerido para el desarrollo de aplicaciones según el ámbito de estudio, entre otros.

Lo que permitió a la Líder del Proyecto, junto con sus colaboradores, obtener el objetivo final del proceso Inicio del Proyecto, que fue elaborar una Visión del Producto más avanzada y completa que la disponible en el Caso de Negocio extraído del Portafolio de Proyectos de la Licenciatura en Informática, la cual representaba un breve bosquejo del proyecto propuesto por los profesores de las alternativas de grado. Se le dio el visto bueno y se lograron asignar los recursos necesarios para iniciar el proyecto.

Las reglas de negocio que regularon el inicio del proyecto fueron el portafolio de proyectos y la normativa de la Licenciatura en Informática para la gestión de proyectos como tesis de grado en la modalidad alternativas de grado.

3.1.2. Planificación Inicial del Proyecto.

La Planificación Inicial del Proyecto comprendió un conjunto de actividades de Gestión del Proyecto que se ejecutaron al inicio del desarrollo de la nueva aplicación web Softour, con la finalidad de elaborar el Plan Inicial del Proyecto.

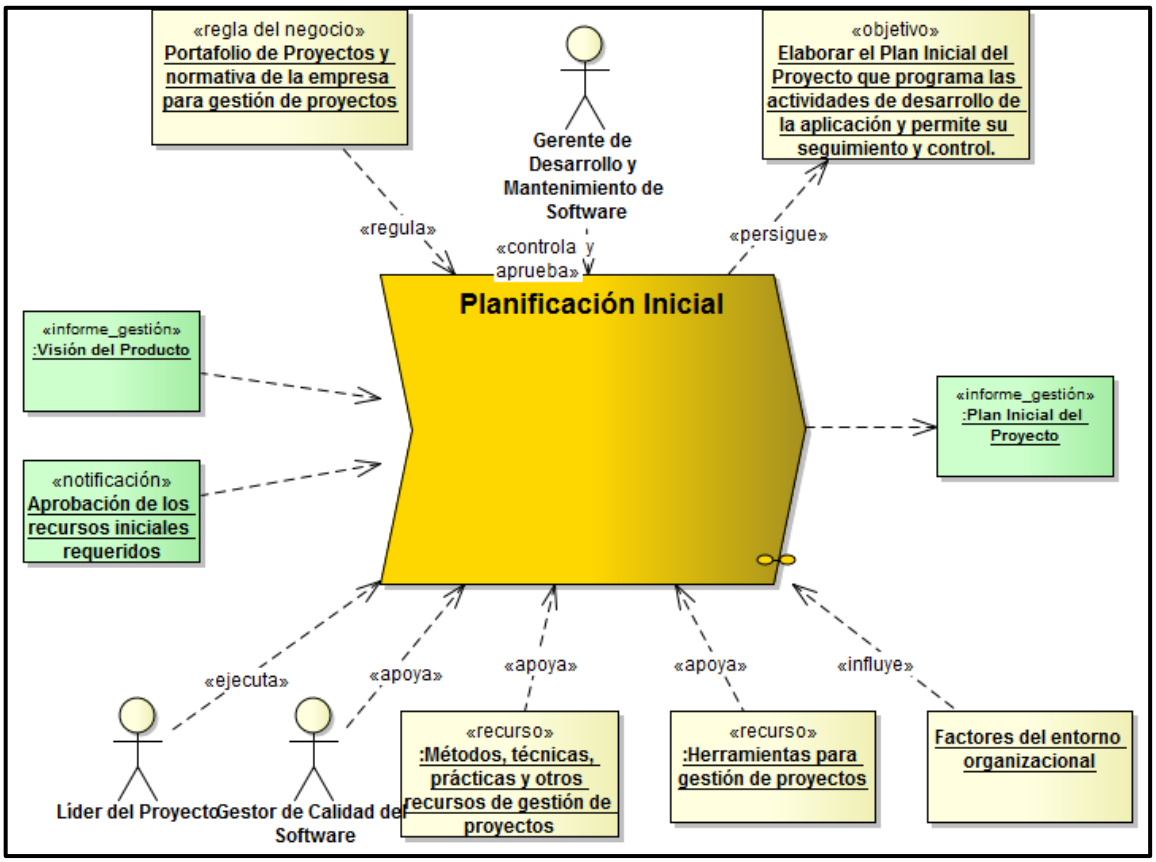


Figura 28. Diagrama de análisis de Planificación Inicial del Proyecto.
Fuente: Montilva y Barrios, 2014.

A través de la planificación del proyecto se lograron establecer los objetivos y las delimitaciones del proyecto, desarrollando el proceso de estructuración del plan del proyecto, el cual define, integra y coordina un conjunto de planes suplementarios (Montilva *et al*., 2014). Para estructurar el plan inicial de este proyecto se cumplieron con las actividades, que a continuación se muestran:

Se estableció el alcance del proyecto, en donde se identificaron y describieron los productos del proyecto, generando una estructura de desglose de productos, y el esfuerzo requerido para elaborar estos productos, generando la estructura de desglose del trabajo.

3.1.1.1. Estructura de Desglose de Productos.

Los productos del proyecto se basaron en tres versiones del producto, las cuales describen las características generales del producto Softour, en donde cada una de ellas se enfoca en los diferentes tipos de usuarios y su perfil:

La primera versión abarcó el perfil del usuario administrador, resaltando el desarrollo en los procesos CRUD del mantenimiento de las colecciones de datos, donde se desarrollaron los procesos de ingresar (Create), consultar (Read), modificar (Update) y eliminar (Deleted) dichas colecciones, las cuales están asociadas a todas aquellas entidades relacionadas con lugares, eventos, experiencias, gastronomía, alojamientos y la planificación de los viajes, además del desarrollo de un módulo denominado Dashboard en donde se consideraron aspectos de la inteligencia de negocios asociados al desempeño, uso y rendimiento de la aplicación Softour.

La segunda versión abarcó el perfil del usuario prestador de servicios, en donde se limita el desarrollo en los procesos CRUD del mantenimiento de las colecciones de datos solo a las entidades asociadas a dicho perfil, desarrollando los procesos de ingresar (Create), consultar (Read), modificar (Update) y eliminar (Deleted) dichas colecciones asociadas a la entidades relacionadas con lugares, eventos, experiencias, gastronomía y alojamientos publicados por dicho prestador de servicios, además del desarrollo de un módulo denominado Dashboard en donde se consideraron aspectos de la inteligencia de negocios asociados al desempeño, uso y rendimiento de las publicaciones promocionadas por el prestador de servicios.

La tercera versión abarcó el perfil del usuario turista, considerado el usuario final por excelencia, en donde se estableció una interfaz versátil, vistosa, llamativa, dinámica y fácil de utilizar, en donde se resalta de manera gráfica los diferentes lugares, eventos, experiencias, gastronomía y alojamientos que han publicado o promocionado los prestadores de servicios, permitiendo al turista

elaborar una planificación de su viaje y estadía.

Se establecieron los tiempos de ejecución de los procesos y actividades relacionadas que se especificaron mediante el Cronograma del Proyecto.

Se organizaron los recursos humanos requeridos, que se representaron mediante los roles necesarios y la organización interna del Equipo de Desarrollo de la aplicación Softour.

3.1.1.2. Plan de Gestión de Alcance.

El Plan de Gestión de Alcance se basa en los objetivos propuestos para el desarrollo del proyecto Softour:

Identificar y describir los roles que se requieren para desarrollar la aplicación Softour.

Asignar las responsabilidades de cada rol y establecer la estructura para organizar los equipos de desarrollo.

Clasificar y describir el orden de ejecución de los procesos técnicos, de gestión y de soporte establecidos por el método Blue Watch.

Ejecutar los procesos de desarrollo según el orden establecido propuesto por el método Blue Watch para la aplicación Softour.

Elaborar la Visión del Producto para coordinar los recursos que el proyecto requiere durante su desarrollo.

Elaborar el Plan Inicial del Proyecto para programar las actividades de desarrollo de la aplicación para su seguimiento y control.

Coordinar la ejecución del proyecto para asegurar que las actividades sean realizadas de acuerdo a la planificación, los productos sean entregados a tiempo y se cumplan los requisitos de calidad establecidos.

Analizar los problemas y las oportunidades que dieron origen al desarrollo de la aplicación mediante el análisis del contexto del sector turismo enfocado prestar servicios a través de Softour.

Definir y especificar los requisitos de la aplicación, según las necesidades de los usuarios interesados, considerando las nuevas tendencias y propuestas de las aplicaciones web turísticas competitivas, incorporando los requerimientos de los prestadores de servicios en el desarrollo de la aplicación, prestando especial interés a las observaciones propuestas por los profesores que coordinan las alternativas de trabajo grado de la Licenciatura en Informática de la UDO, Núcleo de Sucre.

Diseñar la arquitectura de la aplicación y especificar sus componentes a un alto nivel de abstracción.

Definir el número de iteraciones que se van a realizar para producir los incrementos que serán entregados.

Programar las actividades de desarrollo del incremento de la aplicación correspondiente a cada una de las iteraciones definidas en el objetivo anterior.

Especificar los requisitos de los incrementos mediante sus actualizaciones, modelado y refinamiento.

Asegurar que los requisitos sean apropiadamente implementados, estén completos y actualizados.

Modelar los elementos que definen el dominio de la aplicación Softour en el sector turismo.

Elaborar las especificaciones de diseño de los componentes arquitectónicos que integran los incrementos de la aplicación.

Elaborar los programas del incremento de acuerdo a las especificaciones de requisitos y diseño detallado y probarlos a nivel de unidad e integración.

Generar la codificación de los modelos diseñados, según el orden de iteraciones previamente establecido, para que la aplicación cumpla con los requisitos de los usuarios y del modelo de negocio.

Detectar errores en el incremento codificado durante las iteraciones y corregir los errores encontrados.

Realizar las revisiones técnicas para determinar que el incremento cumpla con los requisitos establecidos y satisfaga las necesidades de sus usuarios.

Realizar los diferentes tipos de pruebas a nivel de integración de los incrementos para obtener las versiones y la aplicación final, en las cuales se deben incorporar a los programadores, analistas, expertos en pruebas, gestor de la calidad del software, gestor de configuración del software, gerente de desarrollo y mantenimiento de software, demás desarrolladores, representantes de usuarios y los usuarios interesados.

Evaluar los productos elaborados durante el desarrollo de la aplicación para asegurarse que cumplan con sus especificaciones y con las necesidades reales de los usuarios.

Controlar los cambios en los diferentes productos o ítems que se generaron durante el desarrollo de la aplicación.

Reducir la probabilidad de ocurrencia y el impacto negativo de los riesgos que pudieran afectar el normal desarrollo del proyecto.

Supervisar la ejecución del proyecto y corregir las desviaciones con respecto a lo establecido en el Plan del Proyecto.

3.1.1.3. Límites del Proyecto.

El sistema de información Web desarrollado está dirigido principalmente a los turistas como usuarios finales, a los prestadores de servicios como generadores

de contenidos de interés turístico y a los administradores del sistema como clientes finales interesados en la aplicación Softour.

Aunque los datos que inicialmente se ingresaron en la base de datos de Softour son referentes al Estado Sucre, ya que se requería un ejemplo, el software se adapta a cualquier zona geográfica del mundo.

Los aspectos que contempla la aplicación Softour están limitados por su funcionalidad en conceptos relacionados con lugares, eventos, experiencias, alojamientos, gastronomía y planificación de itinerarios, por lo que no contempla reservaciones de boletería de viajes.

3.1.1.4. Herramientas de Software para Gestionar el Proyecto.

Para trabajar de forma profesional en un proyecto, es necesario una solución en software para las actividades de gestión de proyectos, para ir más allá que el manejo de una simple lista de tareas o agendas tradicionales, considerando que es un trabajo que se desarrolla por un grupo de personas en equipo, se requiere de herramientas que sean estables, flexibles, potentes, gratuitas, amplias, compartidas y robustas. Por ello se decidió por las herramientas para la gestión del proyecto que están en boga como lo son: Trello, Slack, Whatsapp, GanttProject, Enterprise Architect, Sky, correo electrónico, Facebook, y otras utilizadas en menor medida.

3.1.1.5. Estructura de Costos Asociados.

Se determinaron los costos asociados al desarrollo de la aplicación que se especificaron en el presupuesto de gastos del proyecto. En este proceso, la estimación de costos se hizo más precisa con respecto a la establecida en la Visión del Producto y se pudo ubicar en las clases II (10 - 20%) o clase III (10 - 25%). Estos costos son estimativos y no reales debido a que el desarrollo se cubrió con gastos personales del equipo o grupo de desarrollo, especialmente en el uso de materiales de oficina, papelería, impresiones y recursos

computacionales propios, también se recibió ayuda gratuita de colaboradores con respecto a la disposición de instalaciones para reuniones y sesiones de trabajo diarias, además de recursos informáticos especializados como internet y computadoras de alto rendimiento.

3.1.1.6. Planificación de la Gestión de Tiempos.

Se consideró el tamaño y duración del proyecto y la normativa interna para la elaboración de proyectos de la Licenciatura en Informática, específicamente en el desarrollo de las alternativas de grado, para ejecutarse en un tiempo determinado por dos semestres regulares. Durante esta etapa se logró determinar el número de iteraciones y las actividades que conformarían el desarrollo del proyecto, así como también el establecimiento de la secuencia de las mismas. Para el desarrollo de este trabajo de grado se planificó un (1) ciclo de desarrollo de la aplicación, conformado por tres (3) ciclos de versiones, estas compuestas a su vez por un número determinado de ciclos de incrementos, para cada una de ellos se establecieron las actividades y los tiempos de ejecución, plasmados en el cronograma de actividades (ver apéndice A). Se describe a continuación la estructuración del desarrollo de las versiones e iteraciones para el ciclo de desarrollo de la aplicación Softour versión 1.0:

Tabla 1. Versión del sistema para el módulo Administrador.

Iteración Compuesta	Iteración Base
Iteración 1: CRUD de usuarios.	Iteración 1.1: Búsquedas asociadas al usuario. Iteración 1.2: Ingresar usuario. Iteración 1.3: Consultar usuario. Iteración 1.4: Modificar usuarios. Iteración 1.5: Eliminar usuario.
Iteración 2: CRUD de lugares.	Iteración 2.1: Búsquedas asociadas a lugares. Iteración 2.2: Ingresar lugar. Iteración 2.3: Consultar lugar. Iteración 2.4: Modificar lugar. Iteración 2.5: Eliminar lugar.
Iteración 3: CRUD de eventos.	Iteración 3.1: Búsquedas asociadas a eventos. Iteración 3.2: Ingresar evento. Iteración 3.3: Consultar evento. Iteración 3.4: Modificar evento. Iteración 3.5: Eliminar evento.
Iteración 4: CRUD de experiencias.	Iteración 4.1: Búsquedas asociadas a experiencias. Iteración 4.2: Ingresar experiencia. Iteración 4.3: Consultar experiencia. Iteración 4.4: Modificar experiencia. Iteración 4.5: Eliminar experiencia.
Iteración 5: CRUD de alojamientos.	Iteración 5.1: Búsquedas asociadas a alojamientos. Iteración 5.2: Ingresar alojamiento. Iteración 5.3: Consultar alojamiento. Iteración 5.4: Modificar alojamiento. Iteración 5.5: Eliminar alojamiento.
Iteración 6: CRUD de gastronomía.	Iteración 6.1: Búsquedas asociadas a gastronomía. Iteración 6.2: Ingresar gastronomía. Iteración 6.3: Consultar gastronomía. Iteración 6.4: Modificar gastronomía. Iteración 6.5: Eliminar gastronomía.
Iteración 7: Inicio de sesión	Iteración 7.1: Búsquedas asociadas a perfiles de usuarios. Iteración 7.2: Inicio de sesión por perfiles de usuarios.
Iteración 8: Dashboard	Iteración 8.1: Búsquedas asociadas a las tablas de datos para generar indicadores del Dashboard. Iteración 8.2: Creación de carteles en el Dashboard. Iteración 8.3: Creación de gráficas en el Dashboard.

Tabla 2. Versión del sistema para el módulo Prestador de Servicios.

Iteración Compuesta	Iteración Base
Iteración 9: Configuración de cuenta del Prestador de Servicios.	Iteración 9.1: Búsquedas asociadas al perfil. Iteración 9.2: Consultar datos de cuenta. Iteración 9.3: Modificar datos de cuenta. Iteración 9.4: Darse de baja (Eliminar cuenta de manera lógica del Prestador de Servicios).
Iteración 10: CRUD de eventos asociados al perfil del Prestador de Servicios.	Iteración 10.1: Búsquedas asociadas a eventos. Iteración 10.2: Crear nuevo evento. Iteración 10.3: Consultar evento. Iteración 10.4: Modificar evento. Iteración 10.5: Cerrar o deshabilitar evento.
Iteración 11: CRUD de experiencias asociadas al perfil del Prestador de Servicios.	Iteración 11.1: Búsquedas asociadas a experiencias. Iteración 11.2: Crear nueva experiencia. Iteración 11.3: Consultar experiencia. Iteración 11.4: Modificar experiencia. Iteración 11.5: Cerrar o deshabilitar experiencia.
Iteración 12: CRUD de alojamientos asociadas al perfil del Prestador de Servicios.	Iteración 12.1: Búsquedas asociadas a alojamientos. Iteración 12.2: Crear nuevo alojamiento. Iteración 12.3: Consultar alojamiento. Iteración 12.4: Modificar alojamiento. Iteración 12.5: Cerrar o deshabilitar alojamiento.
Iteración 13: CRUD de gastronomía asociada al perfil del Prestador de Servicios.	Iteración 13.1: Búsquedas asociada a gastronomía. Iteración 13.2: Crear nueva gastronomía. Iteración 13.3: Consultar gastronomía. Iteración 13.4: Modificar gastronomía. Iteración 13.5: Cerrar o deshabilitar gastronomía.
Iteración 14: Dashboard	Iteración 14.1: Búsquedas asociadas a la creación de tablas con datos base para generar los indicadores del Dashboard. Iteración 14.2: Creación de carteles de indicadores en el Dashboard del Prestador de Servicios. Iteración 14.3: Creación de gráficas de indicadores en el Dashboard del Prestador de Servicios.

Tabla 3. Versión del sistema para el módulo Turista.

Iteración Compuesta	Iteración Base
Iteración 15: Módulo lugares.	Iteración 15.1: Vista inicial de presentación de lugares. Iteración 15.2: Filtros de búsqueda avanzada de lugares. Iteración 15.3: Consulta de lugar. Iteración 15.4: Enviar lugar hacia el itinerario. Iteración 15.5: Calificación de lugar visitado.
Iteración 16: Módulo eventos.	Iteración 16.1: Vista inicial de presentación de eventos. Iteración 16.2: Filtros de búsqueda avanzada de eventos. Iteración 16.3: Consulta de evento. Iteración 16.4: Enviar evento hacia el itinerario. Iteración 16.5: Calificación de eventos que ha participado.
Iteración 17: Módulo experiencias.	Iteración 17.1: Vista inicial de presentación de experiencias. Iteración 17.2: Filtros de búsqueda avanzada de experiencias. Iteración 17.3: Consulta de experiencia. Iteración 17.4: Enviar experiencia hacia el itinerario. Iteración 17.5: Calificación de experiencias vividas.
Iteración 18: Módulo alojamientos.	Iteración 18.1: Vista inicial de presentación de alojamientos. Iteración 18.2: Filtros de búsqueda avanzada de alojamientos. Iteración 18.3: Consulta de alojamiento. Iteración 18.4: Enviar alojamientos hacia el itinerario. Iteración 18.5: Calificación de alojamientos utilizados.
Iteración 19: Módulo gastronomía.	Iteración 19.1: Vista inicial de presentación de gastronomía. Iteración 19.2: Filtros de búsqueda avanzada de gastronomía. Iteración 19.3: Consulta de gastronomía. Iteración 19.4: Enviar gastronomía hacia el itinerario. Iteración 19.5: Calificación de gastronomía.
Iteración 20: Módulo itinerario.	Iteración 20.1: Vista inicial de presentación de itinerario. Iteración 20.2: Ingresar nuevo elemento al itinerario. Iteración 20.3: Consultar itinerario. Iteración 20.4: Eliminar elementos del itinerario. Iteración 20.5: Eliminar itinerario.

3.1.1.7. Planes Suplementarios.

Se elaborar documentos que cumplieron con los planes suplementarios, tales como: Plan de Gestión de Requisitos, Plan de Gestión de Riesgos, Plan de Gestión de la Configuración, Plan de Verificación y Validación, Plan de Contrataciones, Plan de Adquisiciones y Compras.

Luego de los procesos de gestión, se continuaron con un conjunto de tres procesos técnicos: Análisis del Contexto de la Aplicación (Modelado del Problema), Desarrollo Inicial de Requisitos y Diseño Inicial de la Aplicación. Estos procesos producen los informes de Descripción del Problema, Documento de Requisitos y Documento de Diseño, respectivamente. Los dos últimos tienen un carácter inicial; pues, fueron mejorados o ampliados durante el proceso posterior de Desarrollo de Incrementos.

3.1.3. Análisis del Contexto de la Aplicación.

En el primer proceso técnico mediante el cual el Equipo de Desarrollo adquiere un amplio concepto del contexto en donde se desarrolló la aplicación Softour, se consideró que el dominio de la aplicación estaba definido por el contexto donde operará el software que está delimitado por actividades de incentivo, promoción y apoyo al sector turismo.

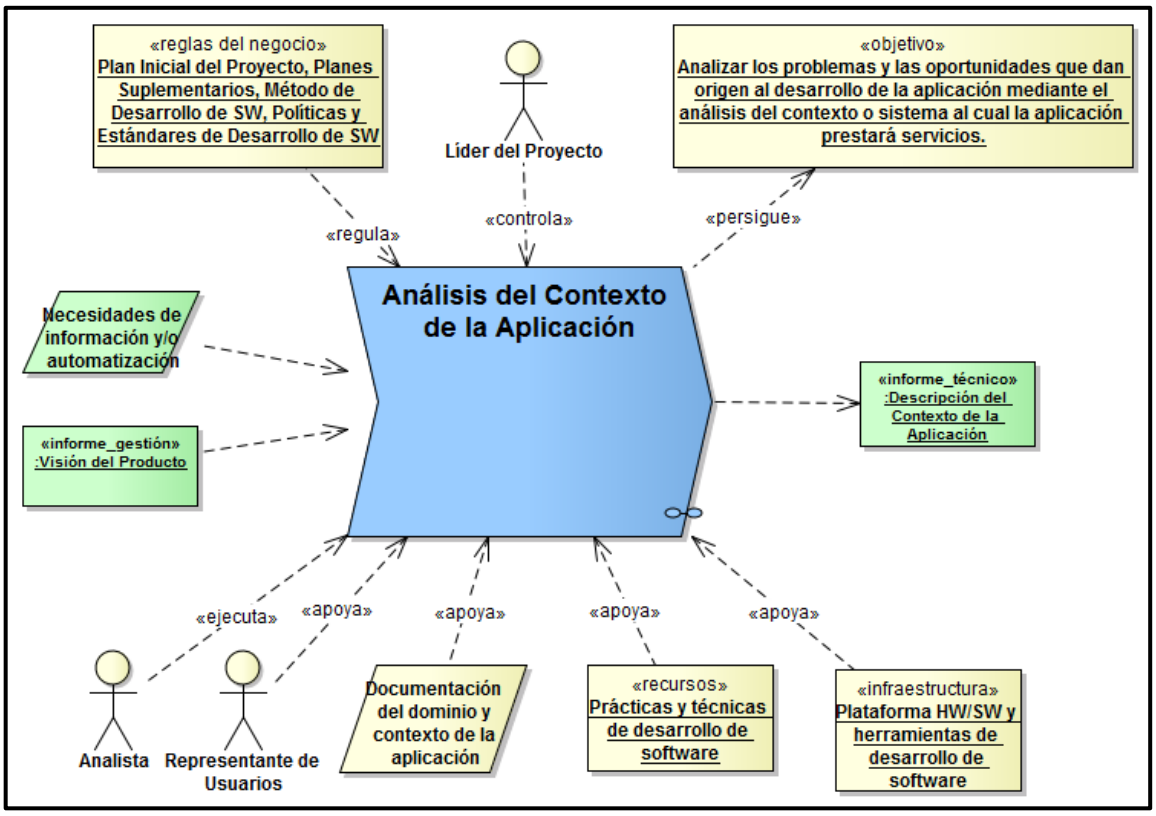


Figura 29. Diagrama de análisis del Contexto de la Aplicación.
Fuente: Montilva y Barrios, 2014.

Se desempeñaron diferentes roles en donde la líder del proyecto se encargó de controlar el desarrollo, el analista de ejecutarlo y los representantes de los usuarios finales (administrador, prestador de servicios y turista) de apoyar y aportar para realizar el análisis del contexto de la aplicación, para lo cual fue necesario evaluar la visión del producto, las necesidades de información y de automatización, apoyarse en prácticas, técnicas y herramientas que se adecuaron a las características del contexto del proyecto estudiado, considerando las competencias del grupo y las normativas internas establecidas para el desarrollo de dicho software.

El análisis del contexto de la aplicación tuvo como objetivo principal analizar los problemas y las oportunidades que dieron origen al desarrollo de la aplicación

mediante el análisis del contexto o sistema al cual la aplicación presta servicio.

3.1.3.1. Modelado de Negocios.

El modelado de negocio consistió en entender el dominio o contexto en donde se enmarca el problema, determinar las razones o motivaciones que impulsaron la investigación y facilitar la identificación de las necesidades de información requeridas para el desarrollo la aplicación Softour. Todos estos aspectos fueron representados utilizando el BMM, obteniendo así el modelado de objetivos, procesos, objetos, actores y roles, reglas y modelo de eventos.

3.1.3.2. Modelado de Objetivos.

Softour consideró los cambios y avances en cuanto a desarrollo de sitios web en el sector turismo, ya que los usuarios, para preparar su viaje, ya no sólo localizan información en buscadores y portales turísticos, ahora se dirigen a comunidades de webs sociales consultando información de los diferentes lugares, fotos, videos, sistemas de geolocalización, actividades y experiencias de otros usuarios que han visitado el destino de interés, la gastronomía que se ofrece y las posibilidades de diferentes tipos de alojamientos, comparando precios y evaluando el presupuesto disponible o necesario para disfrutar de un destino determinado, seleccionando la información que le interesa, con posibilidades de planificar el viaje; posteriormente los turistas se desplazan a su destino, viven su experiencia, realizan visitas, toman fotos, hacen grabaciones, desempeñan actividades, disfrutan de la gastronomía y de un confortable alojamiento; por ello continua con el ciclo de retroalimentación, ya que el turista comparte y califica, en la red, la información sobre su viaje y sobre sus experiencias, transformándose en un agente motivador, para que otros usuarios puedan consultarla.

Se identificó el problema que originó la necesidad de desarrollar la nueva

aplicación Softour, resaltando los siguientes aspectos: se estableció la web como plataforma de desarrollo considerando el uso de tecnología de punta; se manejaron diferentes vistas de software enfocadas a los diferentes perfiles de usuarios, como administrador, prestador de servicios y turista; cobraron mayor relevancia los servicios disponibles en la aplicación y la utilidad que brindaba el software al usuario por encima de los procedimientos de mantenimiento de la base de datos; se consideró que el software debe ser escalable, presentando habilidad para reaccionar y adaptarse sin perder calidad, por lo que se consideraron las buenas prácticas de la ingeniería de software para que al hacerse más grande no perdiera la funcionalidad en los servicios ofrecidos, mejorando a medida que más gente lo usa; se estimó que debía desarrollarse bajo una arquitectura de participación grupal, en donde las piezas, componentes e incrementos estarían debidamente unidos; se establecieron que las colecciones de datos debían brindar la posibilidad de actualización, transformación y mezcla; se subrayó la necesidad de la compatibilidad y la apertura hacia el uso de Softour en los diferentes tipos de dispositivos disponibles, ya sean computadores de escritorio, portátiles, tablets, teléfonos inteligentes y cualquier otro tipo de dispositivo informático; se consideró aprovechar los fundamentos de la inteligencia de negocios para obtener un mejor posicionamiento y mayor seguimiento del software.

Los paradigmas relacionados con el turista fueron considerados por Softour en aspectos como vender una actitud, un estilo de ver al mundo, ya que los turistas dejaron de ser simples destinatarios de la información para desempeñar el papel de productores o coproductores de la comunicación, acrecentándose la posibilidad de compartir y vivir nuevas experiencias, convirtiéndose en entes importantes en la creación de contenidos.

Los prestadores de servicios se visualizaron con el rol de incentivar, promocionar, publicitar y hacer competitivo un nuevo modelo de negocios en el sector turismo enfocado en la web, ya que pasó a ser otro usuario que

incrementa los contenidos y servicios ofrecidos en Softour.

El administrador fue enfocado como el responsable y encargado de la gestión final de la totalidad del software, en funciones de mantener el orden, validar la información que aportan los prestadores de servicios, otorgándole mayor libertad a los turistas, pero siempre considerando los valores y buenas costumbres para la sana convivencia en Softour.

Las oportunidades que el uso de la tecnología informática ofrece a los interesados para ser aprovechadas a través de la aplicación, desde el punto de vista de las empresas que administrarán el software, nos permitió interesarnos en aplicar las tecnologías necesarias para que los usuarios indiquen sus lugares favoritos, las actividades desarrolladas, las experiencias vividas, sus comentarios y sugerencias; además se requieren de procesos de gestión y control de la información, lo que implica nuevas visiones de desarrollo y personal para su mantenimiento.

Un aspecto importante que modificó nuestra visión de Softour fue el hecho negado de que el control de la información suministrada por los usuarios lo realizara la empresa que oferta los servicios, lo que hace que el usuario desconfíe de los datos que aparecen en la Web, ya que no posee ninguna garantía de imparcialidad, generando la necesidad de organizaciones independientes que sean objetivas y velen para que la información sea la proporcionada por la comunidad de usuarios virtuales, ya que si una empresa independiente recoge toda la información suministrada por los usuarios, la cataloga y la mantiene en una o varias bases de datos, se prestaría para manipulaciones ya que le concede la posibilidad de entrar en el negocio de la comercialización del turismo a su antojo, con la posibilidad de hacer prácticas poco éticas de redirigir a los usuarios a sitios asociados para que realicen la compra de los servicios que están buscando.

Es importante el incentivo y la participación de la comunidad de prestadores de

servicios, ya que son ellos los que proporcionan la oferta de lugares, actividades, experiencias, gastronomía y alojamientos que hacen llamativas las diferentes localidades, pueblos, ciudades y países, ya que incentivan el desarrollo económico de la región, generan fuentes de trabajo directa e indirectamente, promocionan y crean nuevos intermediarios en el modelo de negocios del sector turístico.

Desde el punto de vista del usuario turista, este debe tener la libertad de decidir qué contenidos publicar en una Web, pero una vez que están dentro del servicio, deberían mantenerse el control sobre la información aportada este, sin generar pérdidas de la misma; desde el mismo momento en que el usuario se registra con su correo electrónico, suben sus direcciones favoritas, sus fotografías personales, sus opiniones o sus vídeos, dejan al descubierto sus gustos y preferencias, los cuales pueden ser usados por las empresas o las instituciones del estado encargadas de la promoción y el fomento del turismo para coordinar el software especializado y hacer ofertas de servicios novedosas a los usuarios turistas, respetando su privacidad.

El dominio de una aplicación se definió como el contexto o sistema ampliado donde operará la aplicación desarrollada, el cual puede ser de dos tipos: el sistema de Negocios, que es un conjunto de procesos del negocio que persiguen objetivos predefinidos, emplean o involucran objetos del negocio, por ejemplo los recursos requeridos, los clientes turistas potenciales y los prestadores de servicios, que son regulados por reglas del negocio propias del sector turismo, son ejecutados por actores que se organizan en estructuras organizacionales; mientras que el Software del Sistema es un tipo particular de programas que pertenece a la plataforma de software de la empresa y que se caracterizan porque dan soporte a otras aplicaciones, por ejemplo, sistemas operativos, compiladores, middleware, utilitarios, entre otros.

Por ello fue importante analizar el contexto de la aplicación antes de definir y/o

especificar sus requisitos, ya que una aplicación resuelve problemas de información y aprovecha las oportunidades que la tecnología informática proporciona.

3.1.3.3. Modelado de Procesos de Negocios.

El modelo de procesos del negocio se comenzó con la elaboración de la cadena de valor, mostrando aquellos procesos que son la razón de ser de la aplicación en desarrollo, los cuales fueron clasificados en procesos fundamentales (PF) y procesos de apoyo (PA).

El diagrama de procesos representa los conceptos que delimitan la obtención de los propósitos u objetivos de la aplicación, existiendo varios objetivos principales del sistema para cada uno de los perfiles de usuario, para ello fue necesario describir el conjunto de pasos que se llevan a cabo durante estos procesos, así como también se destacaron los actores que controlan, ejecutan o apoyan dichos procesos. La Figura muestra la cadena de valor resultante.

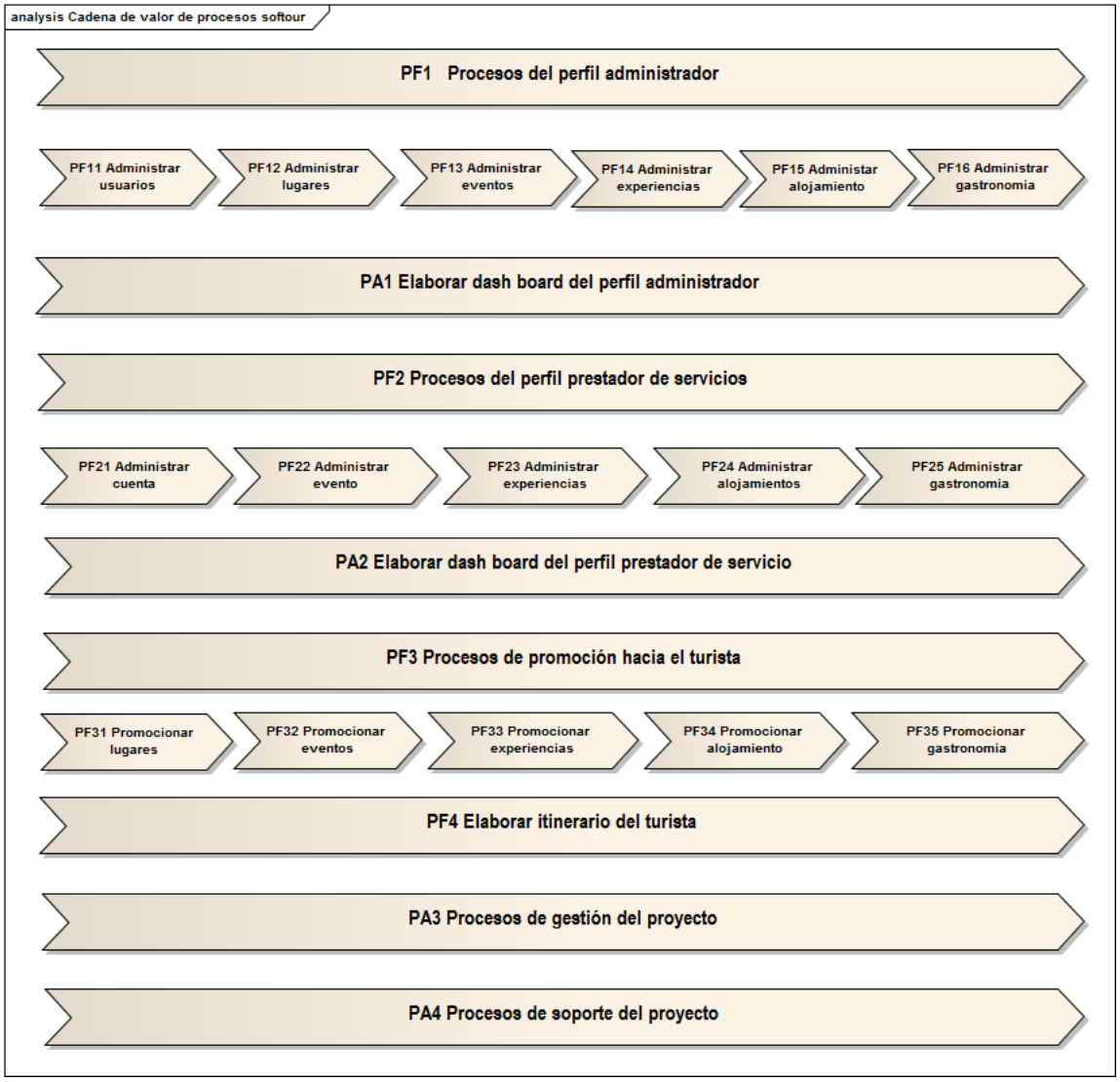


Figura 30. Cadena de valor de los procesos de Softour.

Los procesos de negocio se describen usando diagrama de procesos y actividades. En las siguientes imágenes se visualizan algunos de éstos. En el apéndice B se muestran el resto de los diagramas de procesos y subprocesos.

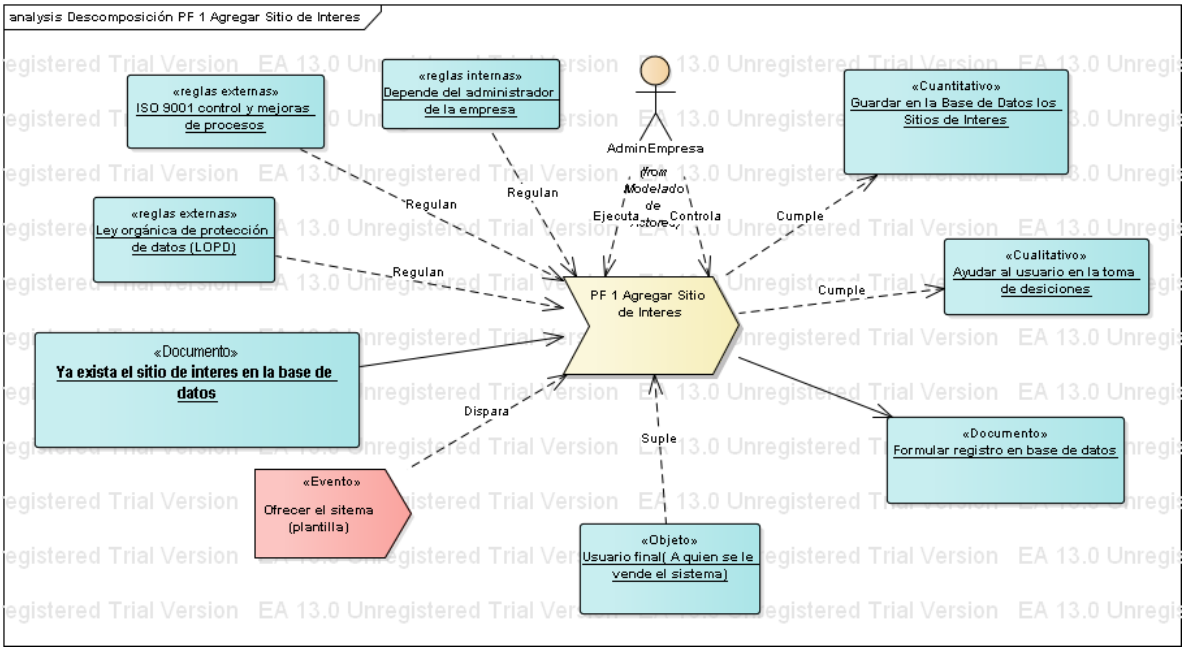


Figura 31. Diagrama del proceso Agregar Lugar.

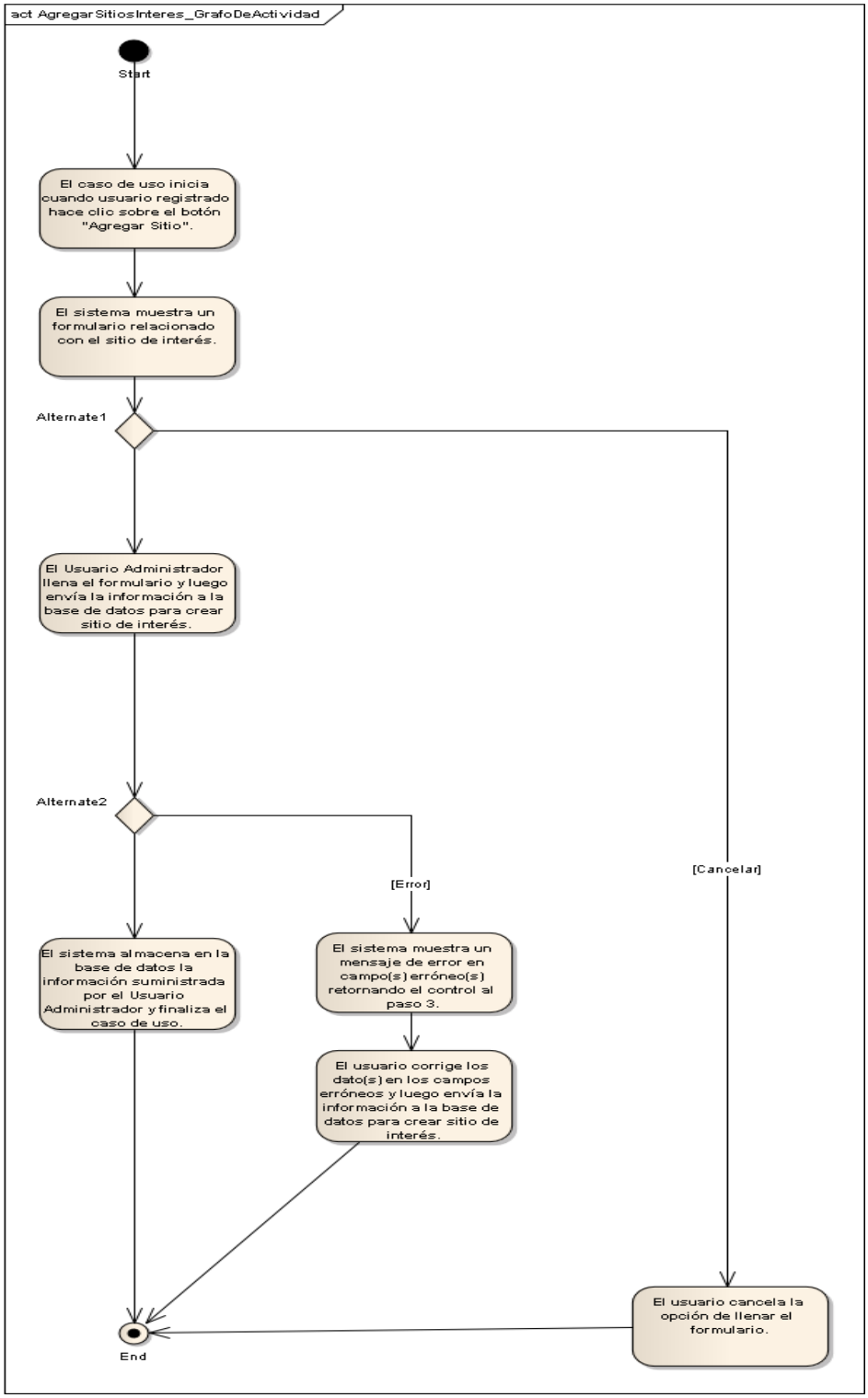


Figura 32. Diagrama de actividad del proceso Agregar Lugar.

3.1.3.4. Modelado de Objetos de Negocios.

El modelo de objetos de negocio facilitó la identificación de las colecciones de datos del modelo de negocio, las cuales vienen a representar las estructuras estáticas que son esenciales para el diseño de las clases y los diagramas de clases sobre el cual se basará el desarrollo orientado a objetos de la presente aplicación, motivado al uso de manejadores de base de datos no relacionales. Para esto se procedió a identificar los objetos del negocio a partir de la revisión de los diagramas de procesos, específicamente los representados en las entradas, salidas y otros elementos de apoyo a la ejecución de un proceso. Mediante la notación UML se representó cada objeto a través del diagrama inicial de clases de objetos.

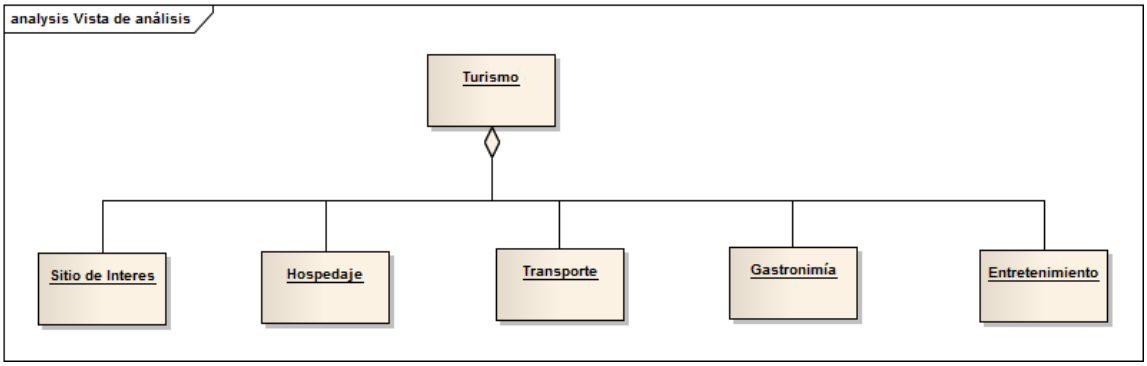


Figura 33. Diagrama inicial de clases de objeto.

3.1.3.5. Modelado de Reglas de Negocios.

A través del modelado de las reglas del negocio se logró representar la dinámica, reglamentos, normas, restricciones, limitaciones y alcances que rigen el funcionamiento de la aplicación Softour, que se muestran a continuación.

El usuario administrador tendrá acceso a todas las colecciones de datos para su administración, actualización y mantenimiento.

El usuario Prestador de Servicios tendrá acceso a la información relacionada

con su perfil y las colecciones de datos de las cuales es propietario, las cuales podrá administrar libremente.

El usuario turista tendrá a su disposición la posibilidad de consultar, seleccionar y agregar a su itinerario aquellos aspectos a los cuales tenga interés desde el punto de vista turístico, como lugares, eventos, experiencias, alojamientos y gastronomía.

El usuario turista podrá crear itinerarios para consolidar su viaje y estadía, una vez que se registra en Softour.

Los usuarios podrán registrar su perfil indicando un correo electrónico que lo identifique, sin establecer ningún otro requisito como login y password.

El usuario turista tendrá la capacidad de contactar a los prestadores de servicios para conocer en mayor profundidad las publicaciones de los mismos.

El usuario turista podrá interactuar con las funcionalidades del sistema para comentar y calificar las publicaciones experimentadas por el mismo.

Las características mínimas a mostrar en una publicación son las imágenes que la ilustren, el prestador de servicios asociado responsable, el sitio de encuentro en donde se desarrolla la publicación y la información que la describa en detalle.

La información de las publicaciones es registrada por el prestador de servicios que la ofrece, pero existe la libertad de que la realice el administrador del sistema a petición del prestador de servicios.

El usuario administrador del sistema es el único con la capacidad de manejar los procesos mantenimiento relacionados con la base de datos del sistema para ingresar, consultar, modificar y eliminar las clases o entidades de la misma. Denominados CRUD (créate, read, update, delete).

Los prestadores de servicios pueden presentar enfoques diferentes,

agrupándose en cuatro conceptos básicos iniciales: lugares, eventos, experiencias, alojamientos y gastronomía.

Los prestadores de servicios están autorizados a incorporar eventos propios, describiendo las características del mismo, indicando la fecha de finalización para desincorporar el evento del sistema.

Los lugares deben presentar enlaces asociados a los prestadores de servicios, brindando diferentes opciones. En el Apéndice I se muestra el diagrama de este modelo.

3.1.3.6. Modelado de Actores.

Mediante este modelado se identificó al conjunto de actores que se verán involucrados en el negocio, estipulando su nombre, sus funciones y roles.

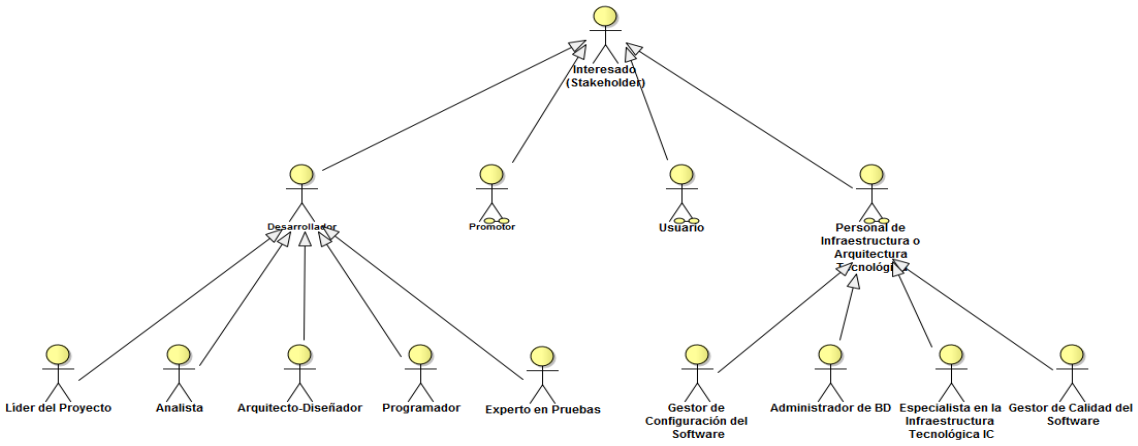


Figura 34. Diagrama del modelo de actores del método Blue Watch.
Fuente: Montilva y Barrios, 2014.

El diagrama de actores del método Blue Watch identifica y clasifica los roles (actores) que, el método recomendó utilizar durante el desarrollo de la aplicación Softour. Una misma persona puede ejercer a lo largo de un proyecto varios roles.

Cualquier persona, grupo u organización que participa en el desarrollo del software Softour es denominado interesado (stakeholder). Los interesados de un proyecto se dividen en tres categorías:

Los Promotores son personas o unidades organizacionales, por ejemplo, empresas o instituciones, que impulsan, apoyan, supervisan y/o requieren que la aplicación se desarrolle, como es el caso de la Gobernación del Estado Sucre, las diferentes alcaldías, los hoteles, las diferentes instituciones relacionadas con el desarrollo del turismo, los prestadores de servicios interesados en la aplicación. El Gerente del área funcional para la cual se desarrolla la aplicación y el Gerente de Desarrollo y Mantenimiento son dos de los principales promotores de cualquier proyecto de desarrollo de software, estos están representados por los Profesores responsables de las alternativas de grado de la Licenciatura en Informática del Núcleo de Sucre de la Universidad de Oriente.

Los Desarrolladores estuvieron representados por el personal técnico-gerencial que tiene a su cargo la responsabilidad de desarrollar la aplicación bajo la modalidad de proyecto. Se clasifican, de acuerdo a sus competencias, conocimientos, habilidades y destrezas en los siguientes: Líder del Proyecto, Analista, Arquitecto-Diseñador, Programador, Experto en Pruebas, Gestor de Soporte. Estos roles fueron desempeñados por los siete estudiantes cursantes de las alternativas de grado.

Los Usuarios son las personas, unidades u organizaciones que harán uso de la aplicación. Se dividen en: usuarios internos pertenecientes a la institución promotora; representante de usuarios representados por usuarios internos que tienen una participación activa en el proyecto de desarrollo de software, representa a los usuarios potenciales de la aplicación y actúa como enlace entre los usuarios internos y el Equipo de Desarrollo de la aplicación; y los usuarios externos que no pertenecen a la empresa pero que utilizan algunas de

sus funcionalidades, por ejemplo los turistas no registrados.

El Personal de Soporte Técnico Compartido, que está representado por la persona o grupo de personas que apoyan a uno o más Equipos de Desarrollo en actividades tales como: gestión de la configuración del software, gestión de la calidad, administración de bases de datos y soporte técnico relacionado con la plataforma de hardware, software y redes sobre la cual se instalan las aplicaciones. Este personal no forma parte del Equipo de Desarrollo; pero, le da soporte a este equipo cuando sea necesario. Los principales roles de soporte son: Gestor de la Configuración del Software, Administrador de Bases de Datos, Especialista en la Plataforma de Hardware, Software y Redes y, el Gestor de la Calidad del Software. Estos roles también fueron desempeñados por los cursantes de las alternativas de grado.

El número de roles que se requieren en un proyecto de desarrollo de software varía dependiendo del tamaño y la complejidad de la aplicación.

A continuación, se muestra una tabla en la que se identifica al actor o rol de Turista con las funciones y actividades que desempeña en cada una de ellas.

Tabla 4. Especificación del actor Turista, roles y actividades que desempeña.

Actor	Función	Actividades
Turista	Buscar, consultar y seleccionar	Buscar, consultar y seleccionar lugares, eventos, experiencias, alojamientos y gastronomía que se encuentre relacionado al área o región que desea conocer.
	Registrarse	El turista podrá registrarse en el sistema permitiéndole obtener funciones adicionales como crear itinerarios.
	Contactar Prestadores de Servicios.	El turista podrá contactar al Prestador de Servicios para obtener más información que la mostrada en el sistema.
	Comentar	El turista podrá comentar sus experiencias sobre lugares, eventos, experiencias, alojamientos y gastronomía mostrados en el sistema.
	Calificar	El turista podrá calificar los diferentes lugares, eventos, experiencias, alojamientos y gastronomía que se encuentre en el sistema de acuerdo a su experiencia. Se debe validar que dicha calificación sea cierta.
	Gestionar Perfil	El turista tendrá la capacidad de gestionar su perfil de usuario para estar identificado dentro del sistema. Este podrá darse de alta, consultar y actualizar sus datos.

Tabla 5. Especificación del actor Prestador de Servicios, sus funciones y actividades que desempeña.

Actor	Función	Actividades
Prestador de Servicios	Responder contacto	Responde al contacto realizado por los turistas, referente a la actividad que promociona.
	Crear, consultar, modificar, eliminar lugares.	Crea, consulta, modifica y elimina los lugares asociados a sus publicaciones como: eventos, experiencias, alojamientos y gastronomía.
	Crear, consultar, modificar, eliminar eventos.	Crea nuevos eventos que desea promocionar o consulta, modifica y elimina aquellos eventos que ya promociona.
	Crear, consultar, modificar, eliminar experiencias.	Crea nuevas experiencias que desea promocionar o consulta, modifica y elimina aquellas experiencias que ya promociona.
	Crear, consultar, modificar, eliminar alojamientos.	Crea un nuevo alojamiento que desea promocionar o consulta, modifica y elimina aquellos alojamientos que ya promociona.
	Crear, consultar, modificar, eliminar gastronomía.	Crea nueva gastronomía que desea promocionar o consulta, modifica y elimina aquella gastronomía que ya promociona.
	Gestionar Perfil	El prestador de servicios tendrá la capacidad de gestionar su perfil de usuario para estar identificado dentro del sistema. Este podrá darse de alta, consultar y actualizar sus datos.

Tabla 6. Especificación del actor Prestador de Servicios, sus funciones y actividades que desempeña.

Actor	Función	Actividades
Administrador	Responder contacto	Responde al contacto realizado por los prestadores de servicios y turistas, referente al uso y administración de la aplicación.
	Crear, consultar, modificar, eliminar usuarios.	Crea, consulta, modifica y elimina los usuarios en cualquiera de los diferentes 03 tipos de perfiles: administrador, prestador de servicios y turista.
	Crear, consultar, modificar, eliminar lugares.	Crea nuevos lugares que se desean promocionar o consulta, modifica y elimina aquellos lugares que ya se promocionan.
	Crear, consultar, modificar, eliminar eventos.	Crea nuevas experiencias que desea promocionar algún prestador de servicios o consulta, modifica y elimina aquellas experiencias que ya promociona algún prestador de servicios.
	Crear, consultar, modificar, eliminar experiencias.	Crea nuevas experiencias que desea promocionar algún prestador de servicios o consulta, modifica y elimina aquellas experiencias que ya promociona algún prestador de servicios.
	Crear, consultar, modificar, eliminar alojamientos.	Crea nuevos alojamientos que desea promocionar algún prestador de servicios o consulta, modifica y elimina aquellos alojamientos que ya promociona algún prestador de servicios.
	Crear, consultar, modificar, eliminar gastronomía.	Crea nuevas publicaciones relacionadas con gastronomía que desea promocionar algún prestador de servicios o consulta, modifica y elimina aquella gastronomía que ya promociona algún prestador de servicios.

Basados en UML los actores quedaron representados de la siguiente manera:

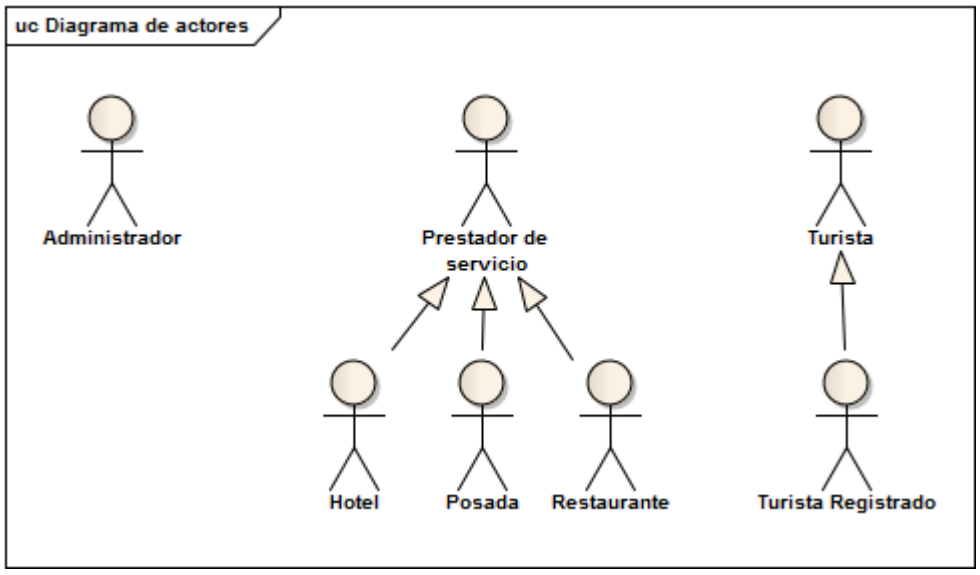


Figura 35. Diagrama de Actores.

3.1.4. Desarrollo Inicial de Requisitos.

El Desarrollo Inicial de Requisitos es un conjunto de procesos técnicos que definieron y/o especificaron las características de la aplicación, incluyendo sus requisitos funcionales y no-funcionales. Su producto, denominado Documento de Requisitos, fue un informe técnico que se elaboró gradualmente a lo largo de todo el proyecto. Su primera versión se elaboró durante este proceso técnico y se refinó durante los ciclos o iteraciones del proyecto. Este proceso se divide en dos sub-procesos relacionados, cuya ejecución depende del tamaño y tipo de aplicación.

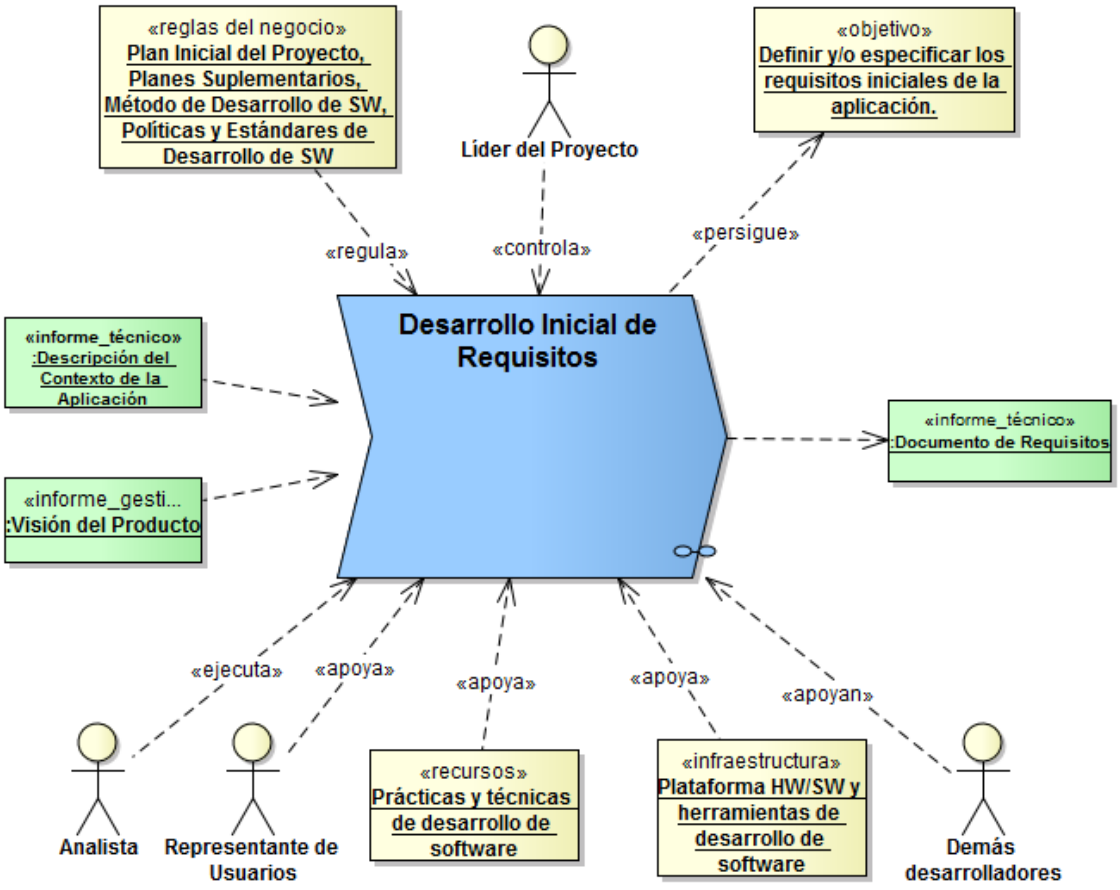


Figura 36. Diagrama de desarrollo inicial de requisitos.
Fuente: Montilva y Barrios, 2014.

La Definición Inicial de Requisitos, que produjo un informe del mismo nombre en donde se describió informalmente, a través de historias de usuarios, los requisitos iniciales de los interesados.

La Especificación Inicial de Requisitos, que especifica de una manera más formal, mediante modelos, los requisitos que fueron establecidos en la Definición Inicial de Requisitos.

El Desarrollo Inicial de Requisitos fue controlado por la líder del proyecto, ejecutado por los analistas y apoyado por el resto de los desarrolladores y por los representantes de los diferentes tipos de usuarios; dicho proceso estuvo apoyado en la plataforma de hardware y software seleccionada para utilizar

herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

El objetivo principal obtenido fue la definición y/o especificación de los requisitos iniciales de la aplicación.

El desarrollo de requisitos fue un proceso fundamental para plasmar los requerimientos que la aplicación debe cumplir para satisfacer las necesidades de los diferentes tipos de usuarios, en donde el equipo de trabajo sostuvo varias reuniones en las que realizó análisis donde se determinó las funciones que debía contener la aplicación. Este proceso está conformado por los subprocesos de descubrimiento, análisis, especificación y validación de requisitos; todos estos descritos a continuación.

3.1.4.1. Descubrimiento de Requisitos.

Los requisitos fueron propuestos por el equipo de trabajo mediante reuniones en las cuales se analizaron posibles sistemas dirigidos al mismo sector y escuchando recomendaciones de usuarios, los cuales expresaban los virtudes y ventajas que poseían dichos sistemas. A continuación, se plantean los requisitos descubiertos mediante la siguiente tabla:

Tabla 7. Lista de Requisitos.

ID	REQUISITOS
01	Agregar, consultar, modificar y eliminar lugar. CRUD de lugar.
02	Agregar, consultar, modificar y eliminar evento. CRUD de evento.
03	Agregar, consultar, modificar y eliminar experiencia. CRUD de experiencia.
04	Agregar, consultar, modificar y eliminar alojamiento. CRUD de alojamiento.
05	Agregar, consultar, modificar y eliminar gastronomía. CRUD de gastronomía.
06	Agregar, consultar, modificar y eliminar usuario. CRUD de usuario.
07	Realizar comentario sobre lugar, evento, experiencia, alojamiento, gastronomía, turista y prestador de servicios.
08	Generar y mostrar indicadores estadísticas o Dashboard.
09	Realizar búsquedas avanzadas a través de filtros.
10	Autoajustar formato, texto, imagen y cualquier otro componente del sistema de acuerdo al dispositivo de navegación.
11	Establecer una interfaz con características de usabilidad orientada a los servicios turísticos.
12	Desarrollar la aplicación en lenguaje javascript con Node.js y Express.js.
13	Desarrollar la aplicación utilizando el manejador de colecciones de datos (base de datos) mongoDB.
14	Realizar un sistema multiplataforma orientado a la web.
15	Soportar carga de imágenes tanto por el administrador como por el prestador de servicios.
17	Realizar la validación de los datos a nivel de interfaz y a nivel de base de datos del sistema
18	Utilizar nombres, etiquetas, identificadores para búsquedas y asociaciones de las colecciones de datos.

3.1.4.2. Análisis de Requisitos.

A continuación, se muestra la clasificación de requisitos en funcionales y no funcionales:

Tabla 8. Clasificación de requisitos funcionales y no funcionales.

ID	REQUISITOS	TIPO
01	Agregar, consultar, modificar y eliminar lugar. CRUD de lugar.	Funcional
02	Agregar, consultar, modificar y eliminar evento. CRUD de evento.	Funcional
03	Agregar, consultar, modificar y eliminar experiencia. CRUD de experiencia.	Funcional
04	Agregar, consultar, modificar y eliminar alojamiento. CRUD de alojamiento.	Funcional
05	Agregar, consultar, modificar y eliminar gastronomía. CRUD de gastronomía.	Funcional
06	Agregar, consultar, modificar y eliminar usuario. CRUD de usuario.	Funcional
07	Realizar comentario sobre lugar, evento, experiencia, alojamiento, gastronomía, turista y prestador de servicios.	Funcional
08	Generar y mostrar indicadores estadísticas o Dashboard.	Funcional
09	Realizar búsquedas avanzadas a través de filtros.	Funcional
10	Autoajustar formato, texto, imagen y cualquier otro componente del sistema de acuerdo al dispositivo de navegación.	No Funcional
11	Establecer una interfaz con características de usabilidad orientada a los servicios turísticos.	No Funcional
12	Desarrollar la aplicación en lenguaje javascript con Node.js y Express.js.	No Funcional
13	Desarrollar la aplicación utilizando el manejador de colecciones de datos (base de datos) mongoDB.	No Funcional
14	Realizar un sistema multiplataforma orientado a la web.	No Funcional
15	Soportar carga de imágenes tanto por el administrador como por el prestador de servicios.	No Funcional
17	Realizar la validación de los datos a nivel de interfaz y a nivel de base de datos del sistema	Funcional
18	Utilizar nombres, etiquetas, identificadores para búsquedas y asociaciones de las colecciones de datos.	No Funcional

Luego de clasificar los requisitos se procedió a realizar el diagrama de casos de uso y el de clases. Las siguientes figuras muestras el diagrama de casos de

usos general y el diagrama de clases.

3.1.5. Diseño Inicial de la Aplicación.

El proceso de Diseño Inicial de la Aplicación es un conjunto de procesos técnicos que describieron la arquitectura de la aplicación y el diseño inicial de cada uno de los componentes que integraron dicha arquitectura.

La descripción de la arquitectura se llevó a cabo de manera gradual a lo largo del proyecto. Se inició en este proceso y se mejoró durante cada ciclo o iteración del proceso de Desarrollo de Incrementos.

El Diseño Inicial de la Aplicación fue controlado por la líder del proyecto, ejecutado por los arquitectos y diseñadores, y apoyado por los analistas, los demás desarrolladores, los representantes de los diferentes tipos de usuarios y por el personal de infraestructura o arquitectura tecnológica.

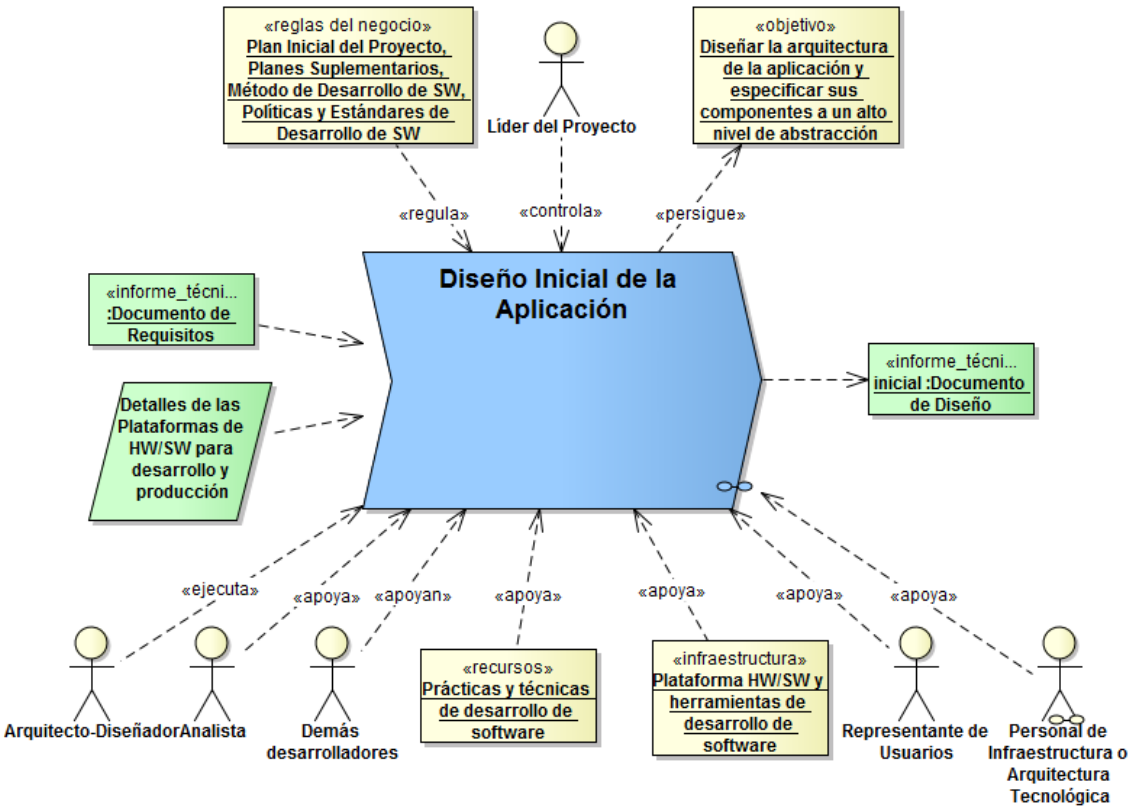


Figura 37. Diagrama de diseño inicial de la aplicación.
Fuente: Montilva y Barrios, 2014.

Dicho proceso estuvo apoyado en la plataforma de hardware y software seleccionada para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

El objetivo principal obtenido fue el diseño de la arquitectura de la aplicación y la especificación de sus componentes a un alto nivel de abstracción.

La Planificación de Entregas es un conjunto de actividades de Gestión del Proyecto que tuvo como objetivos la división de la aplicación en incrementos entregables y la planificación de la entrega de dichos incrementos. Primero se hizo la división de la arquitectura de la aplicación en incrementos entregables, donde un incremento entregable fue producido en una o más iteraciones y

constituyó una entrega parcial y operativa de la aplicación. Luego se distribuyeron los requisitos establecidos en la Pila de Historias del Proyecto entre los incrementos establecidos, en función de las prioridades y las relaciones que existían entre estos requisitos.

El diseño arquitectónico se descompone en subprocesos denominados vistas, las cuales especifican con diagramas el funcionamiento del sistema. A continuación, se mencionan las vistas y el producto de cada una: la vista funcional esta se determina mediante el diagrama de casos de uso; la vista estructural representada por el diagrama de clases; la vista de comportamiento se visualiza con los diagramas de secuencia; la vista de implementación esta soportada por diagramas de componentes y; por último, la vista de despliegue presentada por diagramas que llevan su mismo nombre, diagramas de despliegue.

3.1.5.1. Vista Funcional.

La vista funcional muestra el comportamiento del sistema, este consta de un conjunto de diagramas de casos de usos organizados de acuerdo a los requisitos mencionados con anterioridad. A continuación, se muestra los Diagramas de Casos de Uso generales del sistema.

La Figura 38 muestra la interacción del usuario Administrador en el sistema Sofour, relacionándose con los casos de uso administrar lugares, eventos, experiencias, alojamientos, gastronomías y usuarios.

La Figura 39 muestra la expansión del Caso de Uso Administrar Lugares del usuario Administrador en el sistema Softour, relacionándose con los casos de uso ingresar lugares y administrar sitios de interés, los que a su vez extienden funcionalidades hacia otros casos de uso como consultar, modificar, eliminar lugares.

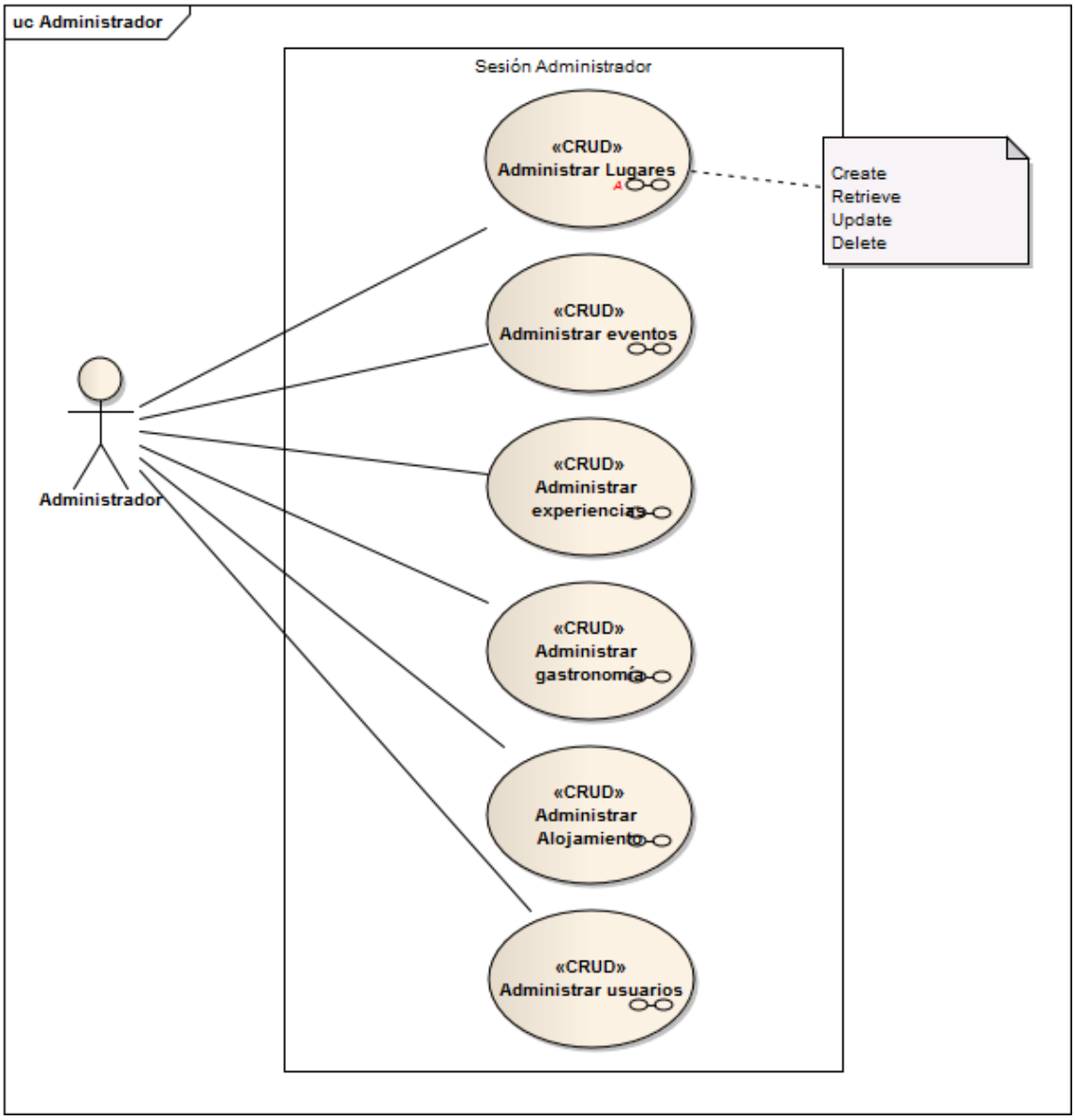


Figura 38. Casos de Uso general del perfil Administrador.

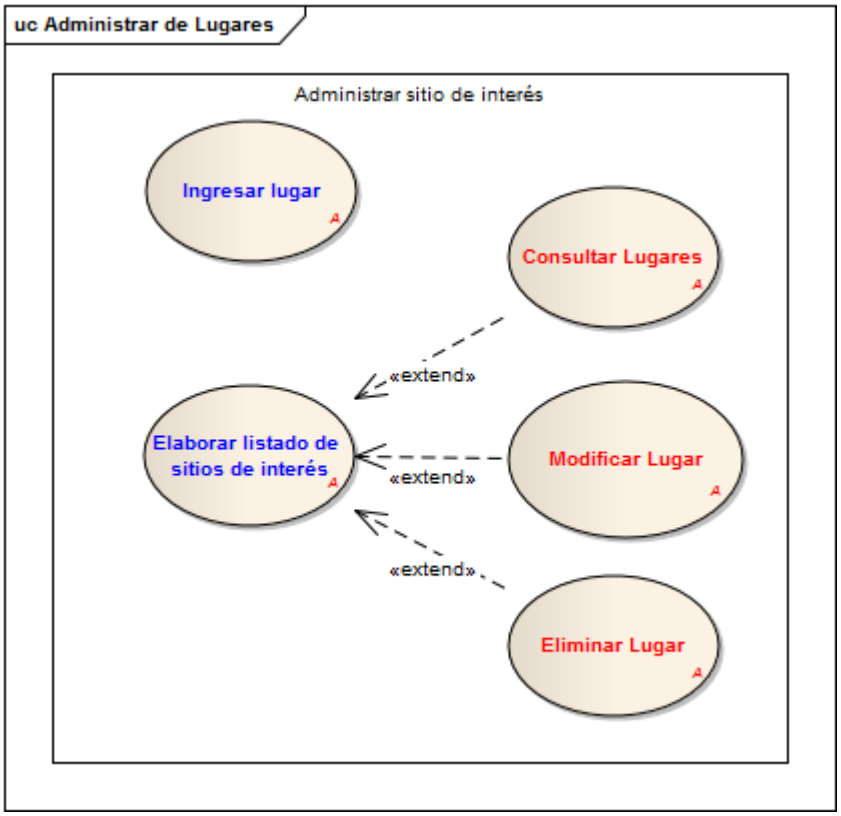


Figura 39. Caso de Uso Administrar Lugares. CRUD Lugares.

La Figura 40 muestra la interacción del usuario Prestador de Servicios en el sistema Softour, relacionándose con los casos de uso reducido de administrar servicios del perfil prestador de servicios con respecto a experiencias y lugares.

La Figura 41 muestra la expansión del Caso de Uso Administrar Experiencias del usuario Prestador de Servicios en el sistema Softour, relacionándose con los casos de uso ingresar lugares y administrar sitios de interés, los que a su vez extienden funcionalidades hacia otros casos de uso como consultar, modificar, eliminar lugares.

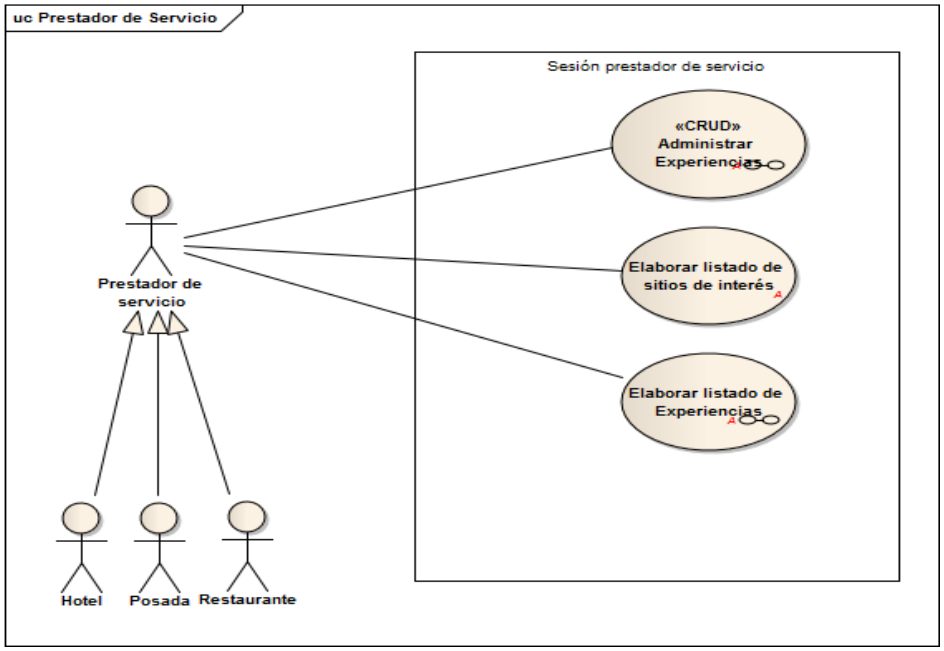


Figura 40. Casos de Uso reducido del perfil Prestador de Servicios.

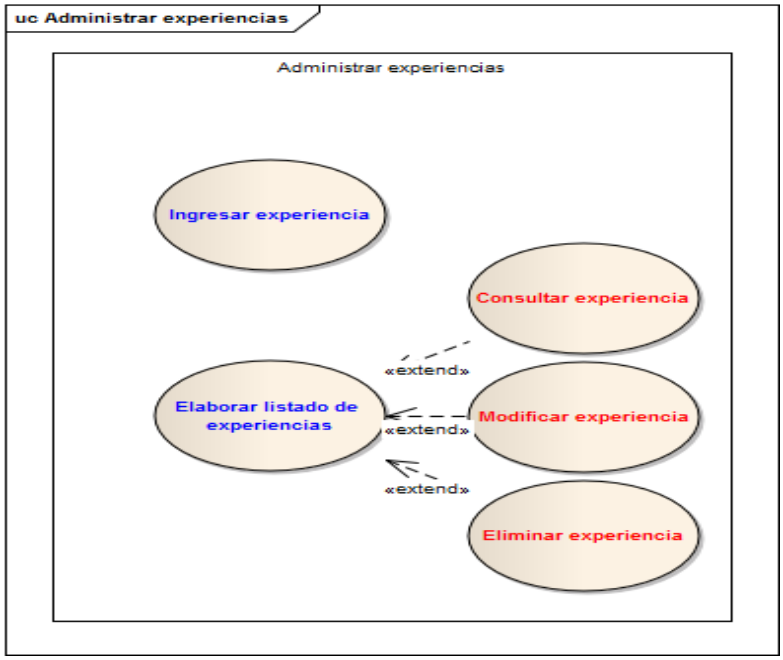


Figura 41.Caso de Uso Administrar Lugares. CRUD Lugares.

3.1.5.2. Vista Estructural.

La vista estructural es representada mediante los diagramas de clases que en este caso se presentan por iteraciones asociadas a los casos de uso, en donde se interrelacionan las clases de persistencias, control e interfaz. Muestra el comportamiento del sistema de acuerdo a las clases definidas y la conexión o comunicación que deben poseer entre sí, especificando las clases que le darán soporte a cada subproceso del sistema de acuerdo a su estructura, atributos, operaciones y relaciones.

En la Figura 43 se observa las clase de interfaz “AgregarLugar” que recibe los requerimientos del usuario externo y presenta asociación con la clase de control “ControladorPrincipal” que recibe las peticiones de la interfaz y las administra en el servidor para asignar la tarea a la clase de control especialista “ControladorLugar” que gestiona las operaciones de la clase persistente “Lugar”, la cual se asocia con la tabla o colección de la base de datos orientada a documentos “Lugar” que requirió de la gestión de la clase o paquete manejador de documentos “Mongoose”, este último se encargó de aportar librerías que permitieron asociar una base de datos no relacional, no orientada a objetos, con el modelo de negocios orientado a objetos plasmado en Softour.

En la Figura 44 se observa las clase de interfaz “ConsultarLugar” que recibe los requerimientos del usuario externo y presenta asociación con la clase de control “ControladorPrincipal” que recibe las peticiones de la interfaz y las administra en el servidor para asignar la tarea a la clase de control especialista “ControladorLugar” que gestiona las operaciones de la clase persistente “Lugar”, la cual se asocia con la tabla o colección de la base de datos orientada a documentos denominada “Lugar” que requirió de la gestión de la clase o paquete manejador de documentos “Mongoose”, este último se encargó de brindar librerías que permitieron asociar una base de datos no relacional, no orientada a objetos con el modelo de negocios orientado a objetos plasmado en

Softour.

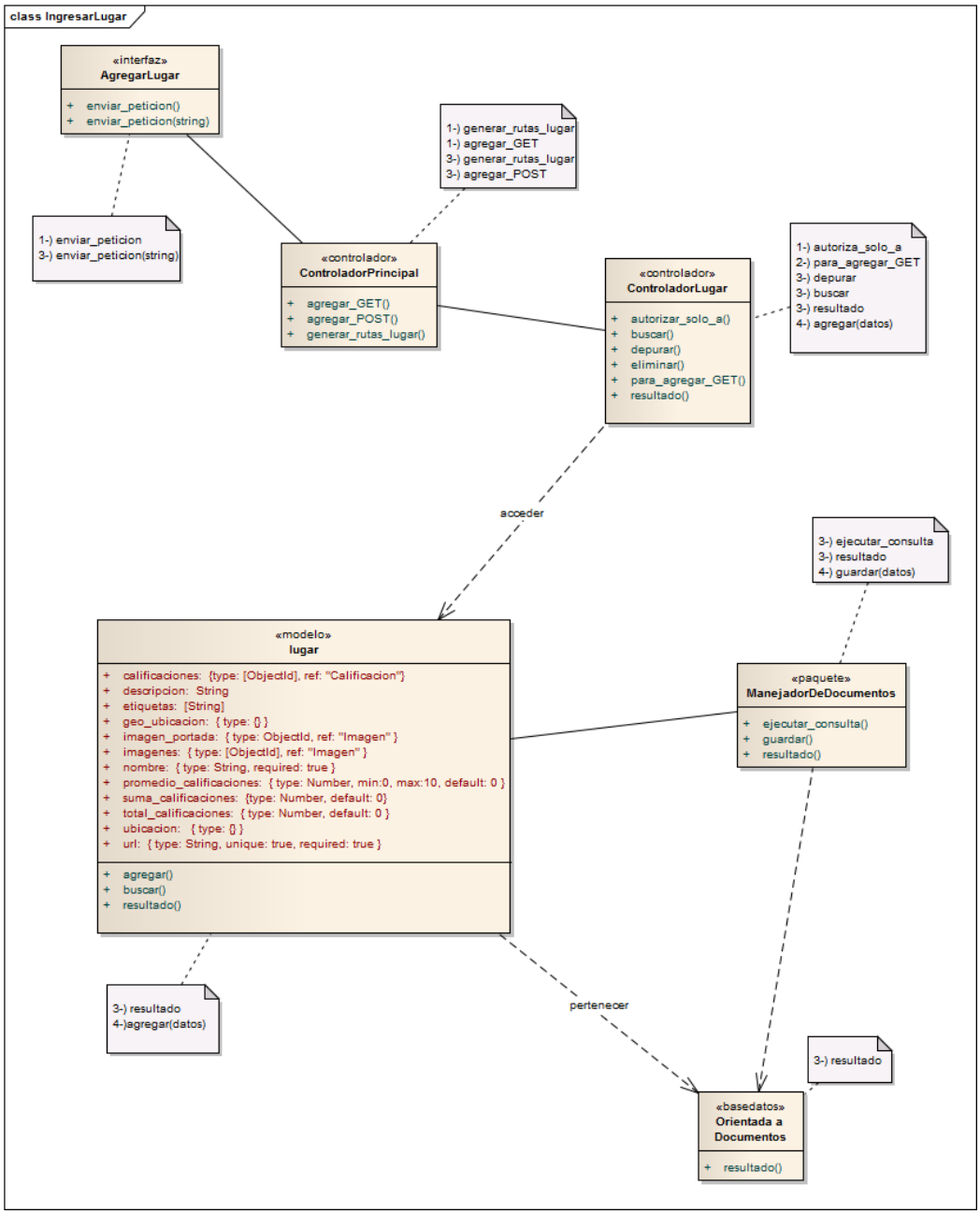


Figura 42. Diagrama de Clases asociada al Caso de Uso Ingresar Lugar.

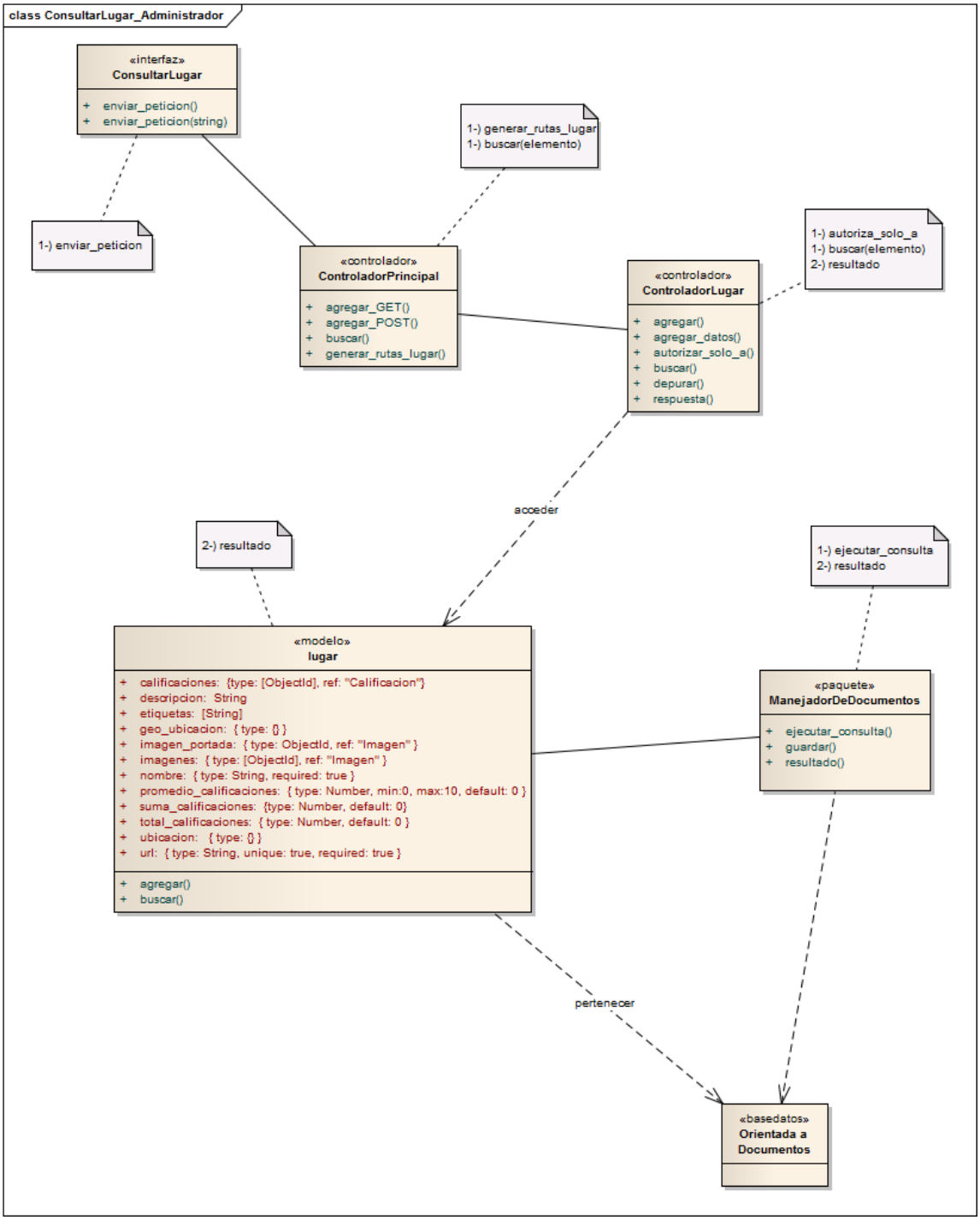


Figura 43. Diagrama de Clases asociada al Caso de Uso Consultar Lugar.

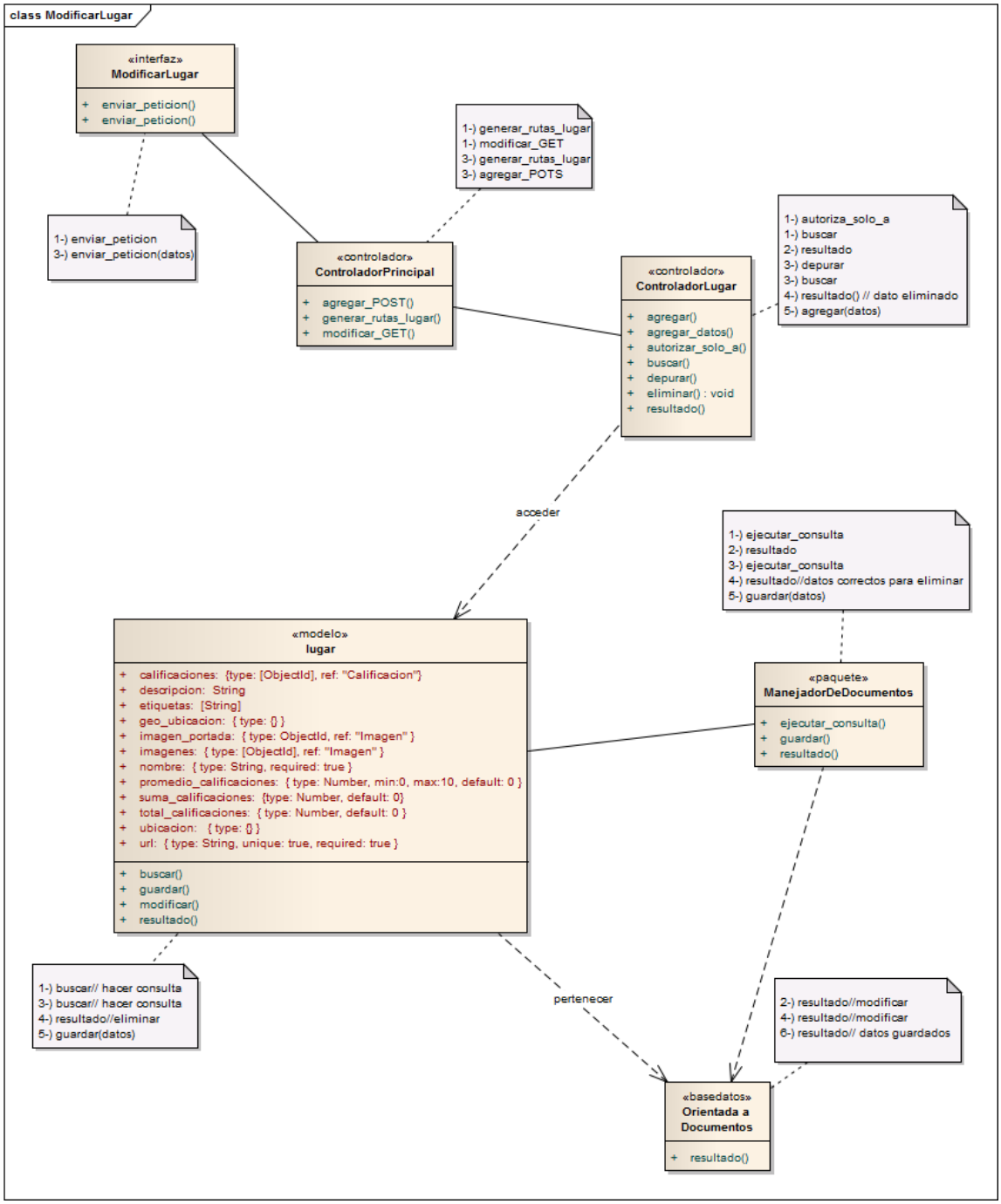


Figura 44. Diagrama de Clases asociada al Caso de Uso Modificar Lugar.

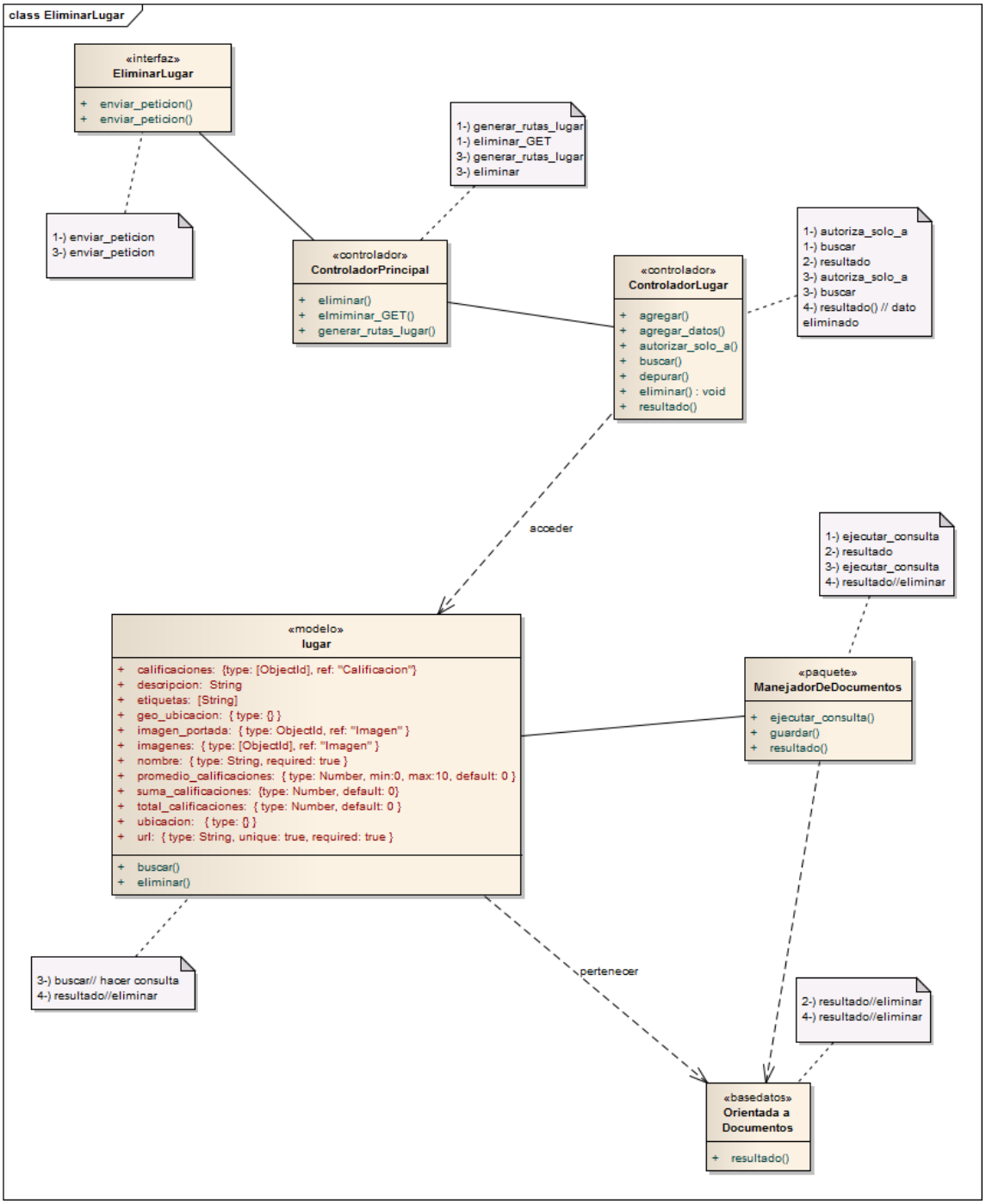


Figura 45. Diagrama de Clases asociada al Caso de Uso Eliminar Lugar.

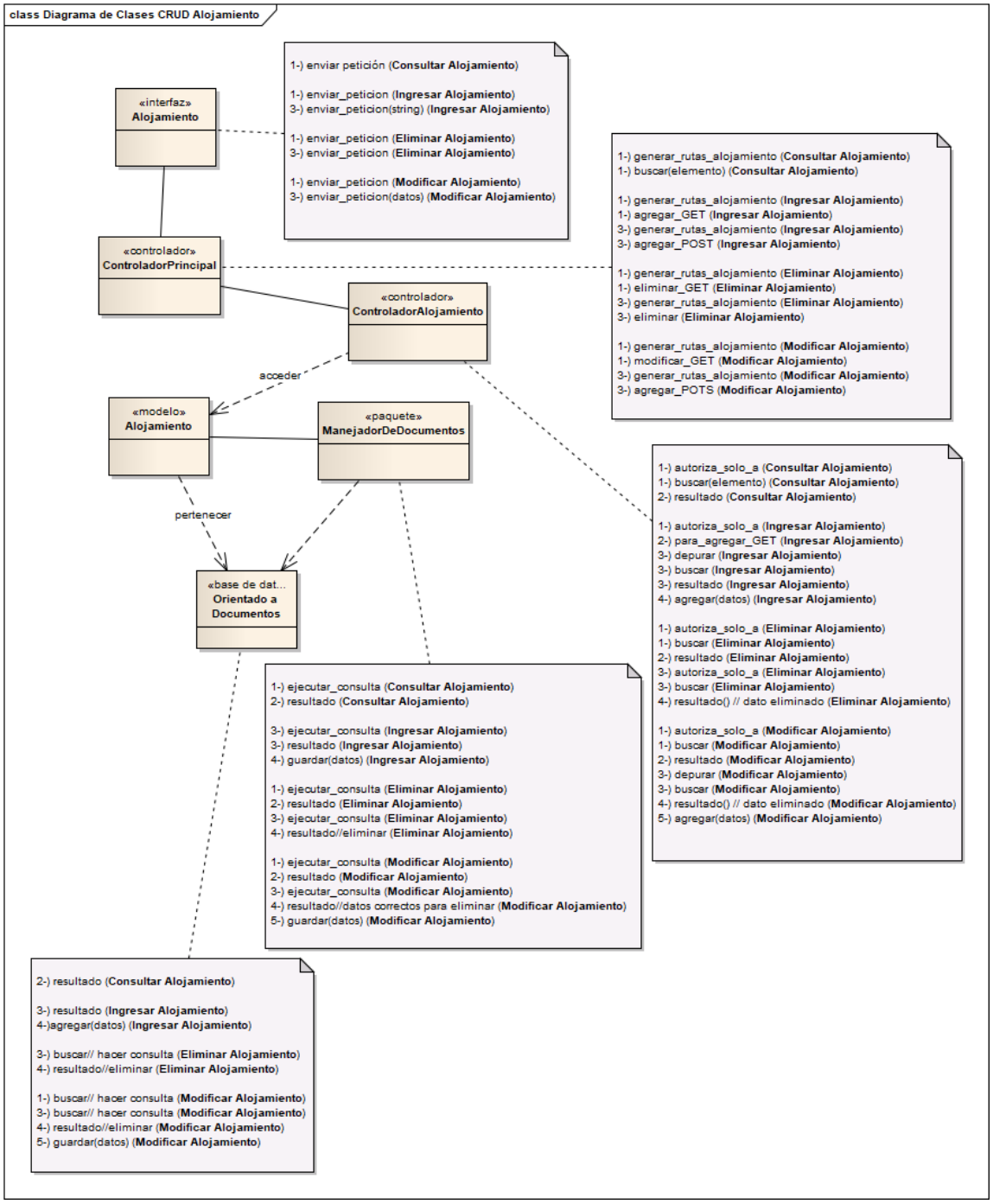


Figura 46. Diagrama de Clases del Módulo Alojamiento del Sistema Softour.

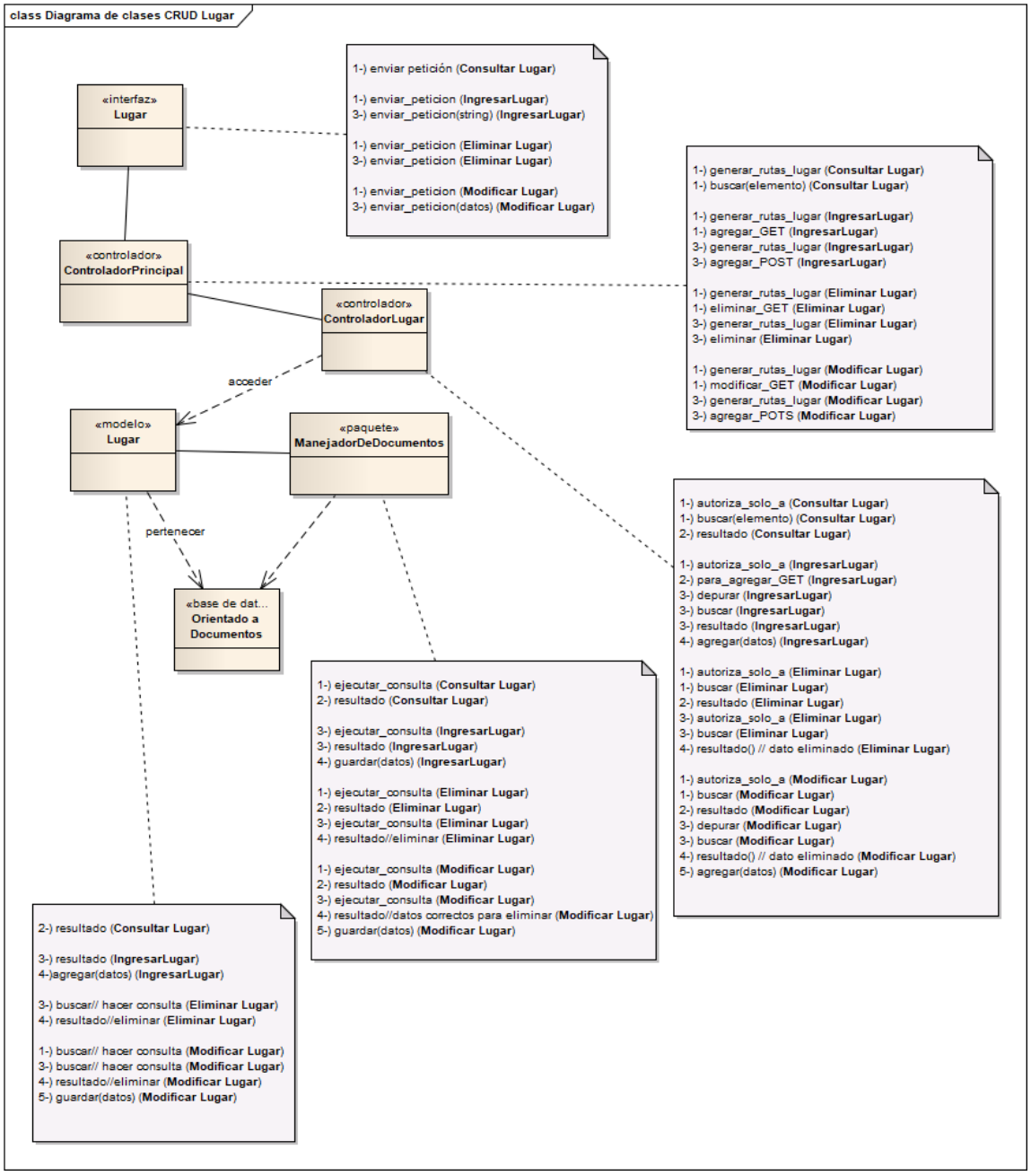


Figura 47. Diagrama de Clases del Módulo Consultar Lugar del Sistema Softour.

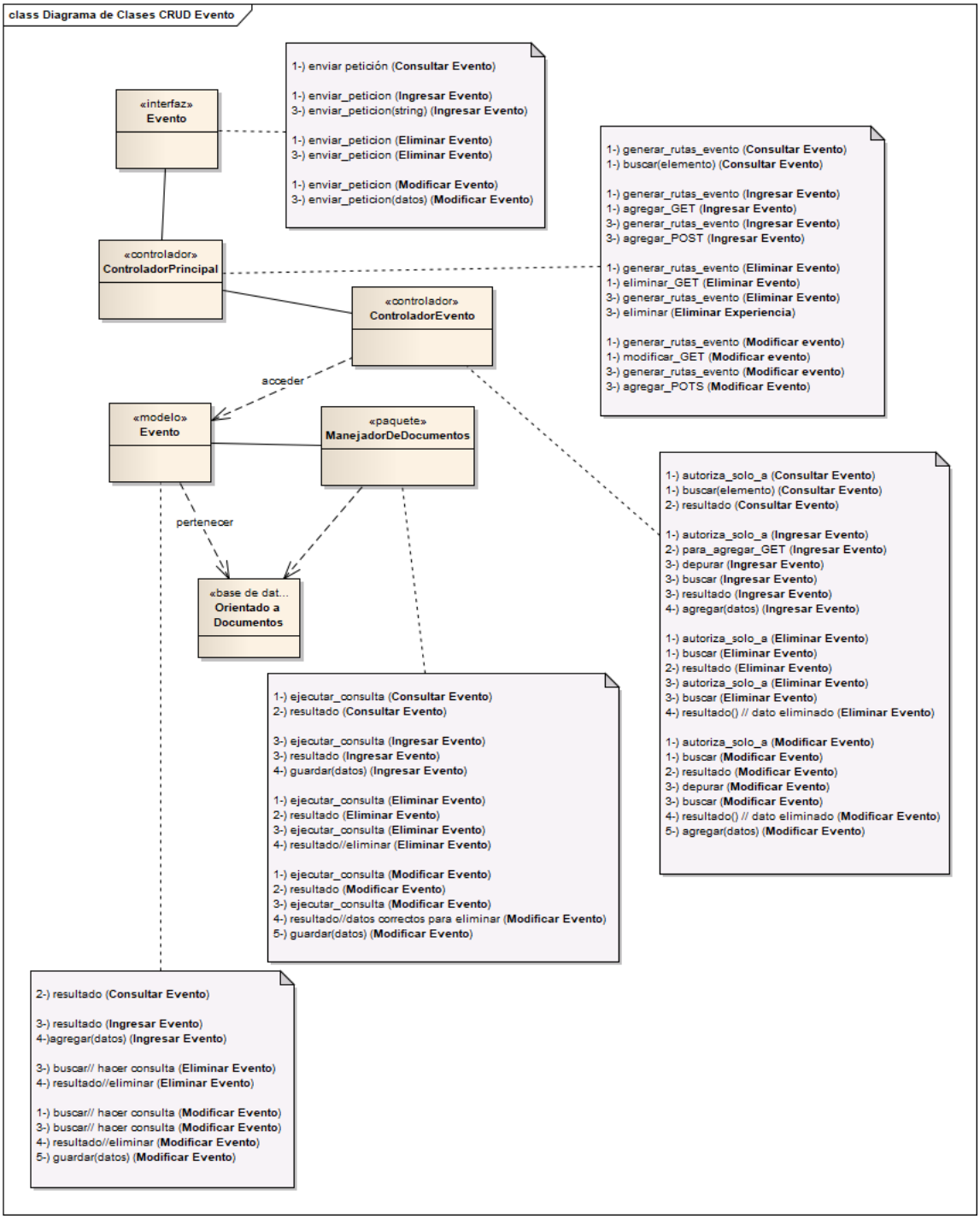


Figura 48. Diagrama de Clases del Módulo Consultar Evento del Sistema Softour.

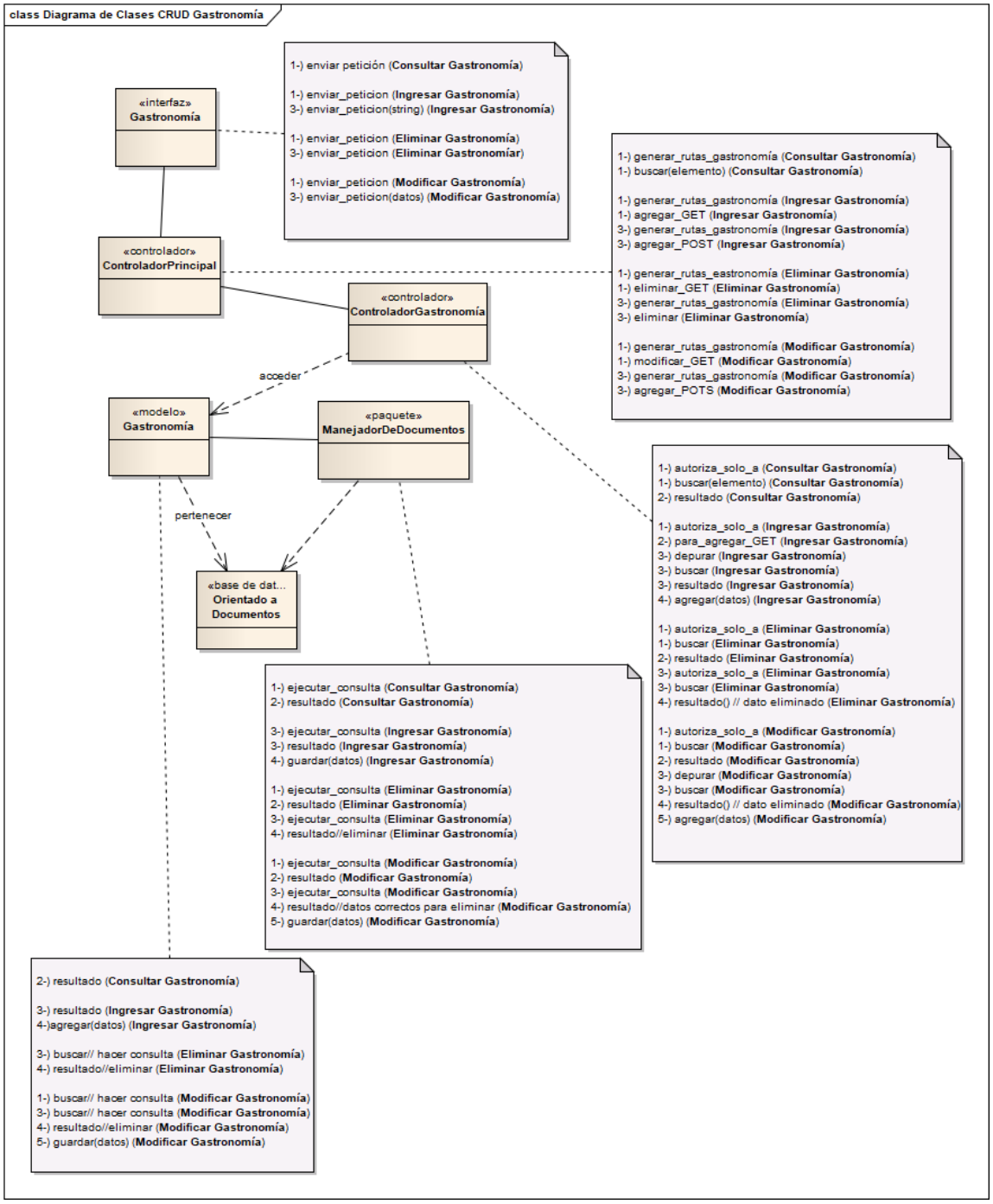


Figura 49. Diagrama de Clases del Módulo Consultar Gastronomía del Sistema Softour.

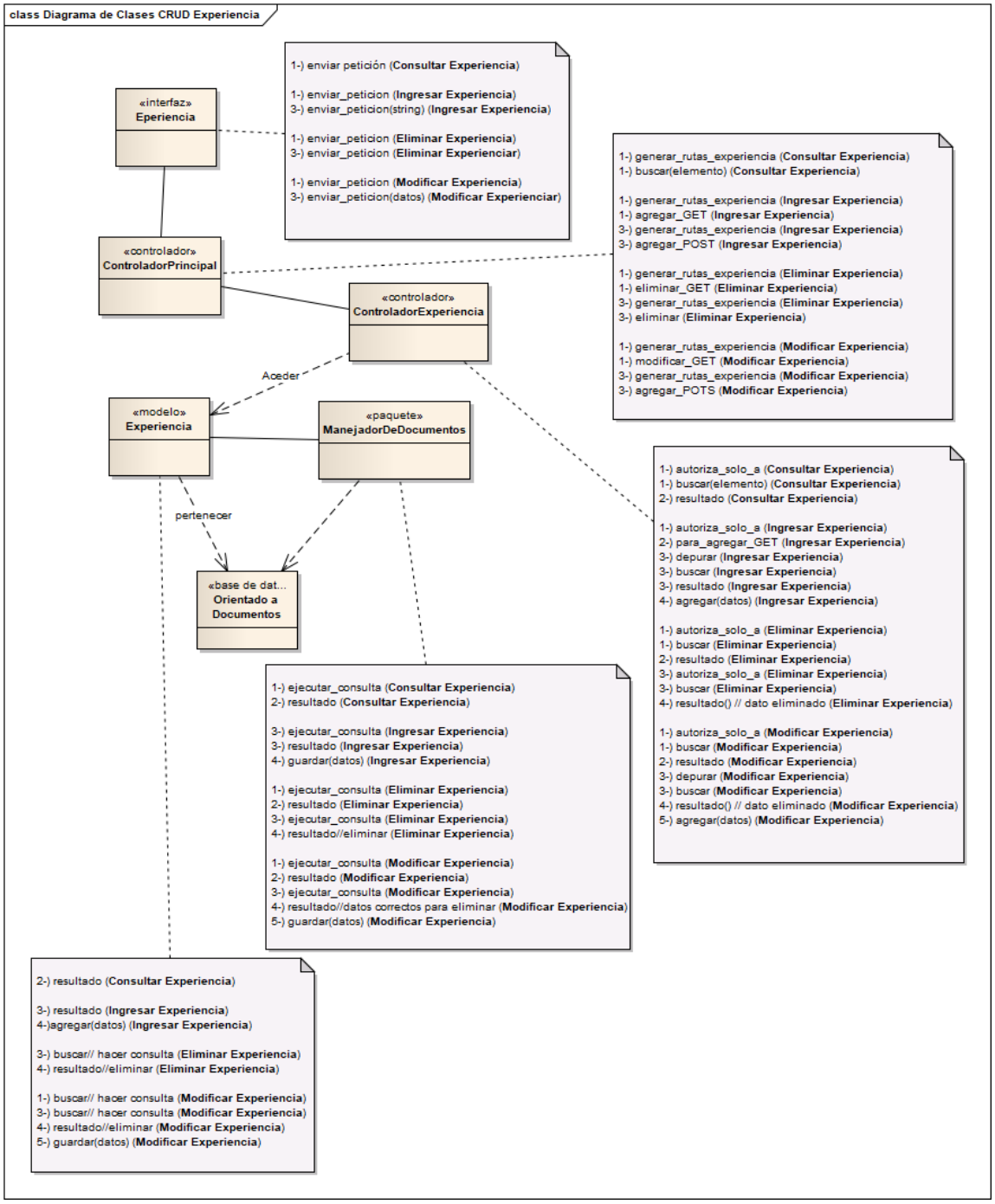


Figura 50. Diagrama de Clases del Módulo Consultar Experiencia del Sistema Softour.

3.1.5.3. Vista de Comportamiento.

Esta vista está compuesta por los diagramas de secuencias que muestran la interacción de un conjunto de objetos de la aplicación Softour a través del tiempo y los modela para cada caso de uso, fue útil para complementar al diagrama de clases que muestra el modelo estático, mientras el diagrama de secuencia muestra el modelo dinámico, por lo que ambos deben estar relacionados entre sí con las mismas clases, métodos, atributos y a su vez relacionado con el flujo normal y el flujo alternativo de eventos que se describen en los casos de uso. Por ello, el diagrama de casos de uso permite el modelado de una vista de negocio del escenario, para que el diagrama de secuencia contenga los detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario y mensajes intercambiados entre los objetos.

Típicamente se examina la descripción de un caso de uso para determinar qué objetos son necesarios para la implementación del escenario. Si se dispone de la descripción de cada caso de uso como una secuencia de varios pasos, entonces se puede "caminar sobre" esos pasos para descubrir qué objetos son necesarios para que se puedan seguir los pasos. Un diagrama de secuencia muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales.

Se muestran ejemplos de diagramas de secuencia, los cuales van asociados a los casos de uso.

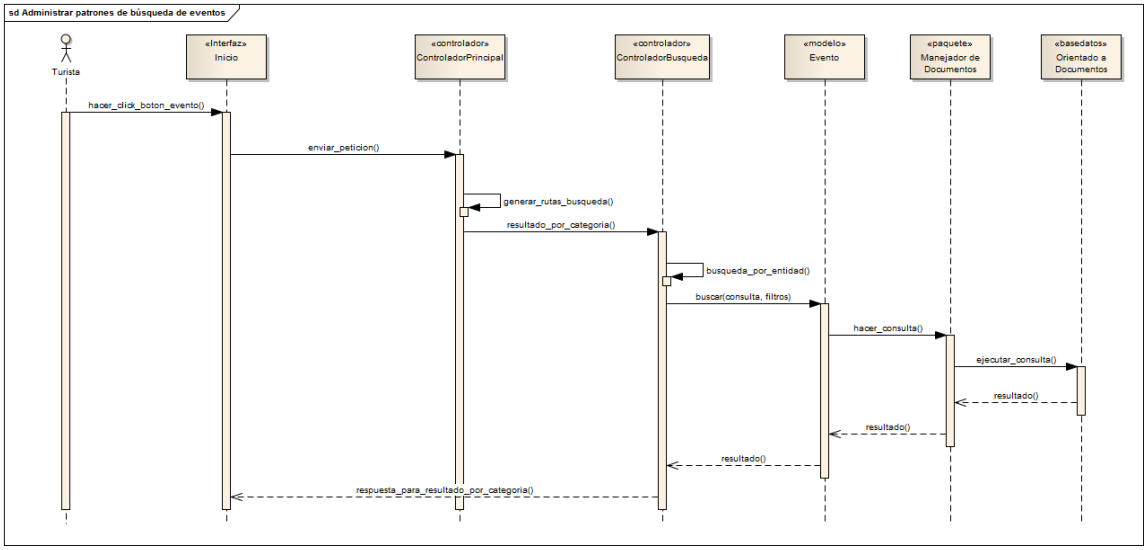


Figura 51. Administrar patrones de búsqueda de eventos.

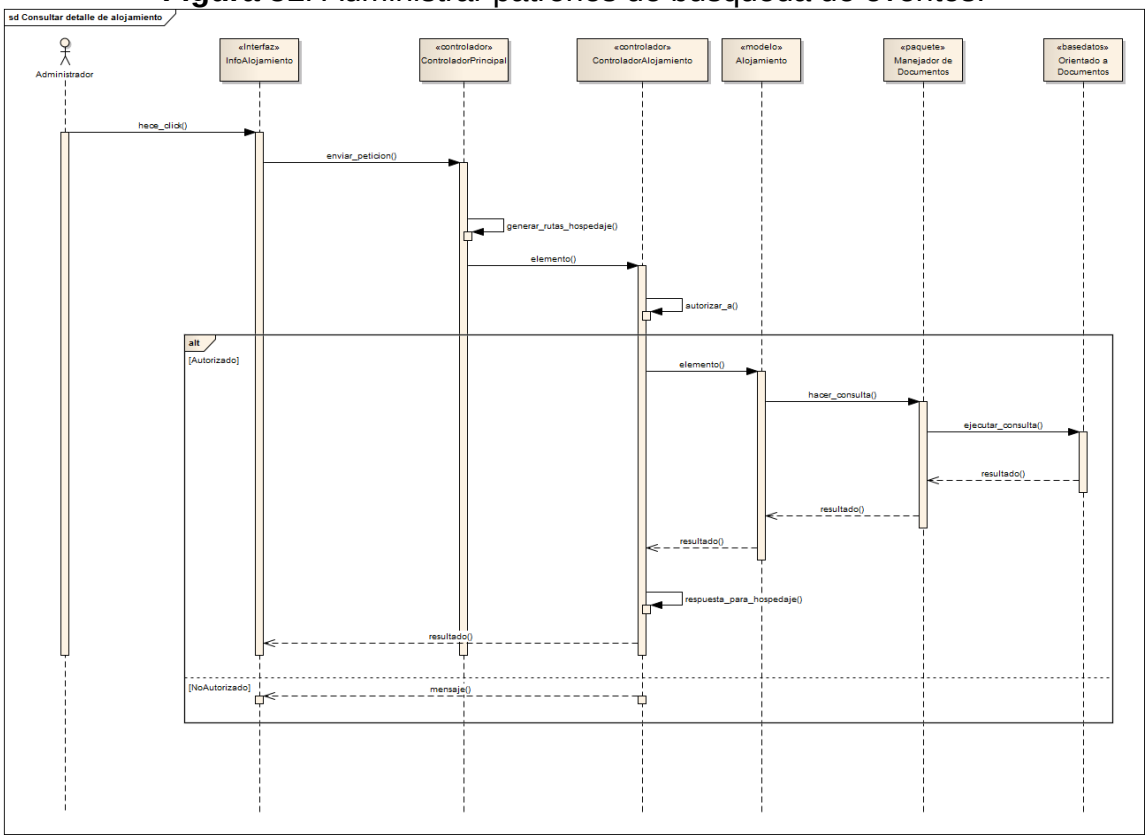


Figura 52. Diagrama de Secuencias Consultar Detalle de Alojamiento.

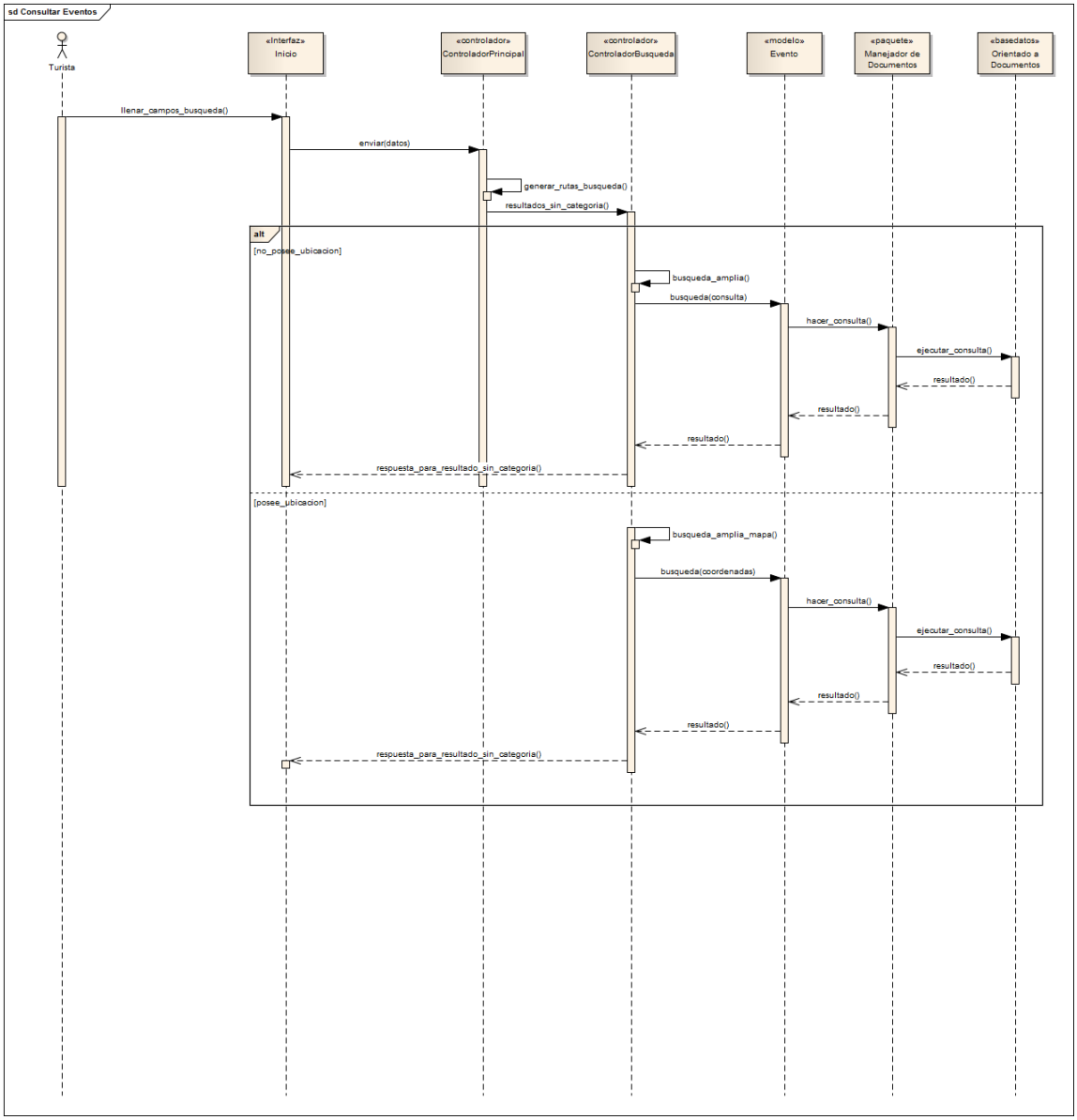


Figura 53. Diagrama de Secuencias Consultar Eventos Turista.

3.1.5.4. Vista de Implementación.

En este apartado se especifican detalles de implementación del sistema como lo son lenguajes a utilizar, plataformas de desarrollo, etc. Esta vista se vera de forma gráfica mediante el diagrama de componentes especificado.

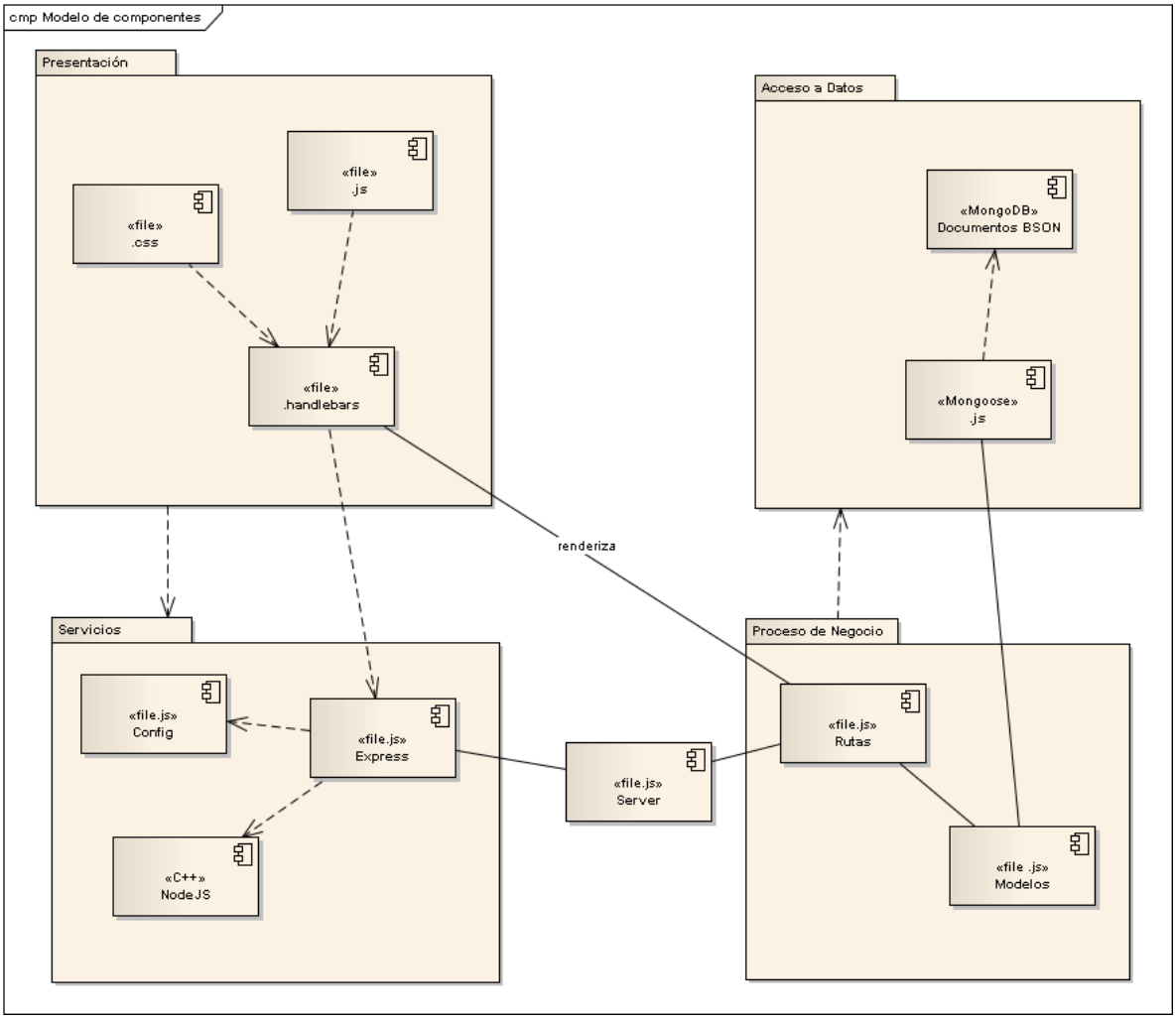


Figura 54. Diagrama de Componentes.

3.1.5.5. Vista de Despliegue.

Esta vista muestra instalación y operación de la aplicación, es decir, muestra como instalaran los componentes en la plataforma. Esta vista esta soportada por el diagrama de despliegue mostrado.

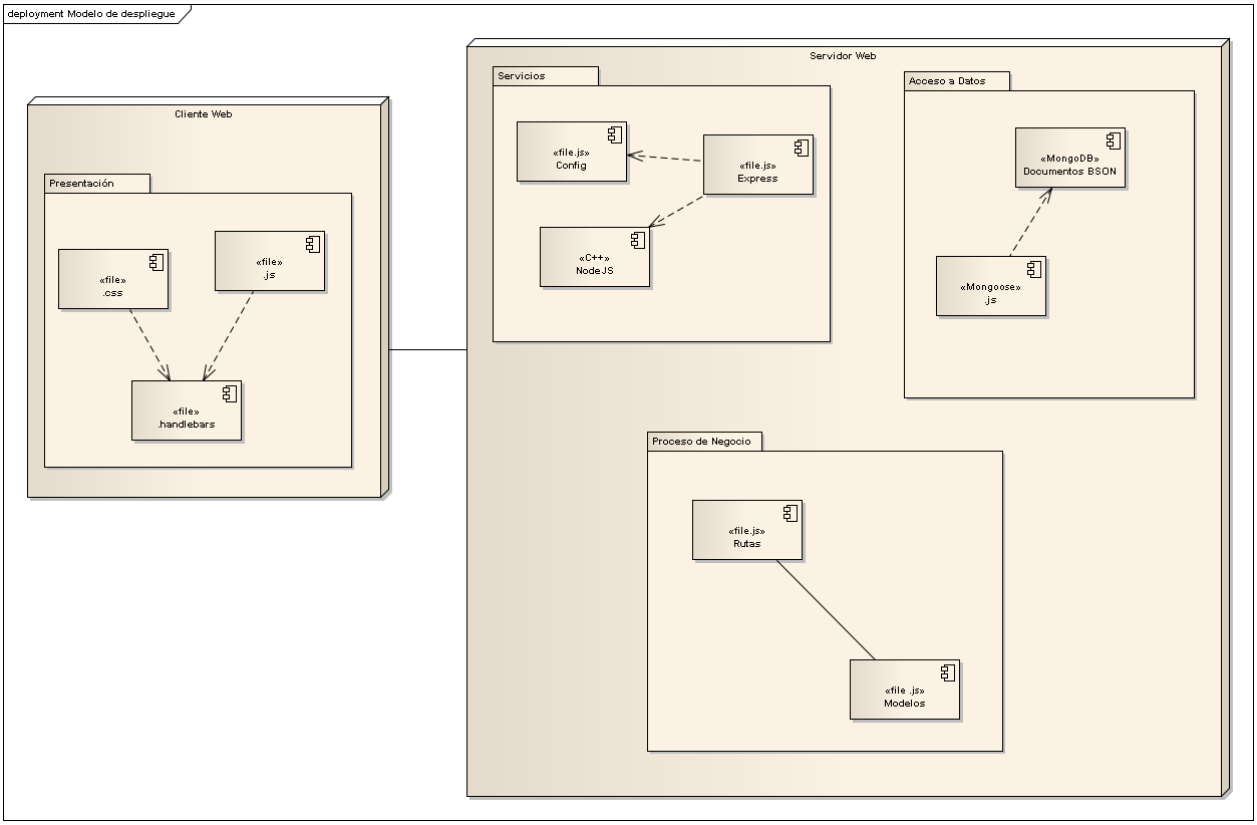


Figura 55. Diagrama de Despliegue.

3.1.6. Planificación de la Iteración i.

La Planificación de la Iteración es un conjunto de actividades de gestión que se ejecutó con la participación de todo el Equipo de Desarrollo y tuvo una duración estimada de seis (6) horas. El resultado fue un plan que describió y programó las actividades técnicas de elaboración del incremento asociado a la iteración i, para ello el Equipo de Desarrollo debió actualizar la pila de historias con nuevos requisitos que surgieron de la iteración anterior, actualizar la pila de historias correspondientes al incremento que se iba a elaborar durante la iteración, determinar las actividades asociadas a cada historia del incremento, planificar las actividades elaborando un cronograma que representó el Plan de la Iteración y, por último, actualizar el Plan del Proyecto con las modificaciones necesarias.

Se consideró que la duración de una iteración es fija y que debe estar entre 1 y 6 semanas, que la iteración produce un incremento que implementa un subconjunto de requisitos funcionales y no funcionales de la aplicación, en donde no todos los incrementos son del tipo entregable, ya que un incremento entregable puede requerir varias iteraciones, mientras que un incremento no entregable es parte de un incremento entregable.

Una vez que se planificó y se inició la iteración, la pila de historias de la iteración no se modificó, ya que los cambios surgidos en los requisitos durante la iteración fueron evaluados al final de la iteración a través del proceso de Verificación y Validación de la Iteración, considerando aquellos cambios que fueron aprobados para incorporarlos a la pila de historias del proyecto en la siguiente iteración.

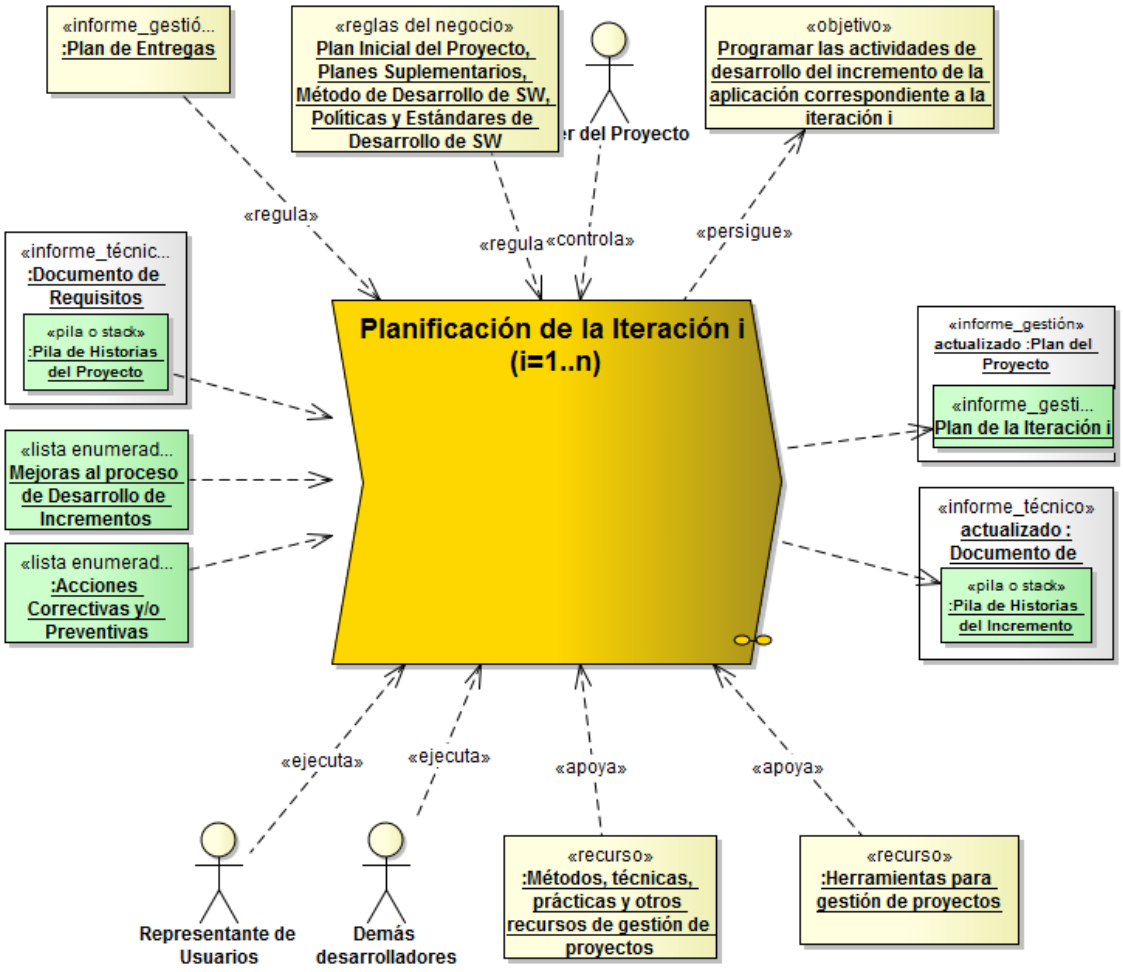


Figura 56. Diagrama de planificación de la iteración i-ésima.
Fuente: Montilva y Barrios, 2014.

La planificación de la iteración i fue controlado por la líder del proyecto, ejecutada por los demás desarrolladores y por los representantes de los diferentes tipos de usuarios; dicho proceso estuvo apoyado principalmente por herramientas de gestión de proyectos y por la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

La Planificación de la Iteración i fue regulada por un conjunto de reglas del negocio que impusieron condiciones y limitaciones a la ejecución de las

actividades de dicho proceso, entre las cuales están el Plan Inicial del Proyecto y los Planes Suplementarios, ya que son productos de gestión que actúan como reglas de negocio a lo largo de la ejecución de todo el proyecto, como el alcance, los tiempos de ejecución, los costos asociados, el uso de los recursos, la calidad de los productos, el enfoque del contexto del software desarrollado, además de las normativas internas de la Licenciatura en Informática para la ejecución de las alternativas de grado.

El objetivo principal obtenido fue la programación de las actividades de desarrollo del incremento de la aplicación correspondiente a dicha iteración i.

3.1.7. Refinamiento de Requisitos del Incremento.

En el proceso de Refinamiento de Requisitos del Incremento, el conjunto de requisitos que está asociado al incremento que se consideró elaborar durante esta iteración i, se refinó y se especificó en detalle.

Cada requisito funcional del incremento se refinó y se modeló usando casos de uso. El resultado obtenido fue una especificación de requisitos del incremento, la cual fue incorporada al documento de requisitos de la aplicación, a fin de mantener este documento actualizado.

Las Especificaciones de Requisitos elaboradas por el equipo de desarrollo durante el proceso de Desarrollo Inicial de Requisitos, fueron profundizadas durante el proceso de Refinamiento de Requisitos del Incremento, considerando sólo aquellos requisitos que correspondían al incremento en desarrollo.

Las vistas funcionales de casos de uso elaboradas durante el proceso de Diseño Inicial de la Aplicación fueron usadas en este proceso como un insumo, junto a la pila de historias del incremento y la especificación de requisitos iniciales, en la elaboración de las especificaciones de requisitos del incremento.

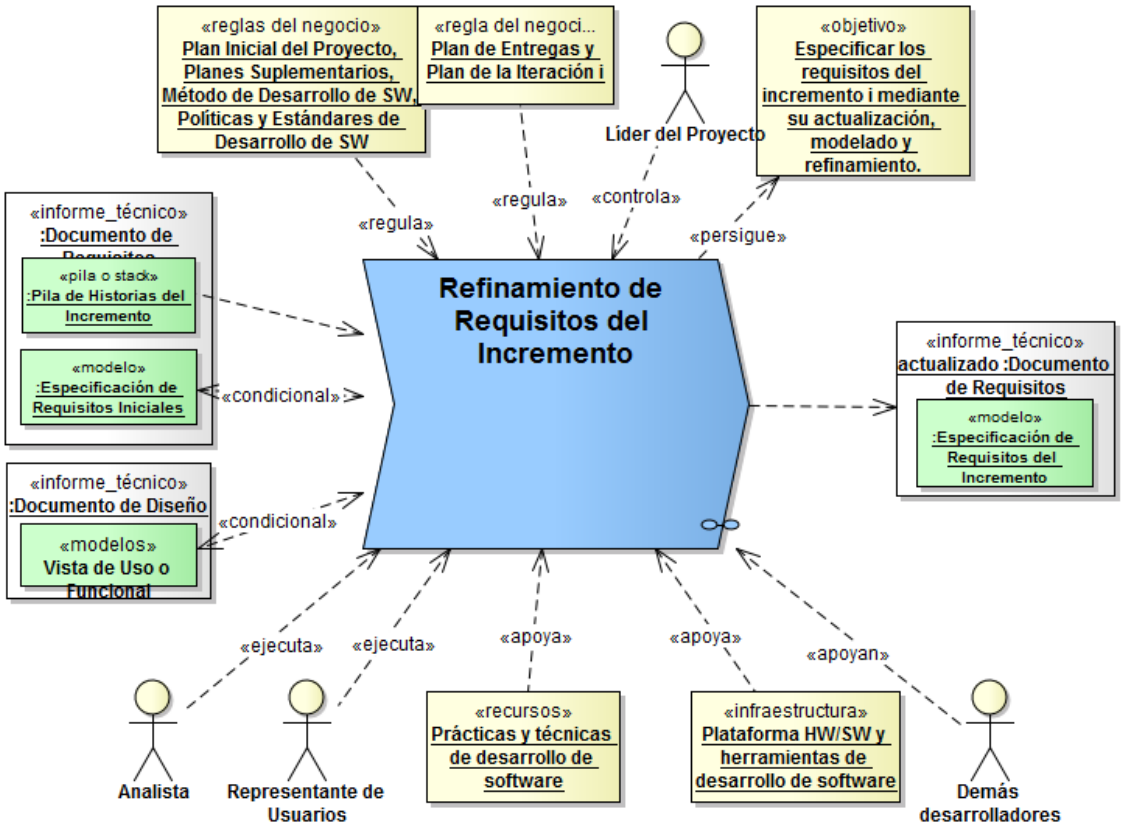


Figura 57. Diagrama de refinamiento de requisitos del incremento i-ésimo. Fuente: Montilva y Barrios, 2014.

Las reglas de negocio que regularon el proceso de Refinamiento de Requisitos del Incremento estaban comprendidas por el Plan Inicial del Proyecto, los Planes Suplementarios, el método de desarrollo de software Blue Watch, las políticas y estándares de desarrollo de software y, en mayor proporción, por el Plan de Entregas y el Plan de la Iteración i.

El Refinamiento de Requisitos del Incremento fue controlado por la líder del proyecto, ejecutado por los analistas y por los representantes de los diferentes tipos de usuarios y apoyada por los demás desarrolladores; dicho proceso contó con una infraestructura en la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y técnicas de

desarrollo de software acordes a Softour.

El objetivo obtenido fue la especificación de los requisitos del incremento i mediante su actualización, modelado y refinamiento.

3.1.8. Diseño Detallado del Incremento.

El Diseño Detallado del Incremento es un proceso iterativo donde se refinaron los modelos obtenidos durante el Diseño Inicial de la Aplicación, concretamente, aquellas partes de los modelos asociados al incremento en fase de producción, a partir del Diseño Inicial de la Aplicación y de los diseños producidos en las iteraciones anteriores, en el caso de no ser la primera iteración.

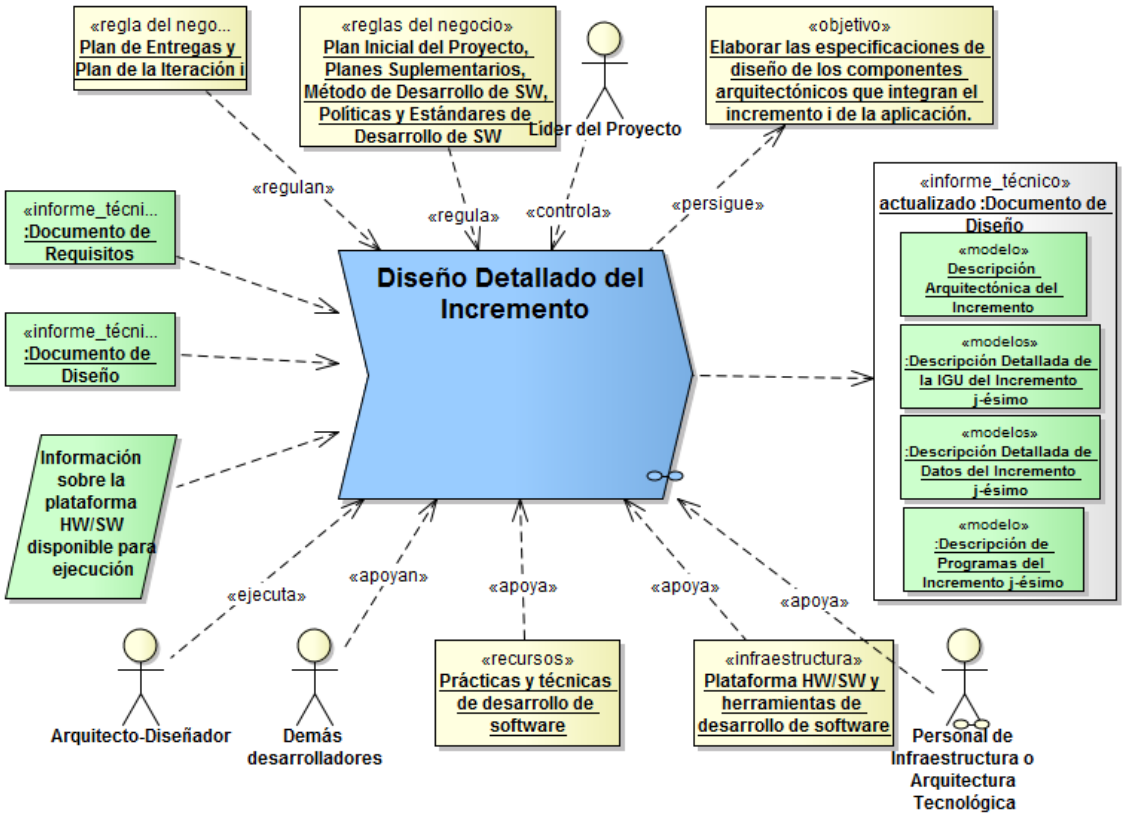


Figura 58. Diagrama de diseño detallado del incremento i-ésimo.
Fuente: Montilva y Barrios, 2014.

Las reglas de negocio que regularon el Diseño Detallado del Incremento estaban comprendidas, de manera específica, por el Plan de Entregas y el Plan de la Iteración i, y, de manera general, por el Plan Inicial del Proyecto, los Planes Suplementarios, el método de desarrollo de software Blue Watch, las políticas y estándares de desarrollo de software.

El Diseño Detallado del Incremento fue controlado por la líder del proyecto, ejecutado por el arquitecto diseñador, apoyado por los demás desarrolladores y por el personal de infraestructura o arquitectura tecnológica; además dicho proceso contó con la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

El objetivo obtenido fue la elaboración de las especificaciones de diseño de los componentes arquitectónicos que integran el incremento i de la aplicación.

3.1.9. Codificación del Incremento.

La Codificación y Pruebas del Incremento fue un proceso iterativo compuesto por actividades de codificación y pruebas que se combinaron para agilizar el desarrollo del incremento. En este proceso, se codificó y probó cada incremento o la parte de él que se desarrolló durante la iteración i-ésima. Las pruebas se realizaron en los tres niveles de pruebas conocidos: unidad, integración y sistema.

Las pruebas de unidad e integración fueron ejecutadas por los programadores, durante la codificación de los programas, mientras que las pruebas del incremento, que representaron las pruebas a nivel del sistema, fueron ejecutadas por el experto en pruebas.

En los niveles de unidad e integración, el proceso de codificación fue guiado por las pruebas, según los pasos siguientes:

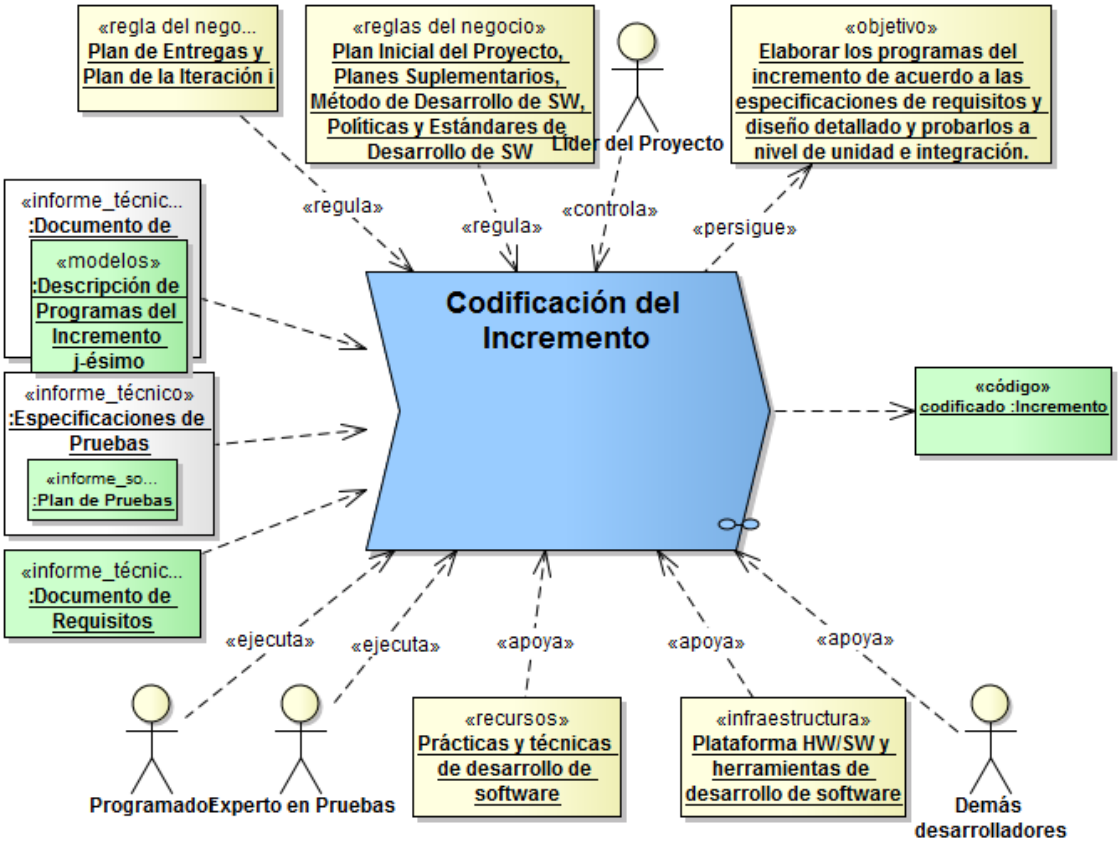


Figura 59. Diagrama de codificación del incremento i-ésimo.
Fuente: Montilva y Barrios, 2014.

Se eligió un requisito determinado de la lista o pila de historias del incremento i, considerando el requisito que dio mayor conocimiento del problema y que a la vez fue implementable.

Se escribió la prueba para el requisito, donde se orientó al programador para que entendiera las especificaciones y los requisitos de la funcionalidad por implementar y tomara la perspectiva de un cliente, considerando el código a través de sus interfaces.

Se verificó que la prueba fallaba, ya que si la prueba no fallaba era porque el requisito ya estaba implementado o porque la prueba que se planteó no era la correcta para la codificación del requisito.

Se escribió la implementación a través de un código sencillo que hizo que la prueba funcionara, usando la metáfora "déjelo simple".

Se ejecutaron las pruebas automatizadas para verificar si todo el conjunto de pruebas funcionaba correctamente.

Se eliminó el código duplicado a través de la factorización y reutilización del código común, los cambios se realizaron en pequeñas porciones de códigos que fueron transformados en métodos comunes y luego se sometieron a pruebas hasta que funcionaron.

Se actualizó la lista de requisitos en la pila de historias del incremento i, señalando o tachando el requisito implementado, también se agregaron nuevos requisitos que se observaron necesarios durante este ciclo y se agregaron los requerimientos de diseño.

Al finalizar las pruebas de unidad e integración, fue necesario ejecutar las pruebas a nivel del sistema, es decir, las pruebas del incremento con el resto del sistema que hasta ese punto se había integrado, con el objetivo de verificar el cumplimiento de los requisitos no funcionales, tales como: rendimiento del incremento, interfaz gráfica del incremento, interoperabilidad con otras aplicaciones.

Las reglas de negocio que regularon la Codificación del Incremento fueron el Plan de Entregas, el Plan de la Iteración i, el Plan Inicial del Proyecto, los Planes Suplementarios, el método de desarrollo de software *Blue Watch,* las políticas y estándares de desarrollo de software.

La Codificación del Incremento fue controlada por la líder del proyecto, ejecutado por el programador y el experto en pruebas, apoyada por los demás desarrolladores; además dicho proceso contó con la plataforma de hardware y software seleccionado para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

El objetivo obtenido fue la elaboración de los programas del incremento de acuerdo a las especificaciones de requisitos y el diseño detallado y hacer las pruebas a nivel de unidad e integración.

3.1.10. Pruebas de Incremento.

El Diseño de Pruebas del Incremento es un proceso iterativo que constó de un conjunto de actividades técnicas que tuvieron como objetivo elaborar las especificaciones necesarias para probar el incremento.

En este proceso se especificaron las pruebas del incremento en sus tres niveles: el nivel de unidad consistió en elaborar la suite de pruebas asociada a cada clase que forma parte del incremento i, el nivel de integración consistió en elaborar la suite de pruebas para la integración de las clases que conforman el incremento i, el nivel de sistema consistió en elaborar la suite de pruebas que permite probar el incremento como un todo.

Las reglas de negocio que regularon las Pruebas del Incremento fueron el Plan de Entregas, el Plan de la Iteración i, el Plan Inicial del Proyecto, los Planes Suplementarios, el método de desarrollo de software *Blue Watch*, las políticas y estándares de desarrollo de software.

El proceso de Pruebas del Incremento fue controlado por la líder del proyecto, ejecutado por el experto en pruebas, apoyada por el programador; además dicho proceso contó con la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

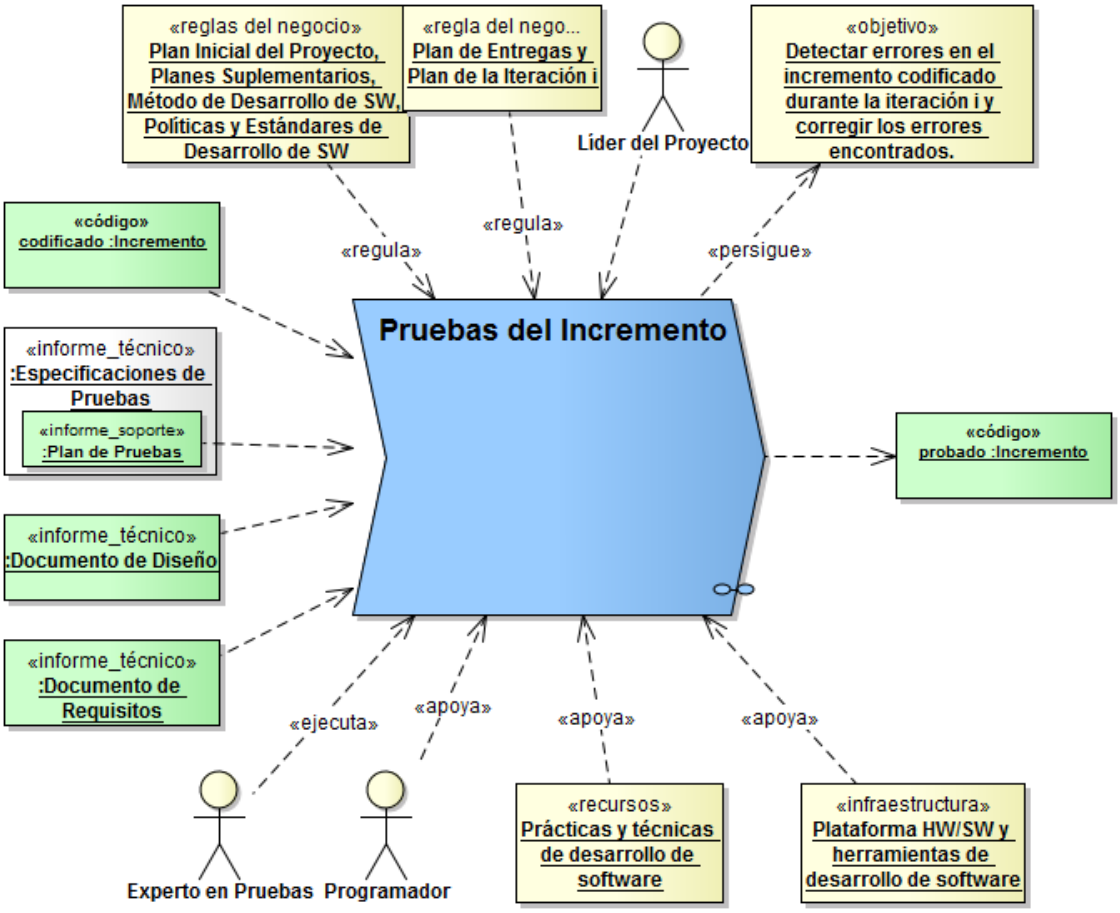


Figura 60. Diagrama de pruebas del incremento i-ésimo.
Fuente: Montilva y Barrios, 2014.

El objetivo obtenido fue la detección de los errores en el incremento codificado durante la iteración i y la corrección de dichos errores, hasta que el incremento superara las pruebas.

3.1.11. Verificación y Validación de Incrementos.

La Verificación y Validación del Incremento es un proceso iterativo de soporte que permitió asegurar que el incremento producido durante la iteración i satisfizo las condiciones establecidas: cumplió con todos los requisitos especificados al inicio de la iteración i y con lo que el usuario esperaba del incremento al inicio de la iteración.

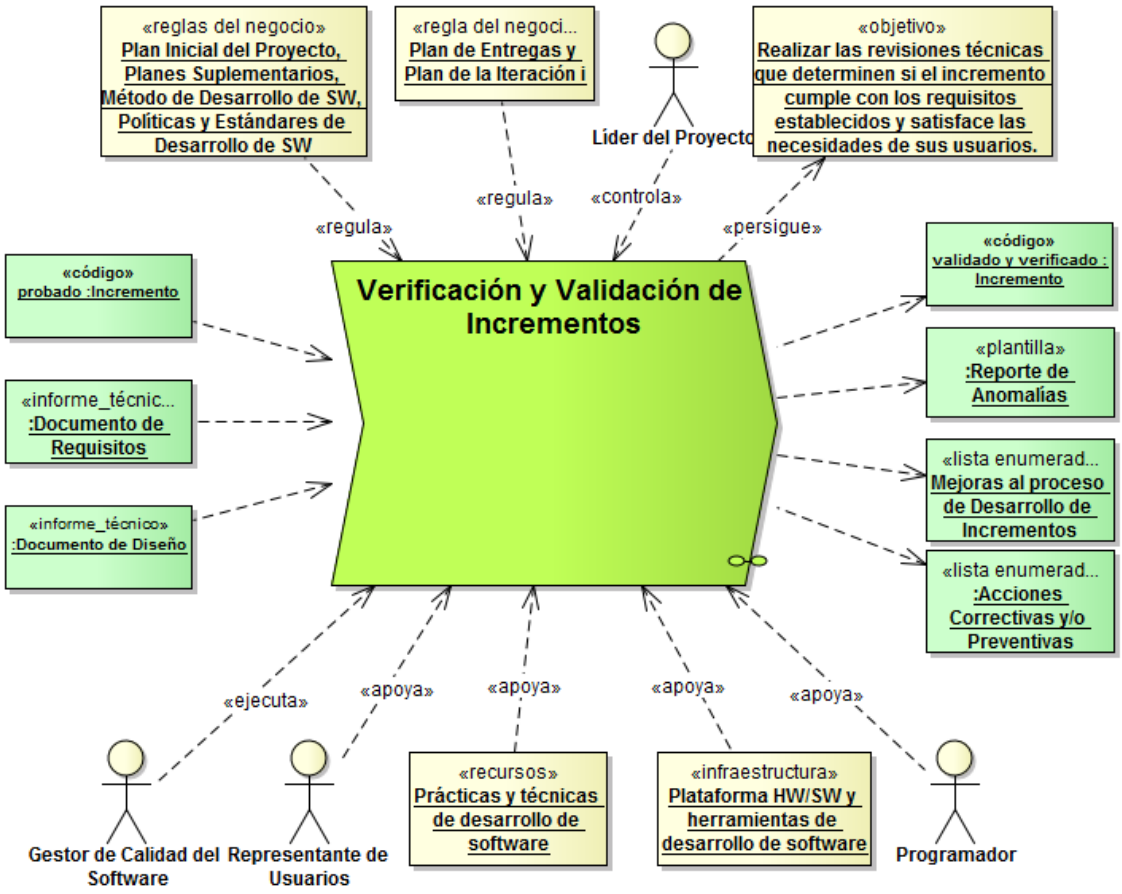


Figura 61. Diagrama de verificación y validación de incrementos.
Fuente: Montilva y Barrios, 2014.

La verificación se llevó a cabo a través de un proceso de Revisión Técnica donde los desarrolladores evaluaron el incremento producido con la finalidad de asegurarse que cumplió con todos los requisitos definidos en la pila de historias del incremento y especificados en detalle al inicio de la iteración i.

La validación la llevaron a cabo los usuarios con la finalidad de determinar si el incremento producido en la iteración i fue el correcto; es decir, satisfizo sus necesidades reales y fue lo que ellos esperaban.

La Verificación y Validación del Incremento se llevó a cabo mediante la aplicación de técnicas tales como: la Revisiones del Producto en donde se

evaluó el incremento en sesiones de grupo, en las que el incremento se sometió a inspecciones y recorridos estructurados; las Demostraciones en donde el incremento se le presentó a los usuarios o clientes a través de una ejecución controlada; las Pruebas Alfa en donde el usuario usó durante un periodo de pruebas determinado el incremento con la finalidad de detectar anomalías, fallas o inconsistencias.

Junto a la Verificación y Validación del Incremento se llevaron a cabo sesiones adicionales de retrospectiva del proceso, que ayudaron a mejorar las actividades relacionadas con la siguiente iteración.

Las reglas de negocio que regularon la Verificación y Validación del Incremento fueron el Plan de Entregas, el Plan de la Iteración i, el Plan Inicial del Proyecto, los Planes Suplementarios, el método de desarrollo de software *Blue Watch*, las políticas y estándares de desarrollo de software.

El proceso de Verificación y Validación del Incremento fue controlado por la líder del proyecto, ejecutado por el gestor de calidad del software, apoyado por el programador y los representantes de usuarios; además dicho proceso contó con la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

El objetivo obtenido fue la realización de las revisiones técnicas que determinaron si el incremento cumplió con los requisitos establecidos y satisfizo las necesidades de los usuarios.

3.1.12. Entrega Final de la Aplicación.

La Entrega Final de la Aplicación fue un conjunto de procesos técnicos que se encargaron de instalar la aplicación en su ambiente de producción, probarla en ese ambiente, capacitar a sus usuarios y ponerla en operación.

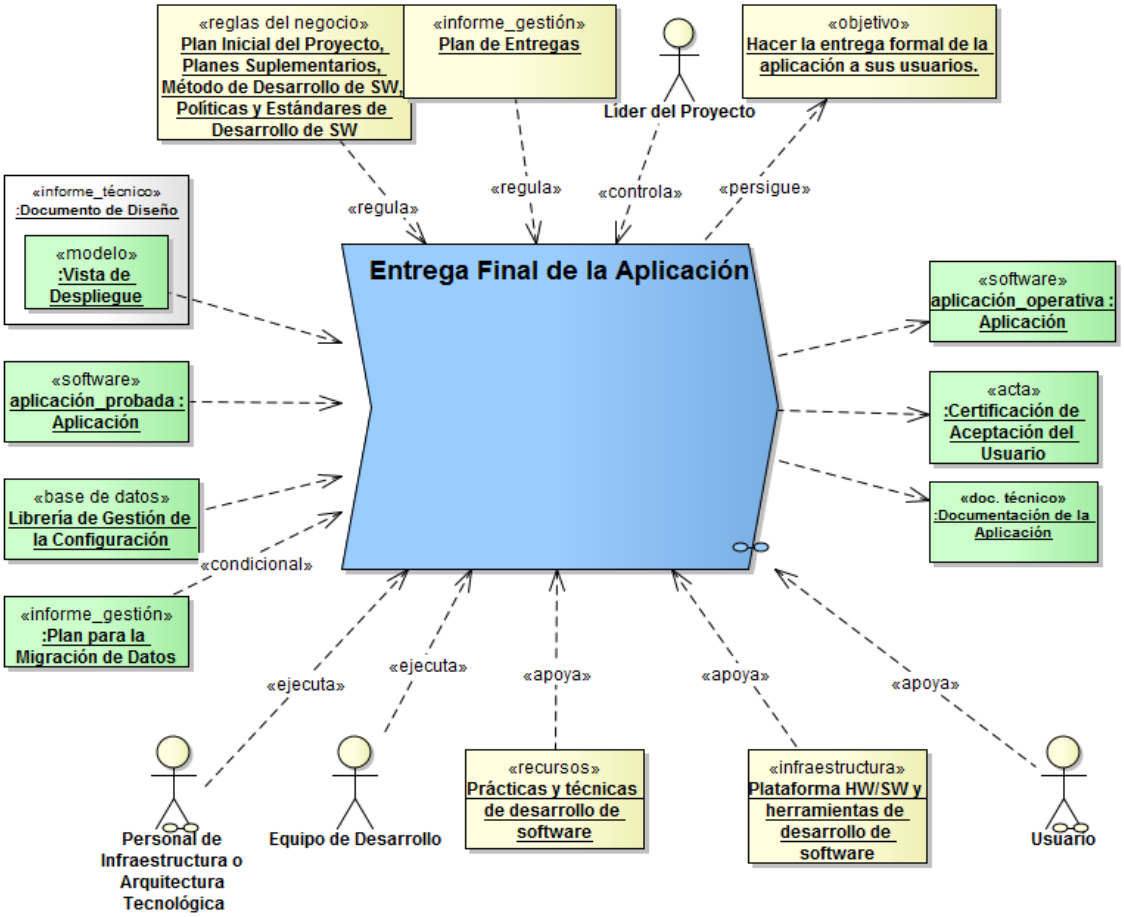


Figura 62.Diagrama de entrega final de la aplicación.
Fuente: Montilva y Barrios, 2014.

Las reglas de negocio que regularon la Entrega Final de la Aplicación fueron el Plan de Entregas, el Plan Inicial del Proyecto, los Planes Suplementarios, el método de desarrollo de software *Blue Watch*, las políticas y estándares de desarrollo de software.

El proceso de Entrega Final de la Aplicación fue controlado por la líder del proyecto, ejecutado por todo el equipo de desarrollo y por el personal de infraestructura o arquitectura tecnológica, apoyado por los representantes de usuarios; además dicho proceso contó con la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y

técnicas de desarrollo de software acordes a Softour.

El objetivo obtenido fue entregar formalmente la aplicación al cliente final, el cual quedó representado por el Departamento de la Licenciatura en Informática del Núcleo de Sucre de la Universidad de Oriente. Esta entrega se realizó en una exposición final ante un jurado evaluador.

3.1.13. Cierre del Proyecto.

El Cierre del Proyecto fue un conjunto de actividades de gestión que se encargó de dar por finalizado, formalmente, el proyecto; aseguró que todas las actividades de los procesos técnicos, de gestión y de soporte del proyecto, o de la fase se habían concluido.

Este proceso contempla dos actividades: el cierre administrativo del proyecto que incluye la evaluación del desempeño administrativo, el cierre de las contrataciones, el pago final a los contratistas y proveedores del proyecto; el cierre técnico del proyecto que consiste en el análisis del desempeño y los resultados del proyecto, la documentación de las lecciones aprendidas, la propuestas de mejoras a los procesos de desarrollo, la elaboración de los indicadores de gestión del proyecto, el almacenamiento de los productos que fueron sometidos a Gestión de Configuración para ser utilizados en futuros proyectos. Todo esto agrupado en la presente tesis de grado.

Las reglas de negocio que regularon el Cierre del Proyecto fueron el Portafolio de Proyectos y la normativa del Departamento de la Licenciatura en Informática del Núcleo de Sucre de la Universidad de Oriente para gestión de proyectos, en este caso el reglamento para la discusión de tesis en la modalidad alternativas de grado.

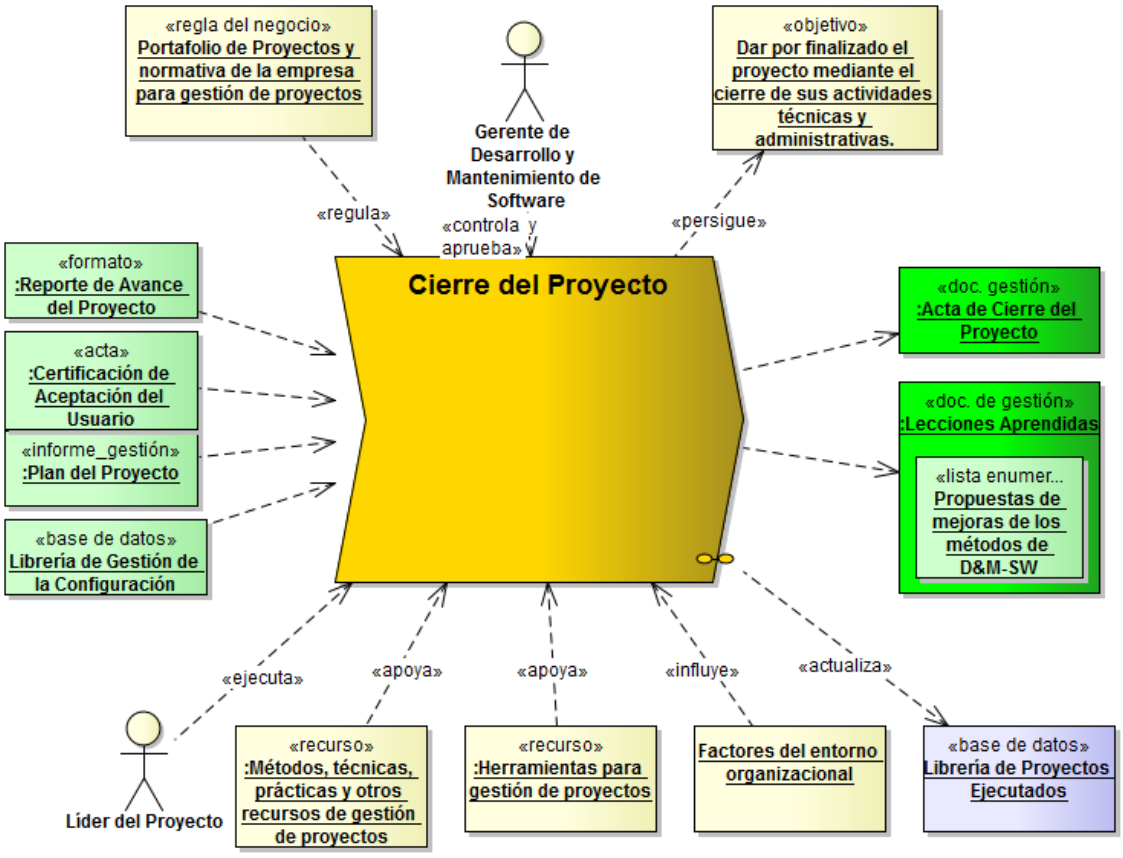


Figura 63. Diagrama de cierre del proyecto.
Fuente: Montilva y Barrios, 2014.

El proceso de Cierre del Proyecto fue controlado por el Gerente de Desarrollo y Mantenimiento de Software representado por los profesores de las asignaturas de alternativas de grado y jurado calificador, ejecutado por todo el equipo de desarrollo incluyendo a la líder del proyecto; además dicho proceso se apoyó en herramientas para gestión de proyectos y utilizó métodos, técnicas, prácticas y otros recursos de gestión de proyectos.

El objetivo obtenido dar por finalizado el proyecto mediante el cierre de sus actividades técnicas y administrativas, y actualizar la librería de proyectos ejecutados por el Departamento de la Licenciatura en Informática del Núcleo de Sucre de la Universidad de Oriente.

3.1.14. Dirección, Seguimiento y Control del Proyecto.

Los procesos de Dirección del Proyecto y Seguimiento y Control del Proyecto se iniciaron una vez que el proyecto fue planificado y se encargaron de asegurar que el proyecto se ejecutara de acuerdo a lo planificado y dentro del presupuesto aprobado y, asegurar que los productos tuvieran la calidad especificada y velar porque el proceso estuviera dentro del alcance establecido durante la planificación.

Figura 64. Diagrama de subprocesos de gestión del proyecto.
Fuente: Montilva y Barrios, 2014.

3.1.14.1. Dirección del Proyecto.

Las reglas de negocio que regularon el proceso de gestión de Dirección del Proyecto fueron el Portafolio de Proyectos y las normativas de la Licenciatura en Informática para la gestión de proyectos en el marco de las alternativas de grado.

El proceso de gestión de Dirección del Proyecto fue controlado y ejecutado por el Gerente de Desarrollo y Mantenimiento de Software representado por los profesores responsables de dirigir las alternativas de grado, contando en el proceso de ejecución con la líder del proyecto; además dicho proceso se apoyó en herramientas para gestión de proyectos y utilizó métodos, técnicas, prácticas

y otros recursos de gestión de proyectos.

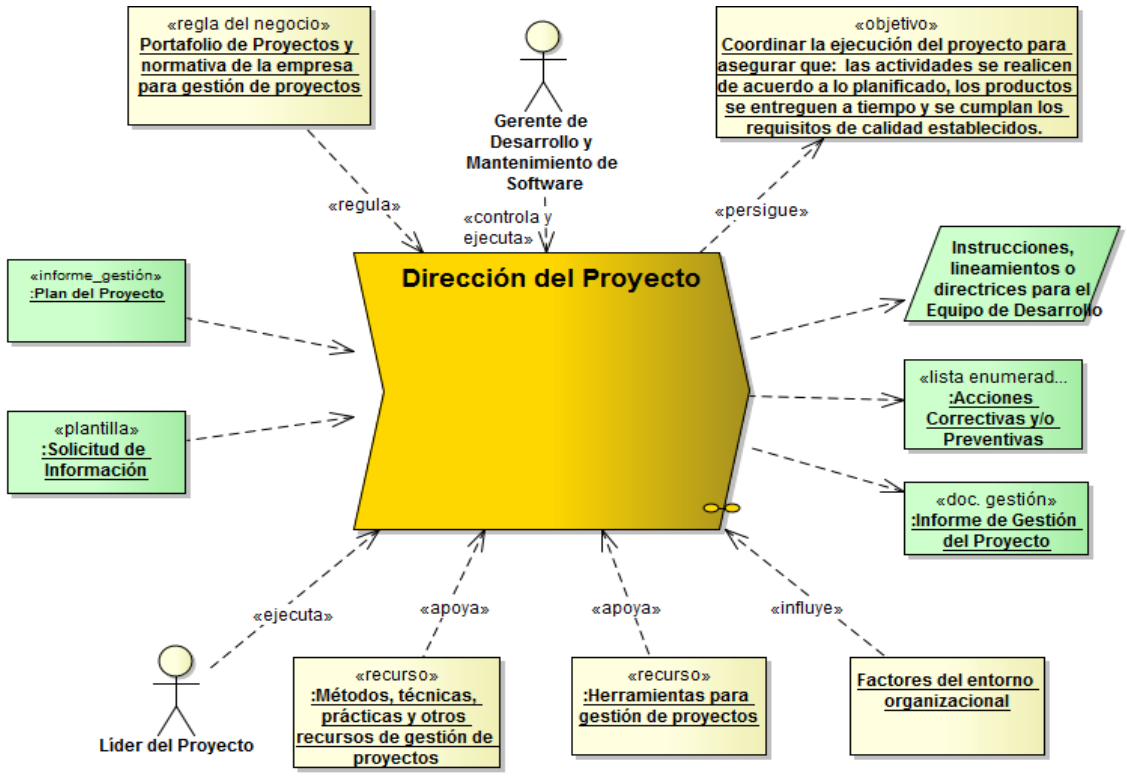


Figura 65. Diagrama de dirección del proyecto.
Fuente: Montilva y Barrios, 2014.

El objetivo obtenido fue coordinar la ejecución del proyecto para asegurar que las actividades se realizaran de acuerdo a lo planificado, los productos se entregaran a tiempo y se cumplieran los requisitos de calidad establecidos.

3.1.14.2. Seguimiento y Control del Proyecto.

Las reglas de negocio que regularon el proceso de Seguimiento y Control del Proyecto fueron el Portafolio de Proyectos y las normativas de la Licenciatura en Informática para la gestión de proyectos en el marco de las alternativas de grado.

El proceso de Seguimiento y Control del Proyecto fue controlado y ejecutado

por el Gerente de Desarrollo y Mantenimiento de Software representado por los profesores responsables de dirigir las alternativas de grado, contando en el proceso de ejecución con la líder del proyecto; además dicho proceso se apoyó en herramientas para gestión de proyectos y utilizó métodos, técnicas, prácticas y otros recursos de gestión de proyectos.

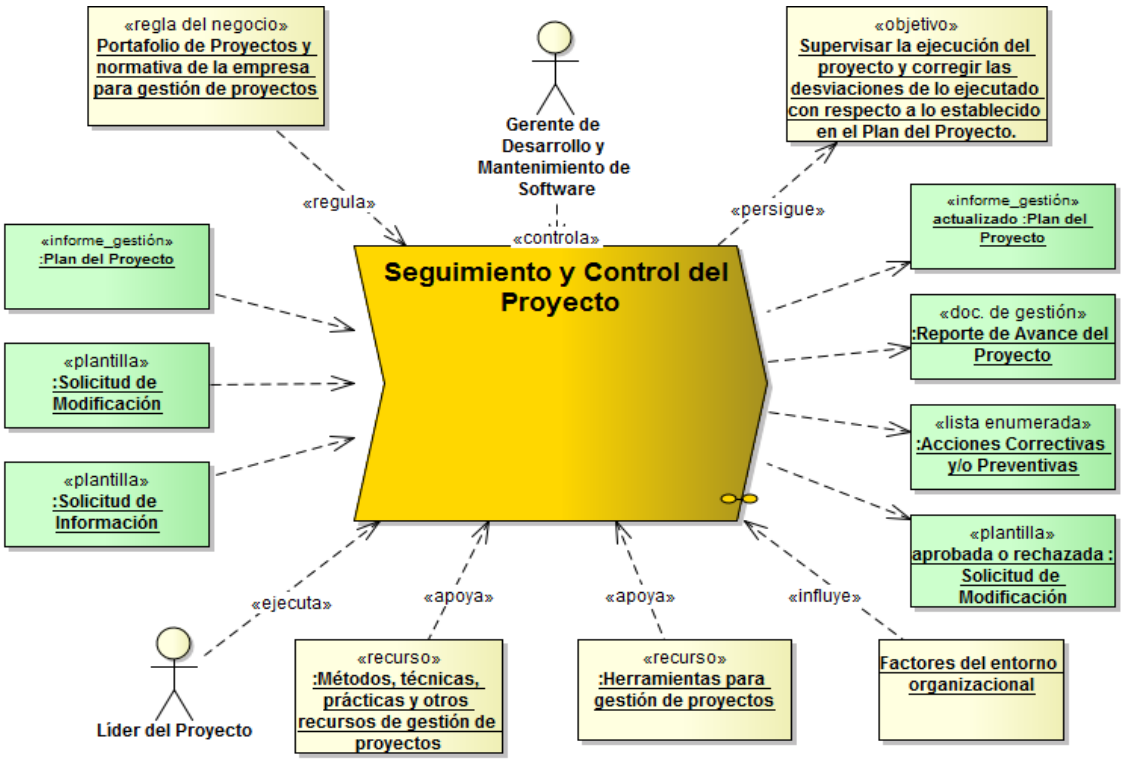


Figura 66. Diagrama de seguimiento y control del proyecto.
Fuente: Montilva y Barrios, 2014.

El objetivo obtenido fue supervisar la ejecución del proyecto y corregir las desviaciones de lo ejecutado con respecto a lo establecido en el Plan del Proyecto.

3.1.15. Gestión de Requisitos.

La Gestión de Requisitos fue el proceso de soporte que se inició una vez que había culminado el Desarrollo Inicial de Requisitos. Es un proceso complementario a los procesos de Desarrollo Inicial de Requisitos y

Refinamiento de Requisitos del Incremento.

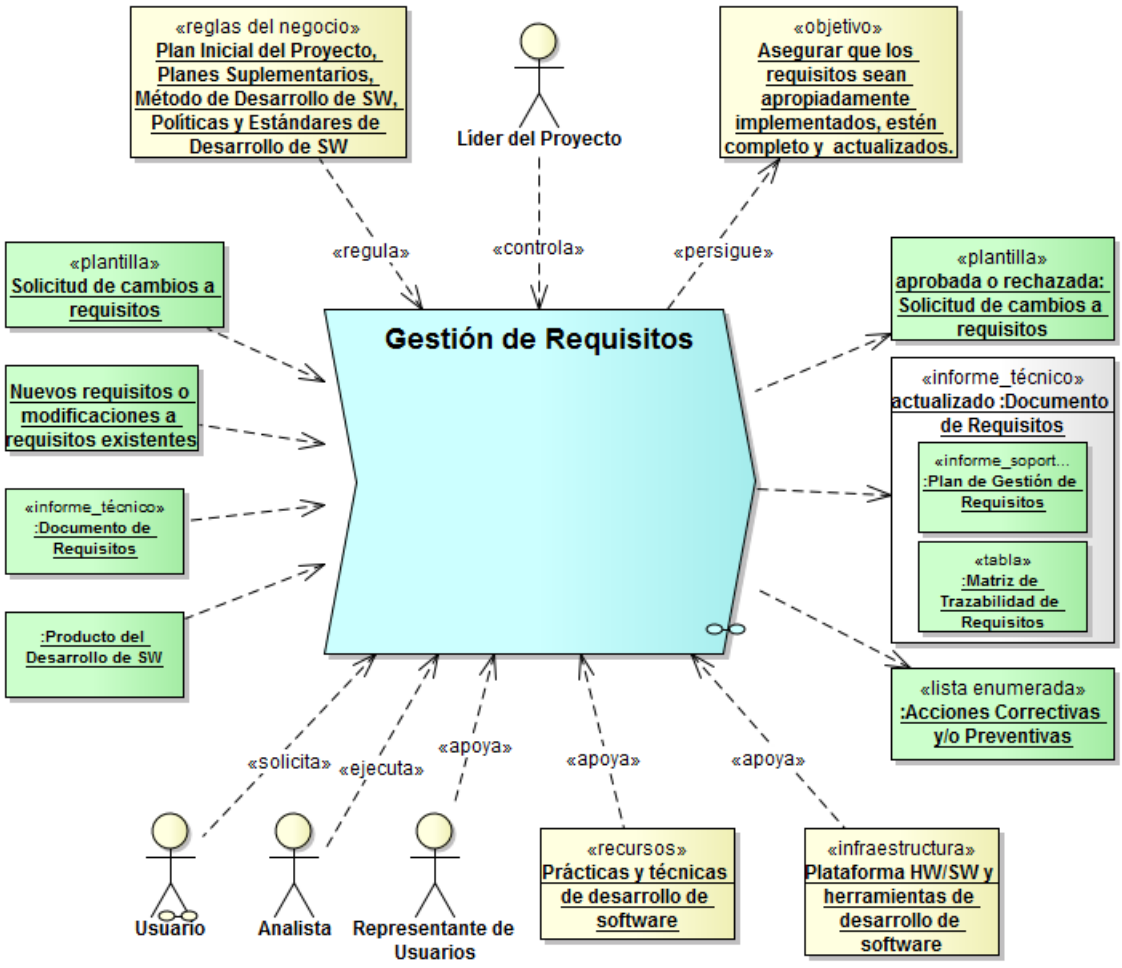


Figura 67. Diagrama de gestión de requisitos.
Fuente: Montilva y Barrios, 2014.

La Gestión de Requisitos se encargó de: administrar los cambios a los requisitos que se habían definidos y/o especificados en el Documento de Requisitos, dicho cambio ocurrió después de realizado el proceso de Desarrollo Inicial de Requisitos; hacerle seguimiento a cada uno de los requisitos definidos y/o especificados en el Documento de Requisitos, a fin de garantizar que el requisito fue considerado en todos los demás productos del desarrollo de la aplicación.

Las reglas de negocio que regularon el proceso de Gestión de Requisitos fueron el Plan Inicial del Proyecto, los Planes Suplementarios, el Método de Desarrollo de Software *Blue Watch*, las Políticas y Estándares de Desarrollo de Software.

El proceso de Gestión de Requisitos fue controlado por la líder del proyecto, ejecutado por el analista, y apoyado por los representantes de los usuarios; dicho proceso contó con una infraestructura en la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

El objetivo obtenido fue asegurar que los requisitos estuvieron apropiadamente implementados, completos y actualizados.

3.1.16. Verificación y Validación de Productos (Estática).

La Verificación y Validación de Productos fue un proceso de soporte que consta de un conjunto de subprocesos encargados de garantizar la calidad de los productos que se elaboraron en los diferentes procesos del desarrollo de la aplicación.

La Verificación se encargó de revisar los productos elaborados para determinar si satisfacen los requisitos, especificaciones de diseño y normativas establecidas para el proyecto, fue ejecutado por el personal técnico.

La Validación se encargó de revisar los productos elaborados para determinar si es el correcto; es decir, si satisfacen las necesidades reales de los usuarios, fue ejecutado por los usuarios, promotores y otros interesados.

Este conjunto de procesos Verificación y Validación fue de carácter estático y consistió en revisiones técnicas de los productos. Difiere de las pruebas realizadas durante los procesos técnicos. Las pruebas son consideradas procesos Verificación y Validación, pero de carácter dinámico, pues requieren

de la ejecución de los programas desarrollados.

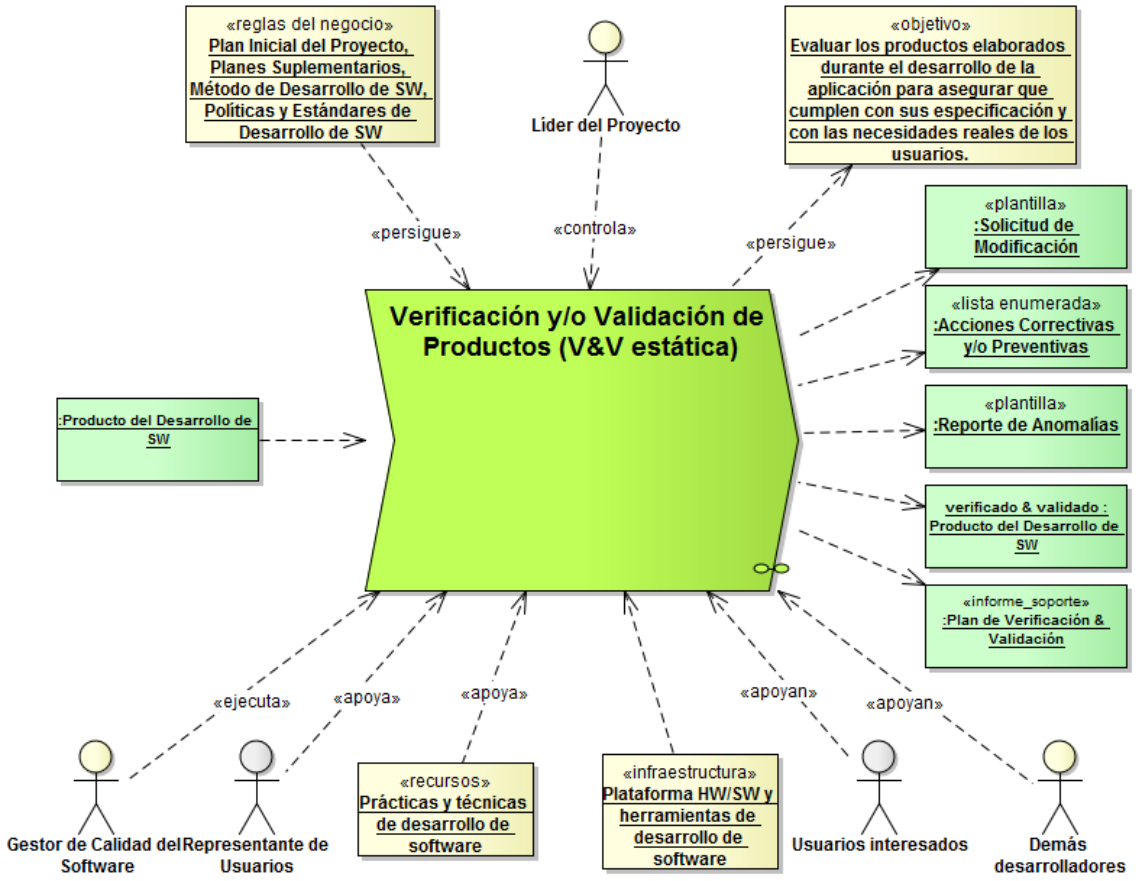


Figura 68. Diagrama de verificación y validación estática de productos.
Fuente: Montilva y Barrios, 2014.

Las reglas de negocio que regularon el proceso de Verificación y Validación de Productos estaban comprendidas por Plan Inicial del Proyecto, los Planes Suplementarios, el Método de Desarrollo de Software *Blue Watch*, las Políticas y Estándares de Desarrollo de Software.

La Verificación y Validación de Productos fue controlada por la líder del proyecto, ejecutado por el gestor de la calidad del software, apoyada por los demás desarrolladores, los usuarios interesados y los representantes de los

usuarios; dicho proceso contó con una infraestructura en la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

El objetivo obtenido fue la evaluación de los productos elaborados durante el desarrollo de la aplicación para asegurar que cumplieran con sus especificaciones y con las necesidades reales de los usuarios.

3.1.17. Gestión de Configuración de Software.

La Gestión de la Configuración del Software fue un proceso de soporte mediante el cual se manejaron los cambios a los diferentes productos que determinaron la configuración de la aplicación. La configuración de la aplicación abarcó un conjunto de productos intermedios o finales que la describieron y fueron sometidos a un control de cambios.

Los productos que se sometieron a la gestión de configuración se denominaron items de configuración. Cada item de configuración se consideró como un conjunto de productos elaborados durante el desarrollo de la aplicación, que fue tratado como una entidad simple durante la gestión de la configuración. Entre los ítems se consideraron los diferentes planes establecidos en la metodología, los modelos de procesos UML, los requisitos, el diseño de las estructuras de datos, los diseños arquitectónicos, el código fuente, la documentación, y los archivos de datos.

Una vez que un ítem es verificado y validado, se almacenó en un repositorio denominado Librería de Gestión de Configuración y, a partir de ese momento, cualquier cambio que se iba generando en dicho item fue controlado debidamente.

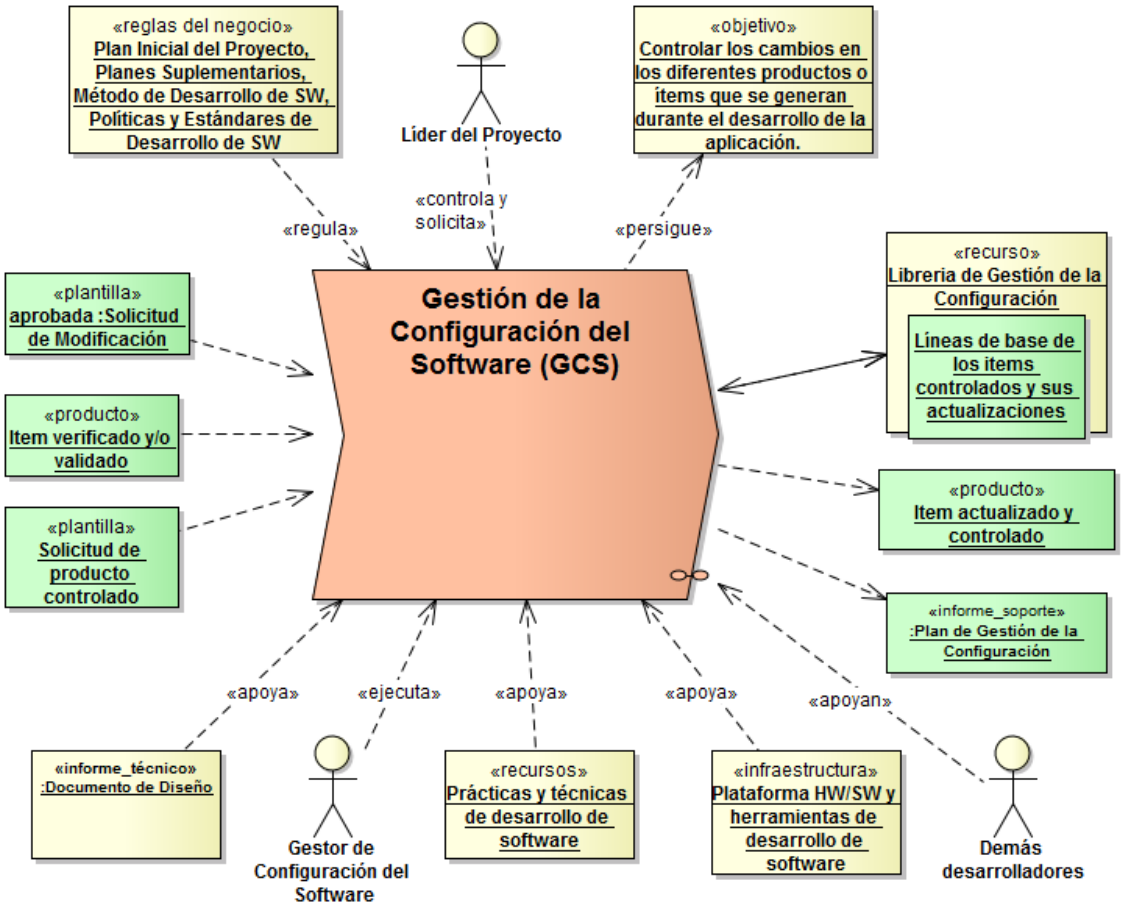


Figura 69. Diagrama de gestión de la configuración del software.
Fuente: Montilva y Barrios, 2014.

Las reglas de negocio que regularon el proceso de Gestión de la Configuración del Software estaban comprendidas por Plan Inicial del Proyecto, los Planes Suplementarios, el Método de Desarrollo de Software *Blue Watch*, las Políticas y Estándares de Desarrollo de Software.

La Gestión de la Configuración del Software fue solicitada y controlada por la líder del proyecto, ejecutado por el gestor de configuración del software, apoyada por los demás desarrolladores; dicho proceso contó con una infraestructura en la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software

acordes a Softour.

El objetivo obtenido fue controlar los cambios en los diferentes productos o ítems que se generaron durante el desarrollo de la aplicación.

3.1.18. Gestión de Riesgos.

La Gestión de Riesgos fue un proceso de soporte mediante el cual se gestionaron aquellos eventos o sucesos que afectaron el normal desenvolvimiento del proyecto de desarrollo de la aplicación Softour.

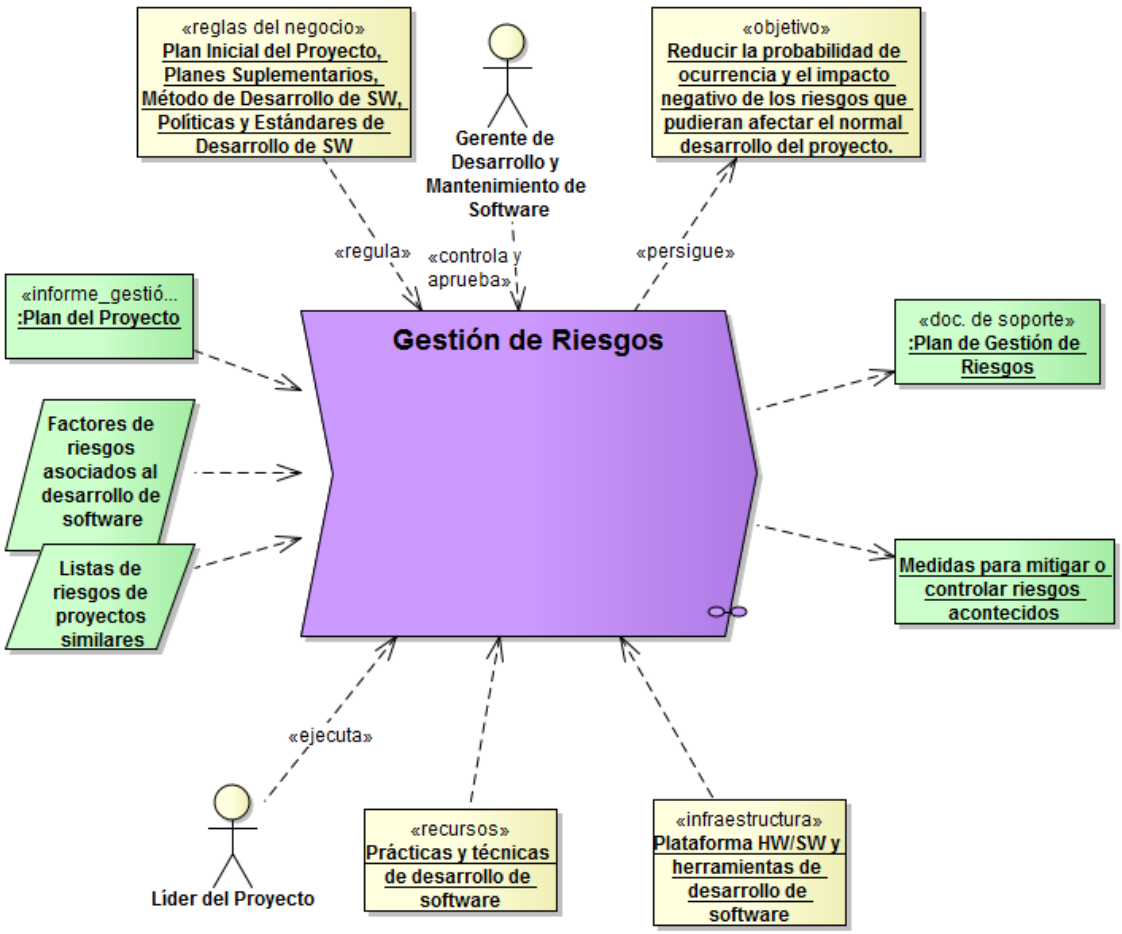


Figura 70. Diagrama de gestión de riesgos.
Fuente: Montilva y Barrios, 2014.

En este proceso se identificaron, clasificaron y analizaron todos aquellos riesgos que afectaban el proyecto. Para aquellos riesgos que ocasionaron el mayor impacto o tuvieron la mayor probabilidad de ocurrencia, se elaboró un plan de contingencia o de respuesta que describió las acciones que la Líder del Proyecto tomaría para prevenir, mitigar o controlar los riesgos que acontecerían.

Las reglas de negocio que regularon el proceso de Gestión de Riesgos estaban comprendidas por Plan Inicial del Proyecto, los Planes Suplementarios, el Método de Desarrollo de Software *Blue Watch*, las Políticas y Estándares de Desarrollo de Software.

La Gestión de Riesgos fue controlada y aprobada por el gerente de desarrollo y mantenimiento de software y ejecutada por la líder del proyecto; dicho proceso contó con una infraestructura en la plataforma de hardware y software seleccionados para utilizar las herramientas y aplicar prácticas y técnicas de desarrollo de software acordes a Softour.

El objetivo obtenido fue reducir la probabilidad de ocurrencia y el impacto negativo de los riesgos que podían afectar el normal desarrollo del proyecto.

Por ser *Blue WATCH* una metodología iterativa debe contemplar procesos balanceados que permitan obtener un proceso de calidad, uno de dichos procesos es el plan de gestión de riesgos, el cual permite identificar, analizar y valorar los posibles riesgos que pudiesen afectar el desenvolvimiento del proyecto.

Basándose en experiencias anteriores se realizó una identificación de los posibles riesgos y se establecieron planes de contingencia. A continuación, se presentan los posibles riesgos:

Cuerpo del producto: riesgo asociado con los elementos que integrarán y darán un tamaño al software a desarrollar.

Impacto en el negocio: riego asociado con las restricciones del mercado.

Características del cliente: riesgo asociado con lo diferente que puede ser cada cliente al que se pretende llegue el software.

Definición del proceso: riesgo asociado con el proceso metodológico y el seguimiento que se le dé.

Entorno de desarrollo: riesgo asociado con la disponibilidad y la calidad de las herramientas que se utilizarán en la construcción del producto.

Tecnología que construir: riesgos asociados con la complejidad del sistema que se construirá y la novedad de la tecnología que está involucrada en el sistema.

Ambiente de desarrollo: riesgos asociados a todo lo que pueda influir al entorno de los desarrolladores.

Tabla 9. Riesgos identificados del proyecto Softour.

ID	Riesgos
R1	Retrasos en un proceso, tarea o función ocasiona retrasos en el desarrollo iterativo.
R2	El tiempo de aprendizaje de una nueva herramienta es más larga de lo programado.
R3	Aparecen requisitos extras.
R4	Los requisitos no han sido comprendidos por el equipo de programación.
R5	Lograr las metas del proyecto requiere más tiempo del planteado.
R6	Perdida accidental de información del proyecto.
R7	Pérdida de tiempo por razones sociales.

Después de identificar estos riesgos se analizó obteniendo el impacto y la probabilidad de ocurrencia que pudiesen tener en el desarrollo del proyecto. El análisis se realizó basado en datos tomados de la experiencia del equipo desarrollador tomando en cuenta factores ambientales, de tiempo y económicos. A continuación, se muestra mediante una tabla, el análisis de

forma más específica:

Tabla 10. Análisis de riesgos.

ID	Probabilidad	Impacto	Estrategia de mitigación	Plan de contingencia
R1	40%	2	Establecer días de holgura en planificación donde se completen los procesos rezagados.	Reajustar el proceso del cronograma.
R2	30%	3	Tener documentación precisa y contacto con expertos que puedan aportar asesoría.	Consultar a los expertos.
R3	40%	4	Identificar claramente si los nuevos requisitos son funcionales o no.	Reajustar el cronograma si es necesario agregar el nuevo requisito.
R4	40%	4	Mantener reuniones constantes donde cada miembro del equipo comprenda los elementos que integrarán el sistema.	Sesiones aclaratorias de trabajo diarias donde se integren a los miembros que no identifiquen los requisitos
R5	40%	3	Plantear los módulos que son elementales para el sistema y ponerlos como prioridad.	Establecer un orden de los requisitos principales a programar y reajustar el cronograma.
R6	20%	3	Mantener un control con la manipulación de la información, sus iteraciones y resultados.	Buscar respaldos y planificar actividades.
R7	10%	3	Poseer diferentes localidades de trabajo y establecer horas de trabajos diferentes para evadir posibles eventos sociales.	Convocar reuniones extras del equipo para recuperar tiempo perdido.

Los valores de impacto seleccionados para esta tabla son:

Catastrófico (1); Crítico (2); Marginal (3); Despreciable (4).

3.1.19. Procesos de Gestión.

Los procesos de gestión encargados de planificar e iniciar el desarrollo de los incrementos son: la Planificación de Entregas y la Planificación de la Iteración i-ésima. Estos procesos agregan, al Plan del Proyecto, los siguientes informes: Plan de Entregas, Planes de las Iteraciones y el Plan de Pruebas.

El desarrollo de los incrementos de la aplicación representó un conjunto de procesos iterativos, con una determinada de iteraciones, las cuales se iniciaron con el Refinamiento de Requisitos del Incremento y continuaron con los procesos de Diseño Detallado del Incremento, Construcción del Incremento y Pruebas del Incremento. Estos procesos actualizaron los informes técnicos respectivos.

Una vez que el incremento fue codificado y probado, este se sometió al proceso de Verificación y Validación, el cual cierra la i-ésima iteración, con la entrega del incremento a los usuarios para la realización de pruebas de aceptación o una mayor validación. En algunos casos, el incremento entró nuevamente a producción.

Una vez producido el último incremento, se procede a la Entrega Final de la Aplicación y al Cierre del Proyecto; procesos con los cuales culmina el desarrollo de la aplicación.

En paralelo a los procesos descritos anteriormente, se lleva a cabo un conjunto de procesos transversales que dan soporte a la gestión del proyecto; estos son: Gestión de Requisitos, Verificación y Validación de Productos, Gestión de la Configuración del Software.

Cabe destacar que para la iniciación de los procesos y actividades pertenecientes a este ciclo se realizó la planificación del sistema Web, estableciendo un conjunto de acciones para determinar la gestión administrativa del sistema. Una vez determinados los procesos con cada uno de sus

productos, durante la instanciación de la metodología *Watch* se pudo definir los productos pertenecientes a los procesos de gestión, soporte y técnicos.

La Gestión de un Proyecto de Desarrollo de Software consta de un conjunto de grupos de procesos que se llevaron a cabo a lo largo del desarrollo del software. Estos grupos de procesos son: Inicio del Proyecto, Planificación del Proyecto, Dirección del Proyecto, Seguimiento y Control del Proyecto, y Cierre del Proyecto.

Figura 71. Diagrama de gestión del proyecto de desarrollo de software. Fuente: Montilva y Barrios, 2014.

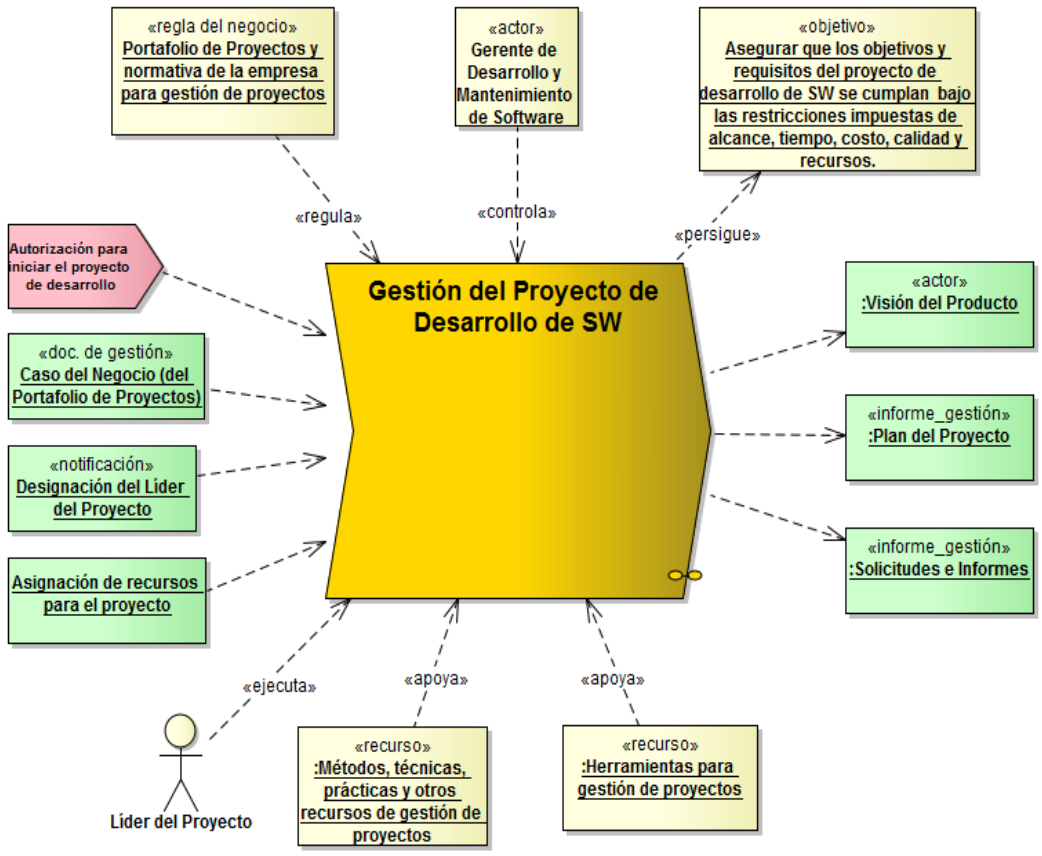


Figura 72. Diagrama de gestión del proyecto de desarrollo de software. Fuente: Montilva y Barrios, 2014.

Las reglas de negocio que regularon el proceso de Gestión del Proyecto de Desarrollo de Software fueron el Portafolio de Proyectos y las normativas de la Licenciatura en Informática para la gestión de proyectos en el marco de las alternativas de grado.

El proceso de gestión de Gestión del Proyecto de Desarrollo de Software fue controlado por el Gerente de Desarrollo y Mantenimiento de Software representado por los profesores responsables de dirigir las alternativas de grado, contando en los procesos de ejecución con la líder del proyecto; además dicho proceso se apoyó en herramientas para gestión de proyectos y utilizó métodos, técnicas, prácticas y otros recursos de gestión de proyectos.

El objetivo finalmente obtenido fue asegurar que los objetivos y requisitos del proyecto de desarrollo de software se cumplieron bajo las restricciones impuestas de alcance, tiempo, costo, calidad y recursos.

3.1.20. Ejemplos de Ciclo de Versión.

En el primer ciclo de versión, se realizaron algunos cambios de requisitos para enriquecer y darle mayor funcionalidad a la aplicación, a medida que se iban desarrollando y desplegando los incrementos se fueron generando nuevos requisitos o modificaciones de los requisitos existentes, enriqueciendo el análisis del modelo del negocio. El ciclo completo de iteraciones se muestra en el Apéndice.

3.1.20.1. Refinamiento de la arquitectura del software.

Se detallaron y especificaron los casos de uso por el equipo de trabajo y se anexaron los cambios en el flujo normal de eventos y en el flujo alternativo a medida que se iba enriqueciendo la visión del proyecto.

3.1.20.2. Diseño detallado de la primera versión.

En este punto el método *Blue Watch* permitió establecer el trabajo orientado al desarrollo de tres subprocesos, los cuales son: diseño de interfaz de usuario y de sistema, diseño de base de datos y el diseño de componentes. A continuación, se definen y se muestran sus resultados.

3.1.20.3. Diseño de interfaz.

En este punto se diseña la secuencia de interfaz de la iteración del perfil administrador relacionada con “Agregar Lugar” la cual se comunicará con el usuario, estableciendo, métodos, metáforas, pantallas, menús y la posición en la cual irán estos elementos en el sistema.

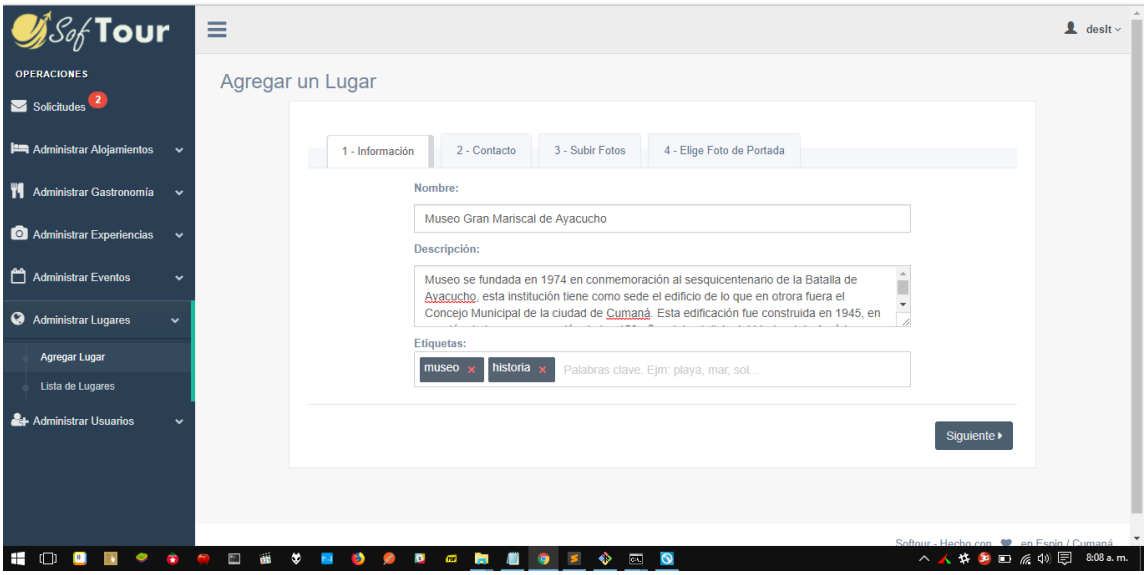


Figura 73. Interfaz perfil Administrador: Vista general Agregar Lugar.

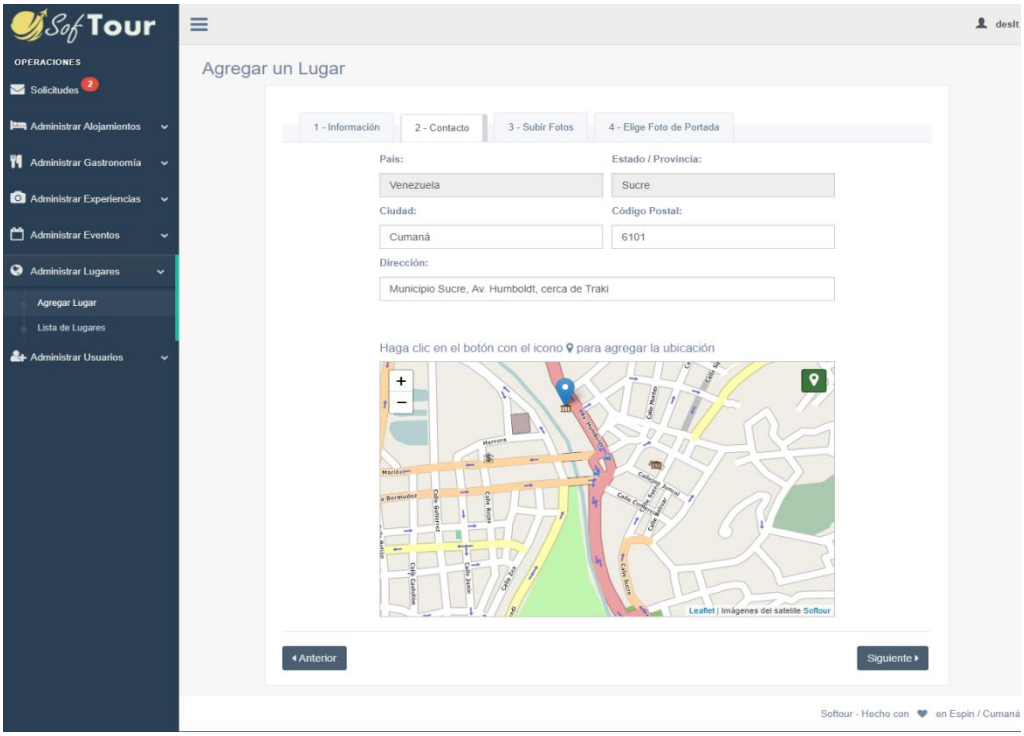


Figura 74. Interfaz perfil Administrador: Vista Agregar Lugar con Posicionamiento en el mapa geográfico satelital.

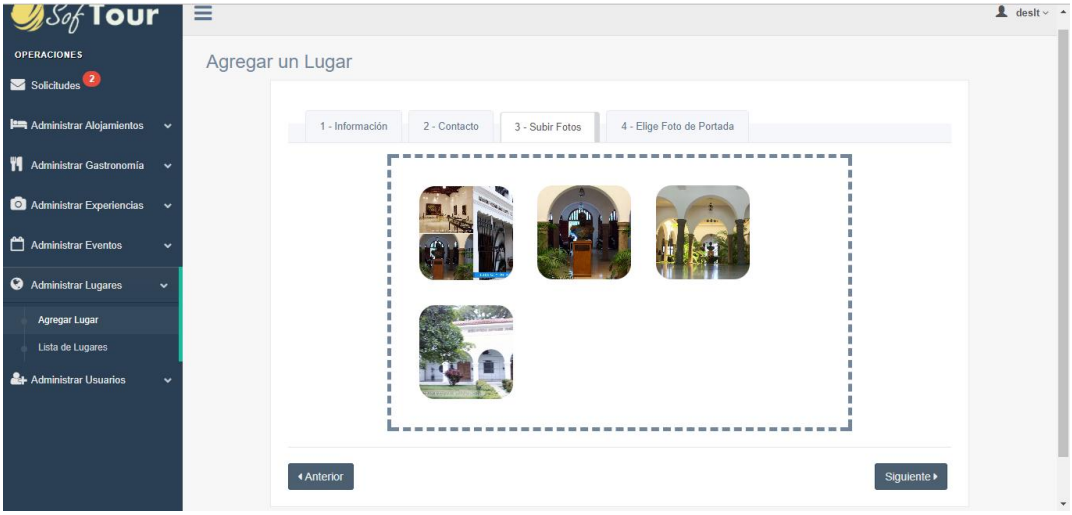


Figura 75. Interfaz perfil Administrador: Vista Agregar Lugar Subir Foto.

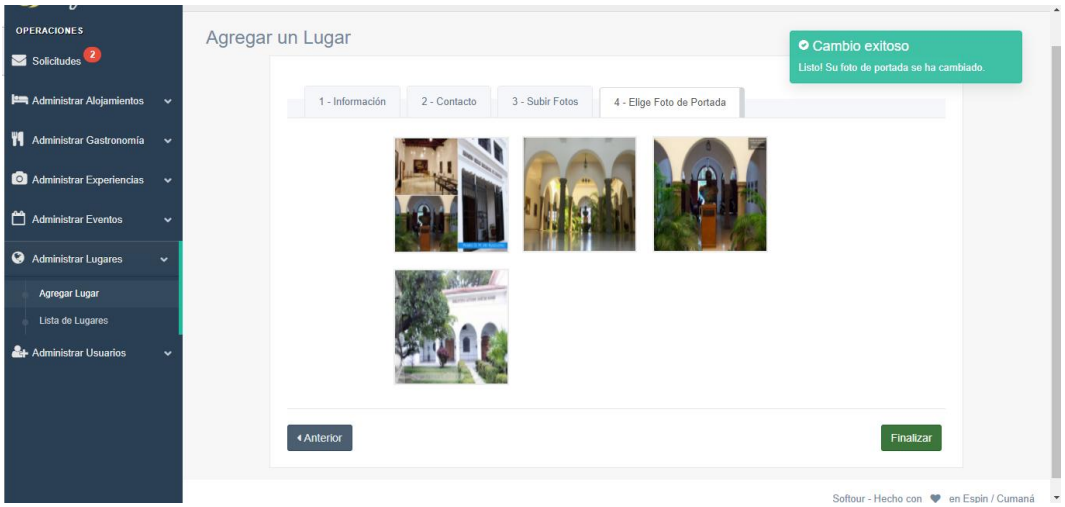


Figura 76. Interfaz perfil Administrador: Vista Agregar Lugar Subir Foto de Portada.

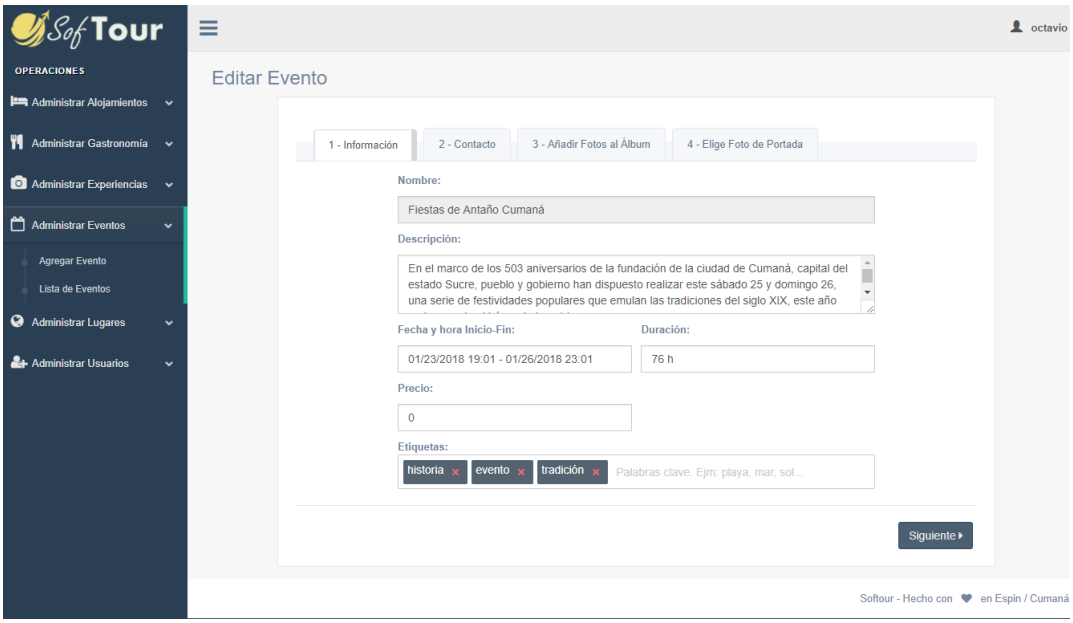


Figura 77. Interfaz perfil Prestador de Servicios: Vista general Editar Evento.

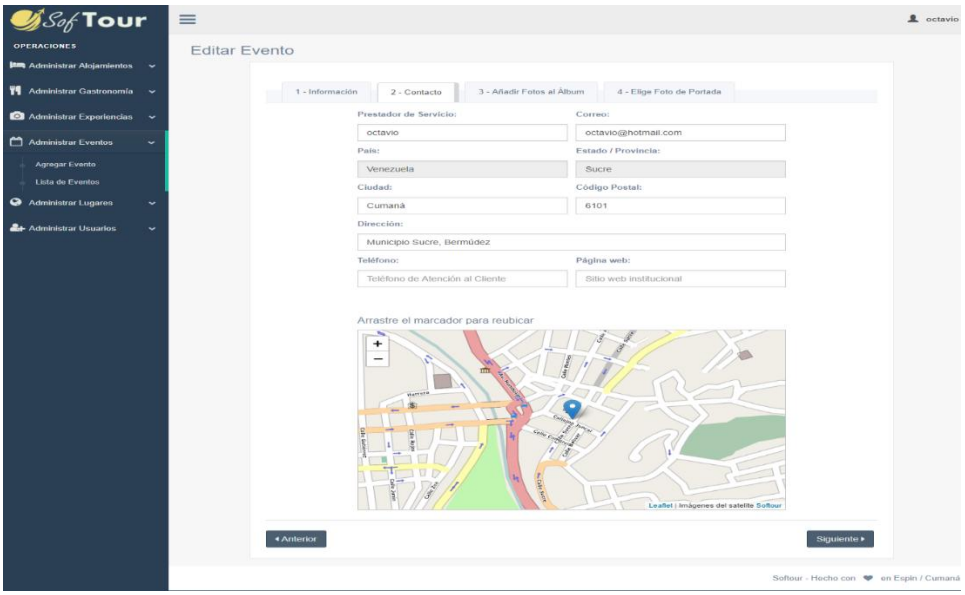


Figura 78. Interfaz perfil Prestador de Servicios: Vista Editar Evento para modificar datos de persona contacto.

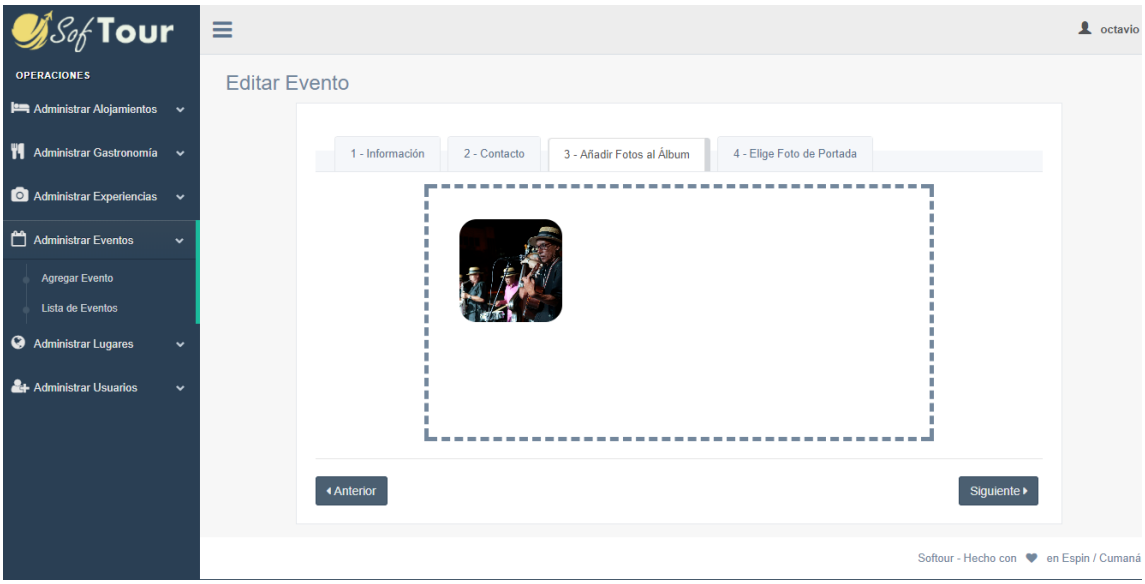


Figura 79. Interfaz perfil Prestador de Servicios: Vista Editar Evento para añadir nueva foto de álbum.

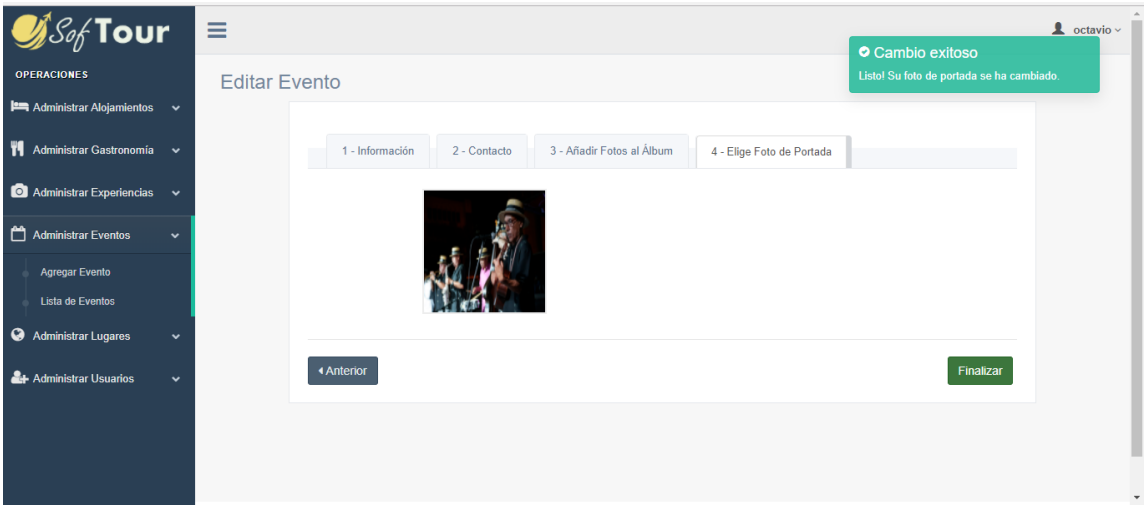


Figura 80. Interfaz perfil Prestador de Servicios: Vista Editar Evento para elegir nueva foto de portada.

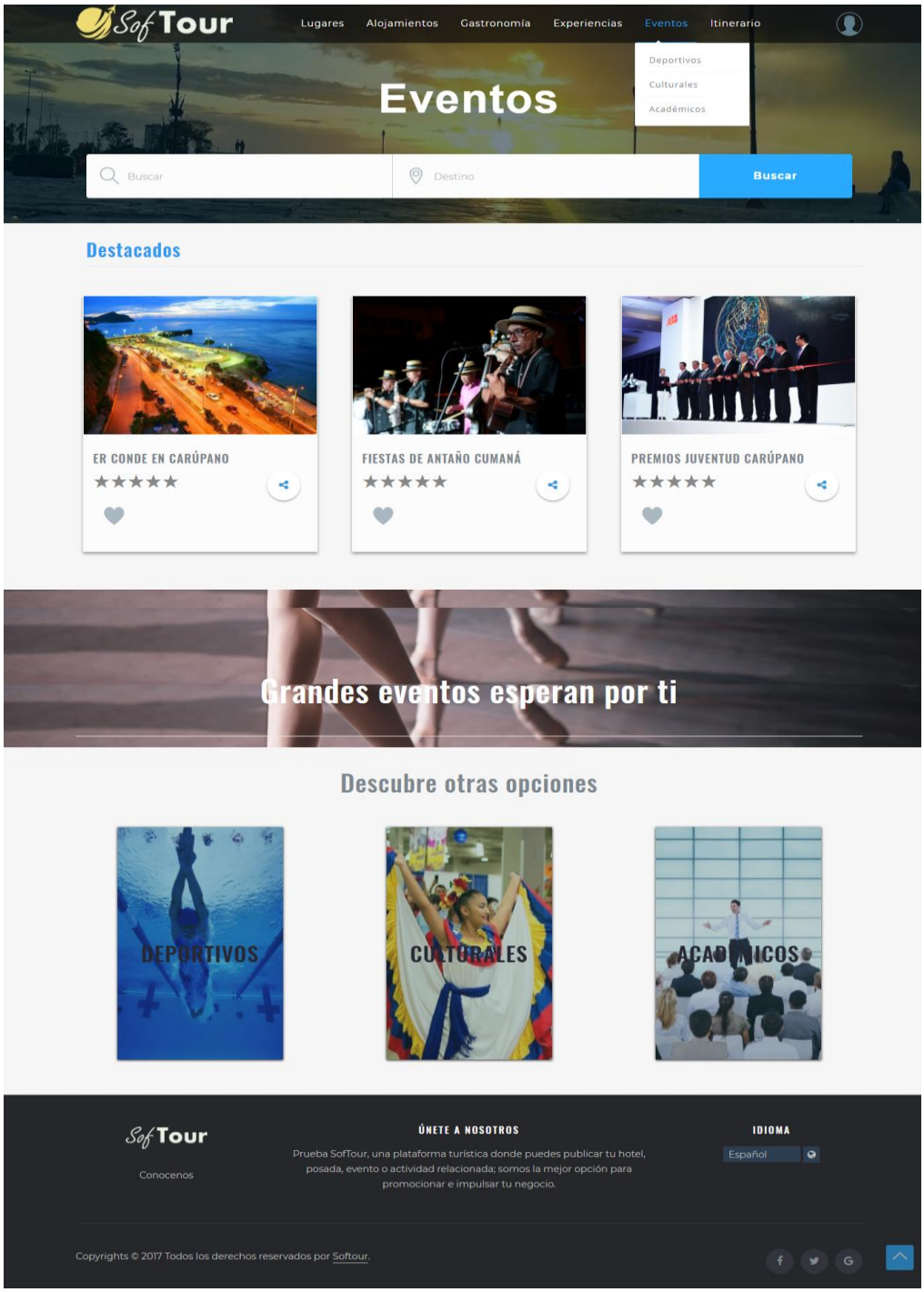


Figura 81. Interfaz perfil Turista: Vista principal de Consultar Eventos.

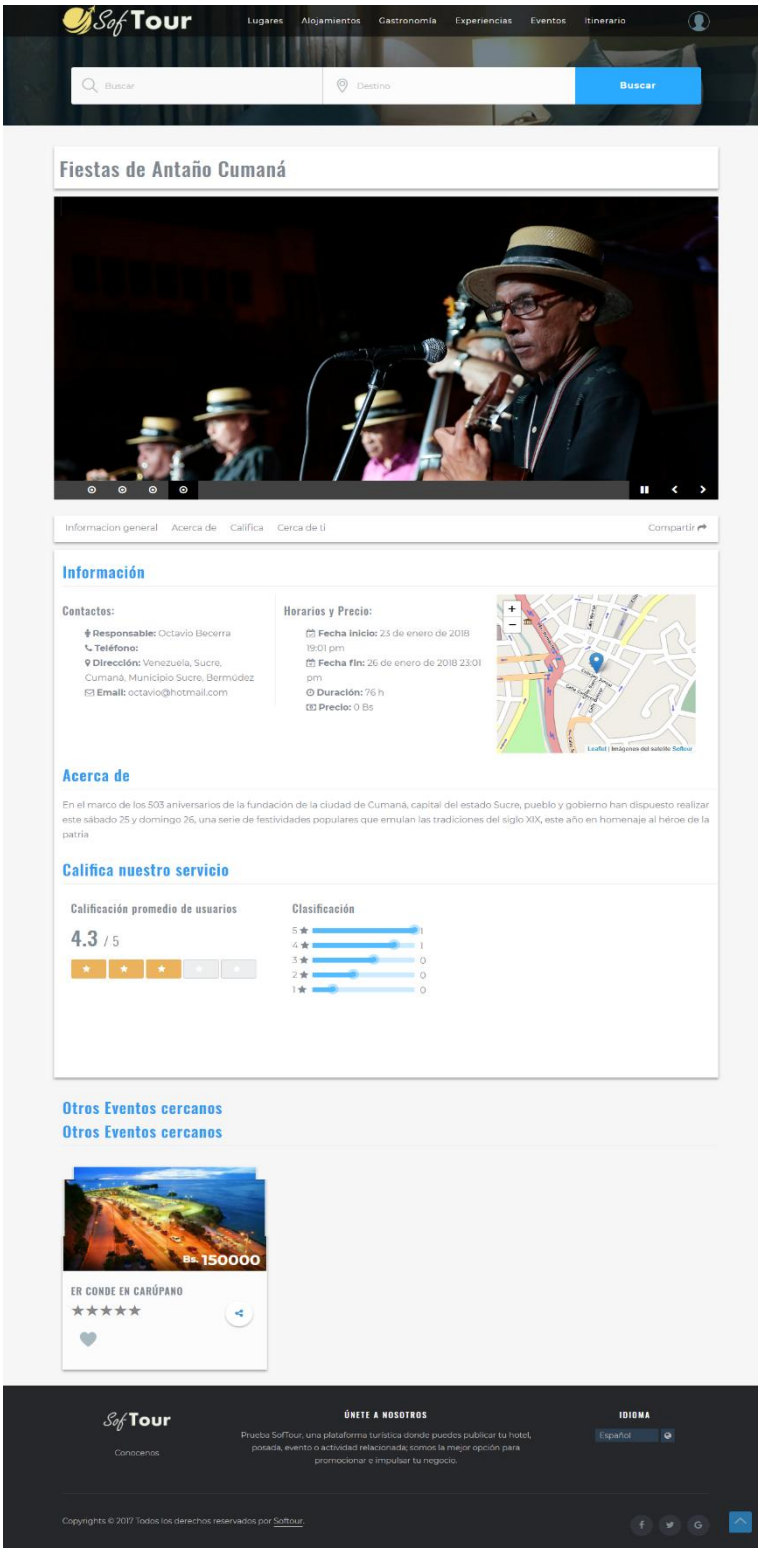


Figura 82. Interfaz perfil Turista: Vista detalle de Consultar Evento.

3.1.20.4. Diseño de base de datos.

Este subproceso buscó refinar de la manera más detallada posible los componentes de la base de datos y las relaciones que dichos componentes tenían entre sí. La base de datos fue modificándose progresivamente a medida que se iban incorporando nuevos incrementos que establecieron nuevos requerimientos de almacenamiento de colecciones de datos.

3.1.20.5. Diseño detallado de componentes.

El diseño detallado de componentes muestra la interacción que existe entre los elementos de la interfaz y los elementos de la base de datos, estos se definen mediante los diagramas de secuencia, los cuales fueron diseñados siguiendo lo establecido en los flujos normales y los flujos alternativos de eventos de cada Caso de Uso. Se elaboró un Diagrama de Secuencia por cada Caso de Uso.

3.1.20.6. Programación e integración de las iteraciones de la aplicación.

La programación e integración tuvo como norte desarrollar los requerimientos, para que el sistema realizara sus funciones. Los elementos principales fueron las funciones del sistema y la base de datos, mediante este proceso se buscó la integración de ambos para que cumpliera con los requerimientos planteados.

3.1.20.7. Gestión de Pruebas.

La calidad de un producto puede ser medida de manera subjetiva, ya que cada usuario o desarrollador posee sus estándares de calidad, pero a nivel informático todo sistema debe superar un proceso de pruebas que lo califica como de calidad o no.

Al momento de desarrollar un software es necesario contemplar el rol de experto en pruebas, quien será el encargado de llevar a cabo ciertas técnicas y evaluaciones que pondrán bajo rigor la funcionalidad del sistema. Este proceso consiste en validar y verificar que cada proceso cumpla su función y de no ser

así documentarlo y hacerlo saber al equipo de desarrollo para solventar el error antes de que el software salga al público.

SofTour por ser un sistema que se espera sea manejado por un sinfín de usuarios se sometió a un riguroso proceso de pruebas, en donde se tomaron en cuenta pruebas de funcionabilidad, pruebas de usabilidad y pruebas de integración de componentes, de manera general se realizaron pruebas de navegación y de contenido.

Pruebas de funcionalidad

Comúnmente en el desarrollo de software los componentes que integran una función se encuentran desarrollados en diferentes herramientas de desarrollo, por ende, al momento de integrar los componentes se pueden presentar ciertas fallas o errores, por lo tanto, se deben llevar a cabo pruebas que saquen a la luz las posibles fallas existentes.

La técnica que se utilizó para llevar a cabo las pruebas funcionales fue la denominada como prueba de caja negra, considerando que es la técnica que mejor se adapta para validar el cumplimiento de la función evaluada.

La prueba de caja negra, es una técnica de pruebas de software en la cual la funcionalidad se verifica sin tomar en cuenta la estructura interna de código, detalles de implementación o escenarios de ejecución internos en el software.

En las pruebas de caja negra, nos enfocamos solamente en las entradas y salidas del sistema, sin preocuparnos en tener conocimiento de la estructura interna del programa de software. Para obtener el detalle de cuáles deben ser esas entradas y salidas, se basa únicamente en los requerimientos de software y especificaciones funcionales.

Muchos autores consideran que estas pruebas permiten encontrar:

Funciones incorrectas o ausentes

Errores de interfaz

Errores en estructuras de datos o en accesos a las Bases de Datos externas

Errores de rendimiento

Errores de inicialización y terminación

Para preparar los casos de pruebas hacen falta un número de datos que ayuden a la ejecución de estos casos y que permitan que el sistema se ejecute en todas sus variantes, pueden ser datos válidos o inválidos para el programa según si lo que se desea es hallar un error o probar una funcionalidad. Los datos se escogen atendiendo a las especificaciones del problema, sin importar los detalles internos, a fin de verificar que el programa corra bien.

Para confeccionar los casos de prueba de caja negra existen distintos criterios. Algunos de ellos son:

Particiones de Equivalencia

Análisis de Valores Límite

Métodos Basados en Grafos

Pruebas de Comparación

Análisis Causa-Efecto

Las pruebas realizadas al sistema se basan en la partición de equivalencias, que se define como un método de prueba que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. La partición equivalente se dirige a una definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar. En otras palabras, este método intenta dividir el dominio de entrada de un programa en un número finito de clases de equivalencia. De tal modo que se pueda asumir razonablemente que una

prueba realizada con un valor representativo de cada clase es equivalente a una prueba realizada con cualquier otro valor de dicha clase.

Esto quiere decir que, si el caso de prueba correspondiente a una clase de equivalencia detecta un error, el resto de los casos de prueba de dicha clase de equivalencia deben detectar el mismo error. Y viceversa, si un caso de prueba no ha detectado ningún error, es de esperar que ninguno de los casos de prueba correspondientes a la misma clase de equivalencia encuentre ningún error.

El diseño de casos de prueba según esta técnica consta de dos pasos:

Identificar las clases de equivalencia

Las clases de equivalencia se identifican examinando cada condición de entrada y dividiéndola en dos o más grupos. Se definen dos tipos de clases de equivalencia, las clases de equivalencia válidas, que representan entradas válidas al programa, y las clases de equivalencia no válidas, que representan valores de entrada erróneos.

Identificar los casos de prueba

El objetivo es minimizar el número de casos de prueba, así cada caso de prueba debe considerar tantas condiciones de entrada como sea posible. No obstante, es necesario realizar con cierto cuidado los casos de prueba de manera que no se enmascaren faltas. Así, para crear los casos de prueba a partir de las clases de equivalencia se han de seguir los siguientes pasos:

Asignar a cada clase de equivalencia un número único

Hasta que todas las clases de equivalencia hayan sido cubiertas por los casos de prueba, se tratará de escribir un caso que cubra tantas clases válidas no incorporadas como sea posible

Hasta que todas las clases de equivalencia no válidas hayan sido cubiertas por

casos de prueba, escribir un caso para cubrir una única clase no válida no cubierta.

Los casos de prueba a utilizar se obtendrán a partir de los casos de usos permitiendo de esta manera: Verificar el resultado de la interacción entre los actores y el sistema, comprobar que se satisfagan las precondiciones y postcondiciones del caso de uso y por último comprobar que se siga la secuencia de acciones especificado por el caso de uso.

Prueba de interfaz

En este punto se busca verificar que la interfaz cumpla con los estándares de heurística establecidos por Jakob Nielsen, los cuales contemplan que mediante su aplicación se logrará obtener una interfaz usable. Mediante esta evaluación se validarán diez puntos y de acuerdo a la ponderación obtenida se podrá deducir si se tiene una interfaz usable o no. A continuación se presenta las diez mejores prácticas para lograr una aplicación con interfaz usable:

Visibilidad del estatus del sistema, el sistema siempre debe mantener informado al usuario de lo que está ocurriendo, a través de retroalimentación apropiada dentro de un tiempo razonable.

Consistencia entre el sistema y el mundo real, el sistema debe hablar el lenguaje de los usuarios mediante palabras, frases y conceptos que sean familiares al usuario, más que con términos relacionados con el sistema. Seguir las convenciones del mundo real, haciendo que la información aparezca en un orden natural y lógico.

El usuario es libre y tiene el control, es la salida de emergencia que tiene el usuario al momento de seleccionar un ítem no deseado, el usuario tiene el poder de salir de un área de forma rápida sin tener que seguir una serie de pasos.

Consistencia y estándares, el sistema debe poseer una consistencia entre sus componentes, no debe existir diferencia de contexto en la aplicación y debe seguir los estándares utilizados que generalmente representan ciertos iconos.

Prevención de errores, un buen diseño dirige al usuario a cometer menos errores y esto se logra con la implementación de herramientas como el autocompletado de ciertos navegadores o las sugerencias al momento de realizar búsquedas.

Mejor reconocer que memorizar, los elementos de la aplicación deben ser claramente visibles e inequívocos, los objetos, acciones y opciones deben estar bien definidos para que el usuario los reconozca a primera instancia.

Flexibilidad y eficiencia de uso, el sistema debe ser diseñado para ser fácilmente utilizado por usuarios expertos o novatos.

Diseño estético y minimalista, los componentes no deben contener información que es irrelevante. Cada unidad extra de información en un diálogo, compite con las unidades de información relevante y disminuye su visibilidad relativa.

Ayuda al usuario a reconocer, diagnosticar y recuperarse de los errores, ayuda a los usuarios a reconocer y corregir sus errores, indica siempre el problema concreto que está ocurriendo y sugiere soluciones constructivas.

Ayuda y documentación, a pesar de que con los principios antes mencionados la aplicación deber ser fácil de usar sin manuales, es necesario colocar un manual de usuario fácil de localizar.

Todo sistema que supere esta prueba o cumpla con estos estándares de usabilidad prácticamente tiene el éxito asegurado, ya que estas medidas se basan en años de estudios en el comportamiento de usuarios y expertos.

3.2. Ciclo de versiones

Todo proyecto de software desarrollado bajo el Método Watch esta regido por versiones y, a su vez, estas deben estar compuestas por iteraciones, este proceso consiste en dividir el trabajo o el desarrollo del sofware en pequeños módulos para realizar un trabajo práctico, dinámico y funcional.

El desarrollo del software Softour se planteó bajo el Método Blue Watch, por ende el equipo de trabajo dividió las funcionalidades planteadas comenzando por el diseño y codificacion de las tareas del usuario Administrador, como por ejemplo los Casos de Uso agregar, modificar, visualizar y eliminar lugares, entre otros apartados planteados en el sistema, por tal motivo se realizó una iteración en donde se especifica cada proceso que conlleve una funcionalidad.

Se trabajo a traves de versiones las cuales fueron dividida por administrador, prestador de servicio y turistas, los usuarios que se manejan en dicho sistema, conformadas cada una por una serie de iteraciones, a continuación se mostrara una iteracion por cada una de las versiones.

Caso de Uso “Ingresar Lugar” por el usuario administrador, definiéndolo como la primera iteración de la primera versión.

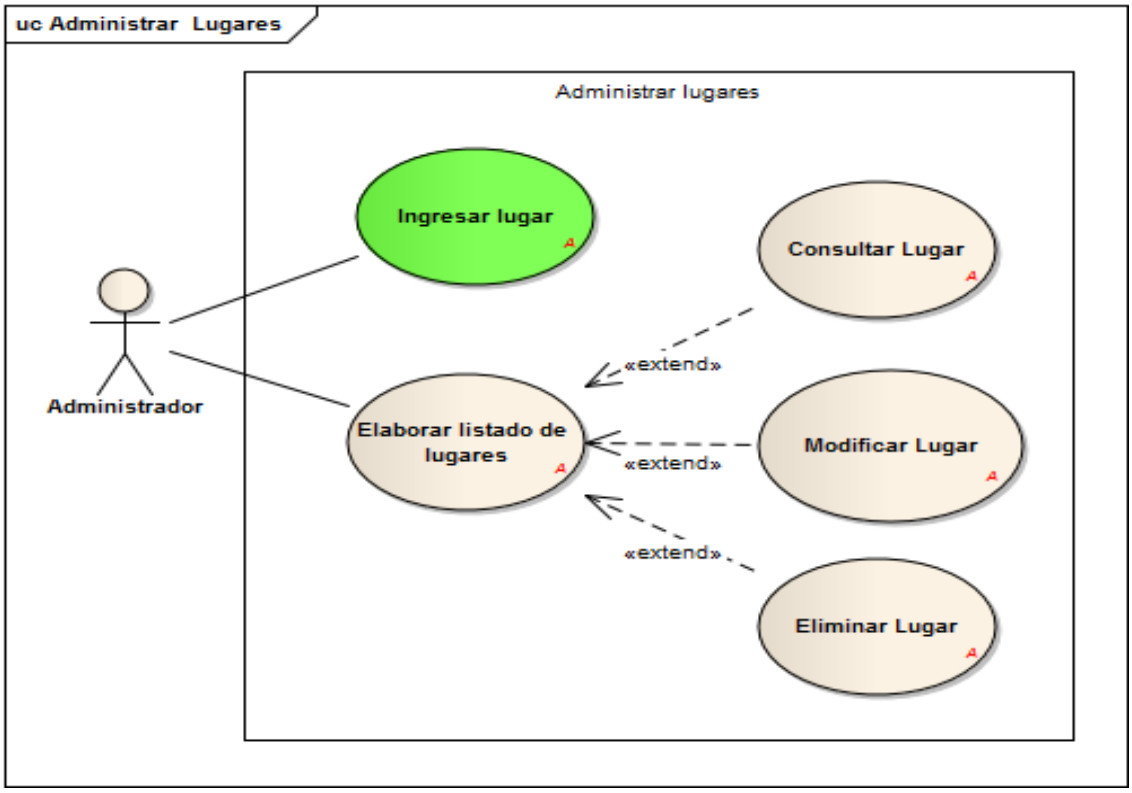


Figura 83. Caso de uso: ingresar lugar por usuario Administrador

Los casos de uso son soportados por una descripción, en donde se especifican el proceso de la funcionalidad, el nombre, el flujo normal de eventos y el flujo alternativo de eventos, además de las condiciones que interfieren en este. Cada caso de uso cuenta con su descripción como se muestra en la Tabla 11.

Tabla 11. Descripción Caso de uso: Ingresar Lugar por usuario Administrador

Nombre del Caso de uso	Ingresar lugar.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en ingresar los atributos del lugar de manera satisfactoria.	
Precondición	Para ingresar un lugar, no debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Ingresar lugar".
	2	El sistema muestra el formulario correspondiente.
	3	El Administrador ingresar los atributos del lugar.
	4	El sistema almacena los atributos del lugar y finaliza el caso de uso.
Postcondición	Se ingresa los atributos del lugar en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no ingresar el lugar y selecciona cancelar el caso de uso, seleccionando otra opción del sistema.
	3.2.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. Para que un lugar pueda ser ingresado no debe estar registrado en la base de datos del sistema.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

La funcionalidad de un Caso de Uso implica la colaboración de objetos asociados entre sí para que la función se ejecute correctamente, estos objetos pueden estar vinculados con diversos tipos de clases: interfaz, control y persistentes. La vista estática se establece en la representación del Diagrama de Clases asociado al Caso de Uso en estudio.

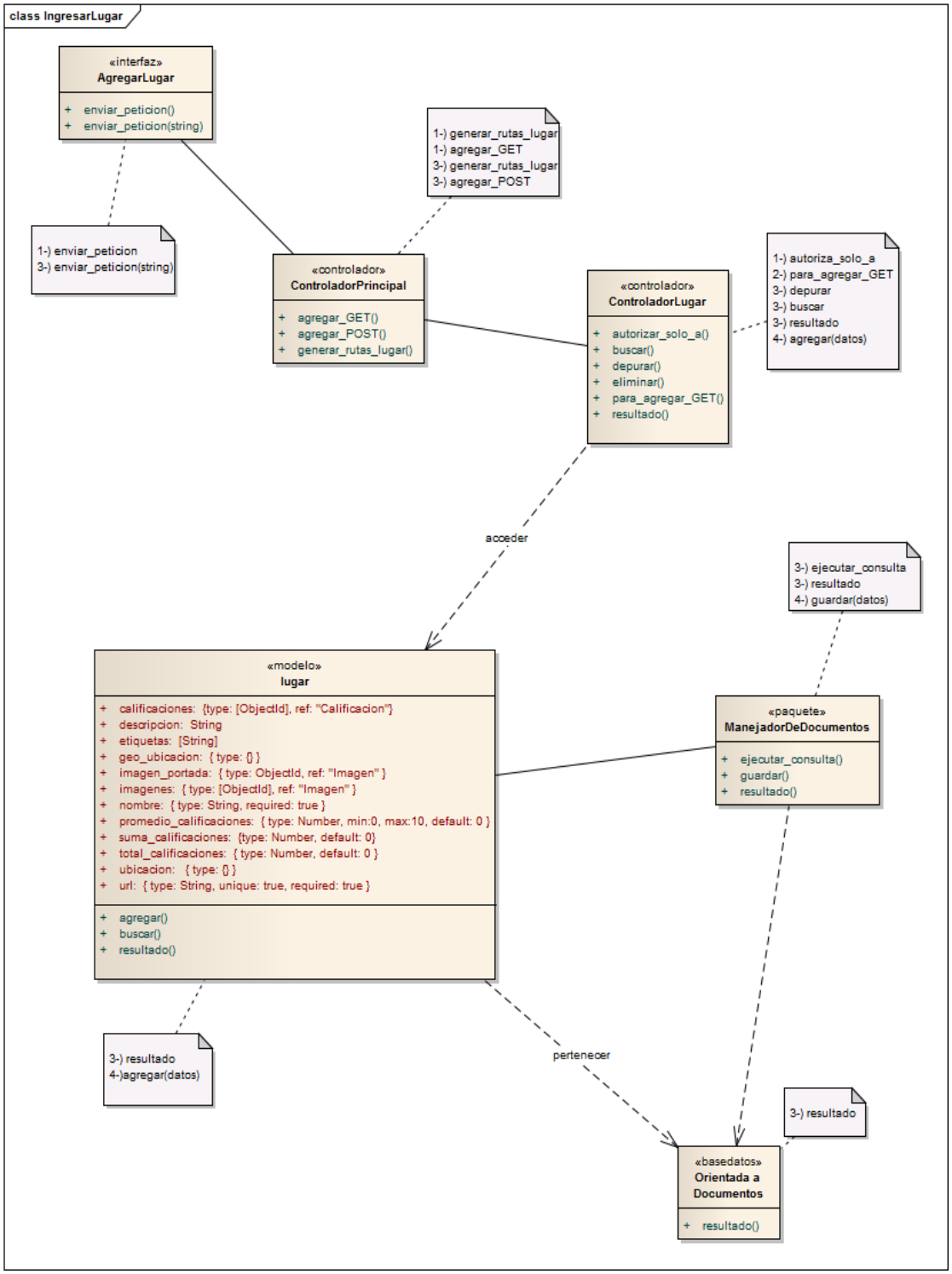


Figura 84. Diagrama de clases: Ingresar Lugar por usuario Administrador.

La interacción entre los objetos del sistema a traves del tiempo se modela según los aspectos dinámicos mostrados en el flujo de eventos del Caso de Uso objeto de estudio con la elaboración del Diagrama de Secuencia correspondiente a dicha funcionalidad, el cual complementa el Diagrama de Clases mostrando detalles de implementación de los escenarios, inlcuyendo a las clases, objetos, métodos, atributos y mensajes.

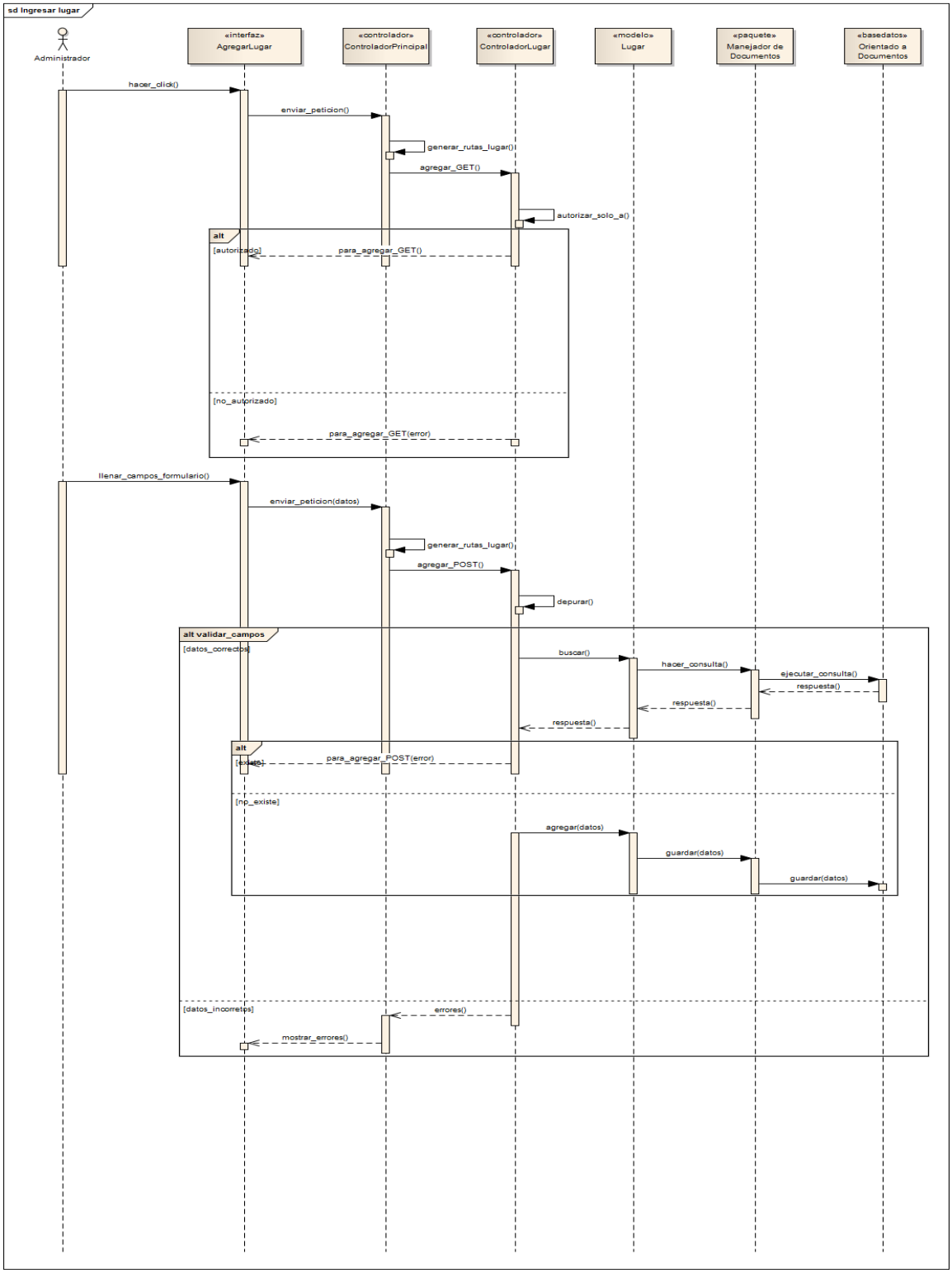


Figura 85. Diagrama de Secuencia: Registrar Lugar por usuario Administrador

El modelado de datos se desarrolló según las definiciones de las clases persistentes planteadas en el Diagrama de Clases, las cuales se organizaron en colecciones de documentos que interactúan según la funcionalidad del Caso de Uso. Se muestra en el diagrama contenido en la Figura 82.

Figura 86.Colección de lugares: ingresar lugar por usuario Administrador.

El diseño de la interfaz de usuario crea un medio eficaz de comunicación entre los seres humanos y la computadora. Siguiendo un conjunto de principios de diseño de la interfaz, se identifica los objetos y acciones para crear una plantilla que constituye la base del prototipo de la interfaz de usuario. Para la elaboración del diseño de interfaz de la aplicación web Softour se siguieron una serie de principios descritos por Tognozzi, resaltando la comunicación, consistencia, autonomía controlada, eficiencia, aprendizaje, lo cual permitió obtener una interfaz consistente, intuitiva, fácil de usar y aprender, eficiente y llamativa. Dichos principios se definen a continuación (Pressman, 2005):

Comunicación. La interfaz debe comunicar el estado de cualquier actividad iniciada por el usuario. En la aplicación Softour, la interfaz le brinda al usuario el contexto necesario para desenvolverse de manera correcta e intuitiva dentro de la aplicación, distinguiendo entre las jerarquías de los diferentes tipos de usuarios.

Consistencia. El uso de controles de navegación, menús, iconos y estética (color, forma y distribución) debe ser consistente entre cada interfaz de usuario. En Softour, el color de los textos se maneja igual para cada una de las vistas, los colores de la interfaz son consistente de acuerdo al tipo de usuario que fue determinado en el análisis. Los iconos utilizados se determinaron para hacer una representación metafórica de ciertas acciones que deben realizar facilitando la comprensión al mismo.

Autonomía controlada. La interfaz debe facilitar el movimiento del usuario a través de la aplicación, pero lo debe hacer de manera que obligue a respetar las convenciones que se hayan establecido para la aplicación. En softour se estableció de acuerdo al perfil del usuario cuales tareas y actividades puede realizar dentro de la aplicación, permitiéndoles a estos que puedan realizar dichas tareas y explorarla más a fondo.

Eficiencia. El diseño de la aplicación y su interfaz deben optimizar la eficiencia del trabajo del usuario, no la del desarrollador que la diseña y construye ni del ambiente cliente-servidor que la ejecuta. La elaboración de la interfaz de la aplicación, Softour se centró en facilitar las herramientas necesarias al usuario para que este pueda realizar las actividades y tareas en un tiempo acorde de forma fluida.

Aprendizaje. Una interfaz de aplicación debe diseñarse para minimizar el tiempo de aprendizaje y, una vez aprendida, minimizar el que se dedique a reaprender cuando se regrese a la aplicación. En general, la interfaz debe hacer énfasis en un diseño sencillo e intuitivo que organice el contenido y

funcionalidad en categorías que resulten obvias para el usuario. La interfaz de la aplicación Softour está diseñada para ser intuitiva, fácil de manejar, permitiendo al usuario realizar tareas básicas la primera vez que este interactúe con dicha aplicación

Una vez definidos los principios de diseño para la realización de la interfaz de la aplicación web Softour, se estableció las tipografías, los distintos tamaños de las tipografías utilizadas, los colores de la interfaz de la aplicación y los diferentes botones, enlaces e iconos que facilitan la interacción entre el usuario y la aplicación. La tipografía que se utilizó es **Montserrat, Oswald** con un tamaño entre 16 pixeles y 20 pixeles, permitiendo una fácil legibilidad al usuario. Los colores elegidos para las tipografías fueron azul, gris y blanco. Para la interfaz se seleccionó la siguiente paleta de colores: blanco para los fondos, lo cual permite centrar la atención del cliente en la información que transmite la aplicación, azul para los distintos botones, negro para el menú de navegación y gris para el pie de página de nuestra aplicación, para la sección de administración se utilizó el color azul para el panel del administrador.

A continuación, se muestran los modelos de plantillas para el proceso de ingresar lugar.

Figura 87. Plantilla de interfaz 1: Ingresar Lugar por usuario administrador.

Figura 88. Plantilla de interfaz 2: Ingresar Lugar por usuario administrador.

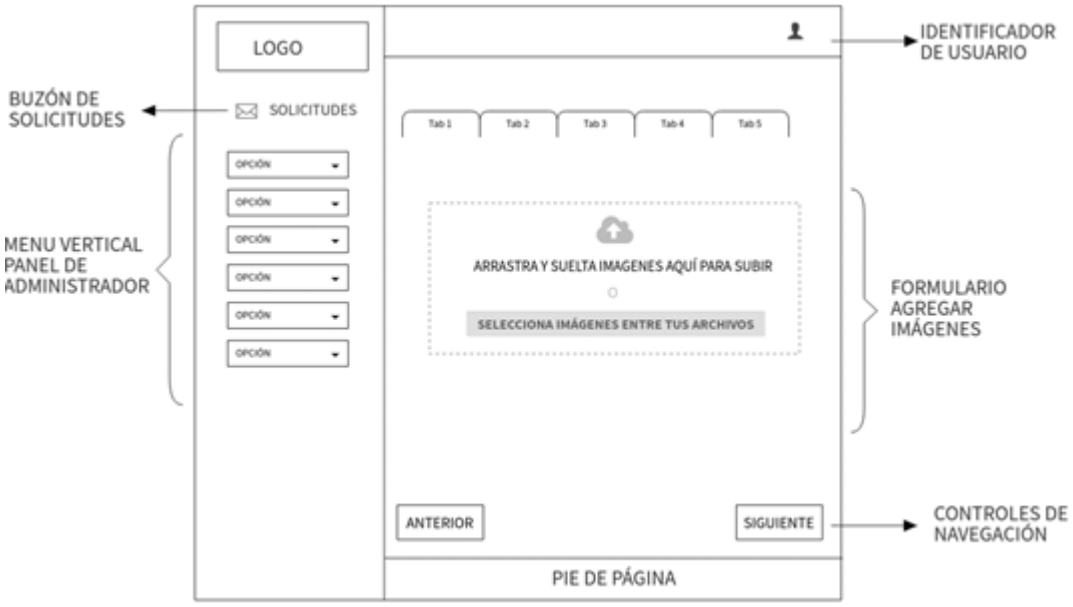


Figura 89. Plantilla de interfaz 3: Ingresar Lugar por usuario administrador.

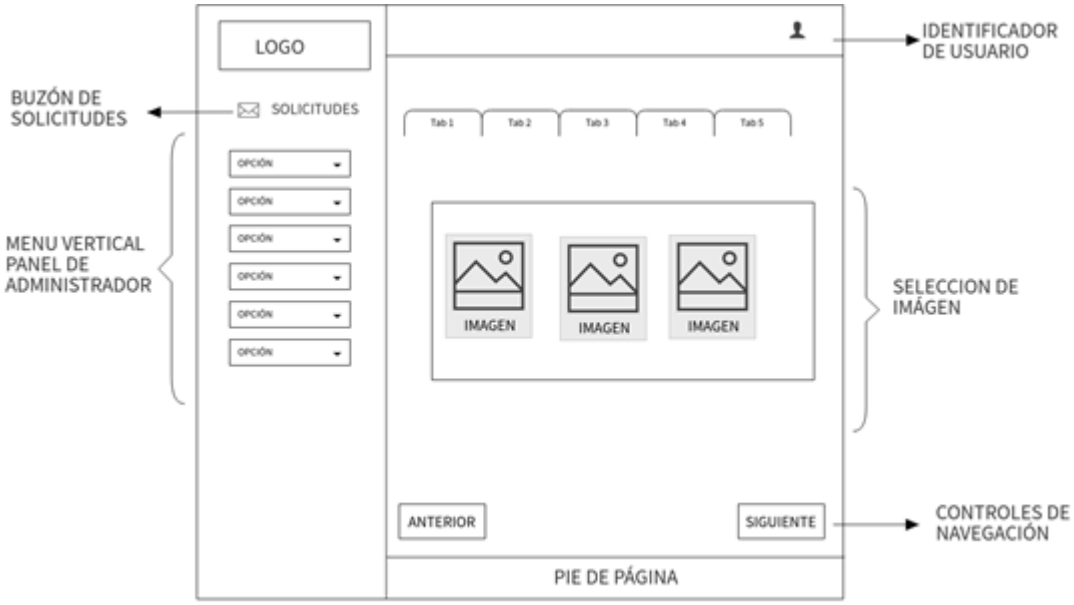


Figura 90. Plantilla de interfaz 4: Ingresar Lugar por usuario administrador.

Los resultados obtenidos nos permiten hacer entrega del incremento obtenido como un módulo del sistema que cumple con su fin, según el flujo normal de eventos y el flujo alternativo del Caso de Uso “Ingresar Lugar”, en el cual se ejecutaron procesos asociados con la planificación, ingeniería de requisitos, diseño detallado, codificación, pruebas y verificación y validación del producto, lo cual se puede visualizar en las siguientes imágenes.

El Administrador selecciona la opción “Ingresar lugar”. El sistema muestra el formulario correspondiente.

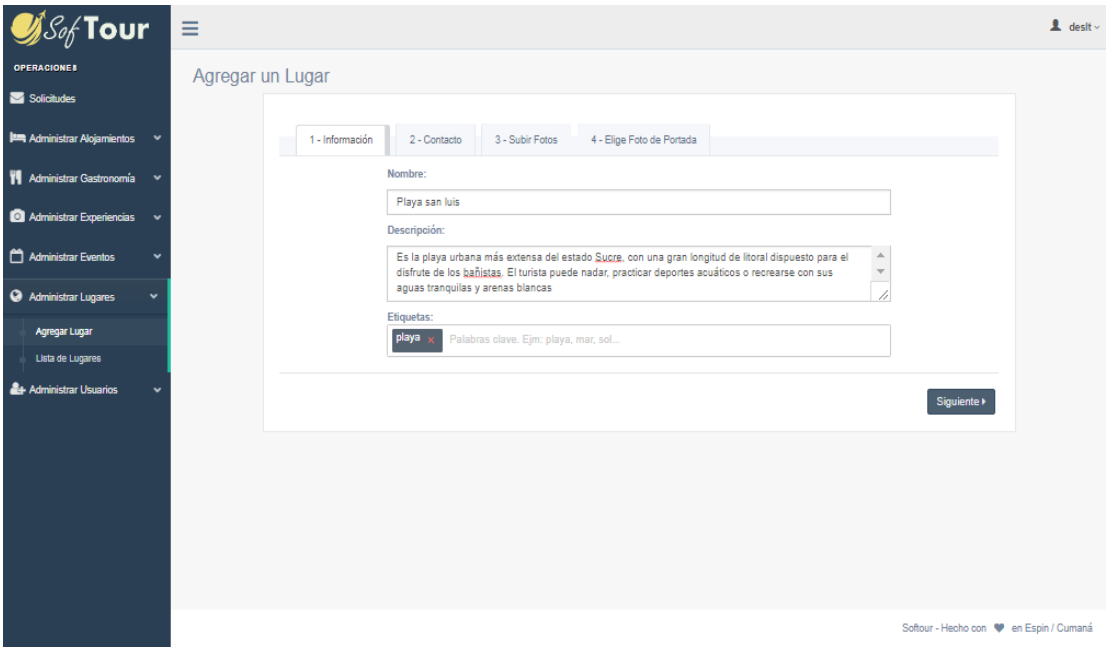


Figura 91. Interfaz 1 de proceso de agregar lugar por usuario Administrador.

El Administrador ingresar los atributos del lugar, lo cual se visualiza en diferentes pestañas: Información, Contacto, Subir fotos y Elige foto de portada.

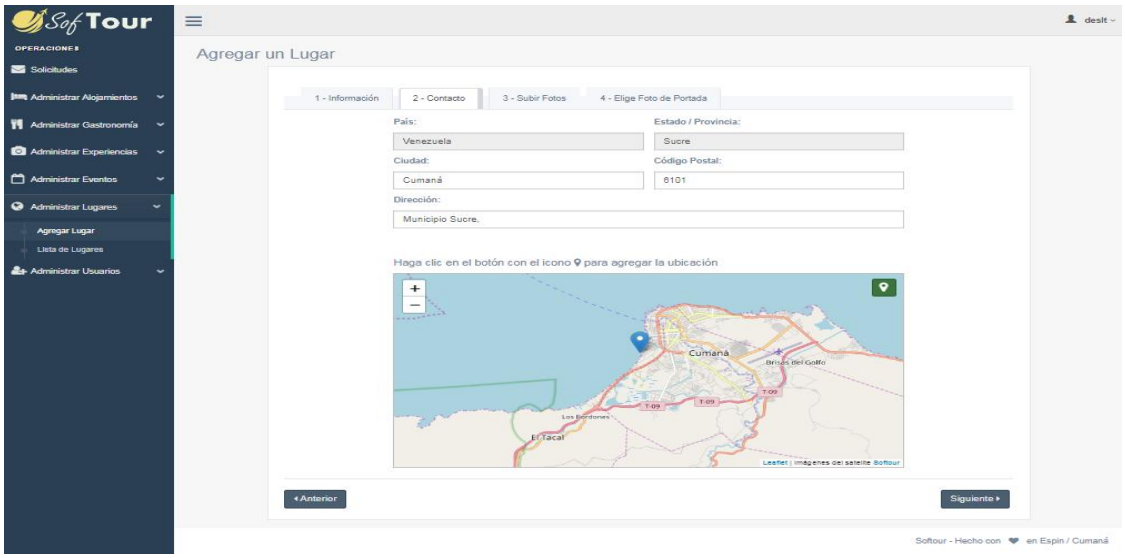


Figura 92. Interfaz 2 de proceso de agregar lugar por usuario Administrador.

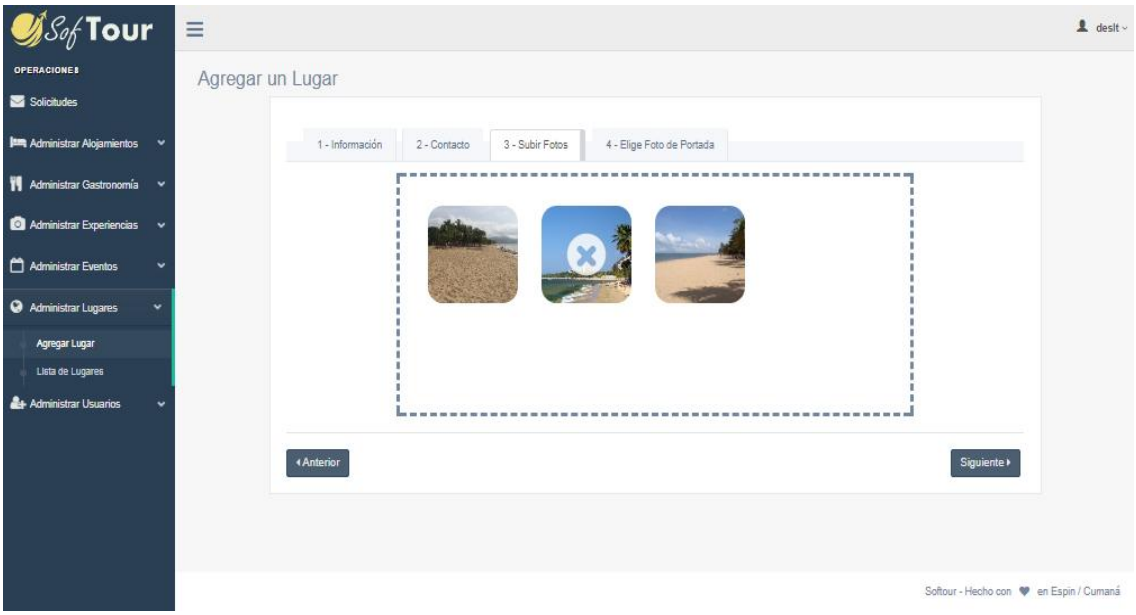


Figura 93. Interfaz 3 de proceso de agregar lugar por usuario Administrador.

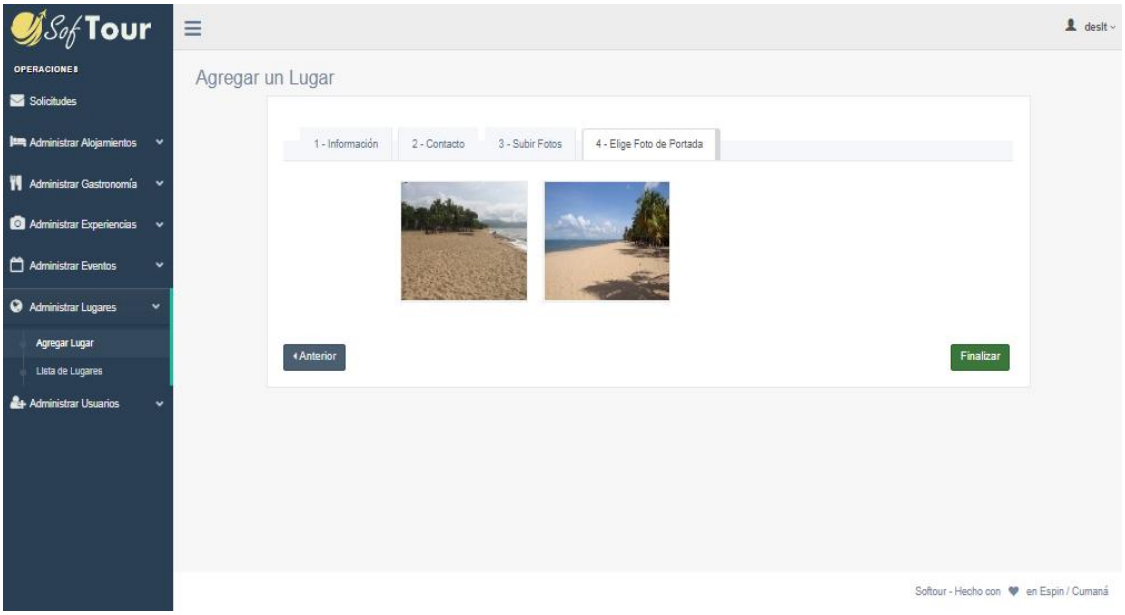


Figura 94. Interfaz 4 de proceso de agregar lugar por usuario Administrador.

Se muestra un ejemplo de codificación en Express.js de los métodos asociados al Caso de Uso “Ingresar Lugar”.

```
ingresar_POST(req, res, next) {
    this._validar.campos(req.body, false, (error, resultados) => {
        if(error) next(error);
        res.locals.datos = resultados;

        if (res.locals.datos.error) {
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;

            this._respuesta.para_ingresar_POST(req, res);
        }
        else {
            res.locals.datos.value.geo_ubicacion = {
                "type": "Point",
                "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
            };
            res.locals.datos.value.ubicacion = {
                "direccion": res.locals.datos.value.direccion,
                "pais": res.locals.datos.value.pais,
                "estado": res.locals.datos.value.estado,
                "ciudad": res.locals.datos.value.ciudad,
                "codigo_postal": res.locals.datos.value.codigo_postal
            };

            this._Lugar.buscar({"url": res.locals.datos.value.url}, (error, lugar) => {
                if (lugar) {
                    res.locals.datos.error = { "details": [ {
                        "path": "nombre",
                        "message": 'Ya existe un lugar con este nombre.'
                    } ] };
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_ingresar_POST(req, res);
                }

                else {
                    let lugar = this._fabricar.lugarTipo(res.locals.datos.value.tipo_lugar);
                    if(req.session.passport.user.tipo_usr == "Administrador" && lugar.__t)
                        res.locals.datos.value.aprobado = true;

                    lugar.ingresar(res.locals.datos.value, (error, creado) => {
                        if(error) {
                            res.locals.datos.error = error;

                            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                            this._respuesta.para_ingresar_POST(req, res);
                        }

                        res.locals.datos = {"value": creado};
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_ingresar_POST(req, res);
                    });
                }
            });
        }
    });
}
```

Figura 95. Código desarrollado para el proceso de agregar lugar por usuario Administrador.

Prueba Ingresar Lugar

Los casos de prueba para la evaluación de la función que realiza el usuario Administrador etiquetada como “Ingresar Lugar”, se realizó de la siguiente manera:

La prueba se realizó de manera manual, el equipo de pruebas diseño los datos válidos y no válidos a ingresar y evaluó que el módulo cumpliera a cabalidad la funcionalidad asignada.

Tabla 12. Clases de equivalencia prueba Ingresar Lugar.

Condición de entrada	Tipo	Clases de equivalencias válidas	Clases de equivalencias no válidas
Nombre de lugar	Alfabético	1: Cadena de caracteres.	2: Campo en blanco 3: Símbolos
Descripción	Alfanumérico	4: Cadena de caracteres, números.	2: Campo en blanco 5: Cadena de caracteres menores a 150.
Etiquetas	Alfanumérico	4: Cadena de caracteres, números.	
País	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Estado/Provincia	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Ciudad	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Código postal	Numérico	6: Números	1: Cadena de caracteres 3: Símbolos 7: Letras
Dirección	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Fotos	Imagen	8: JPG, GIF, PNG	2: Campo en blanco
Elegir foto de perfil	Selección		

Tabla 13. Casos de prueba para Ingresar Lugar

Condición de entrada	A1	A2	A3
Nombre de lugar	Parque	¡Museo Humbolt!	Monumento
Descripción	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.		Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Etiquetas	Parque; distracción; diversión	Museo; Teléfono; 123	Paseo; visitas;
País	Venezuela	México	
Estado/Provincia	Sucre		Amazonas
Ciudad	Cumaná		4568
Código postal	6101	5555	nchd
Dirección	Calle Mariño, sector 1		
Fotos	PNG, JPG	Doc, PDF	
Elegir foto de perfil	Selección	Selección	
Clases Equivalencia	1,4,6,8	1,2,3	1,2,4,5,7
Resultado	Registro satisfactorio	Registro no satisfactorio por datos no válidos para el sistema.	Registro no satisfactorio por datos no válidos para el sistema.

Para el caso A1 se obtuvo un registro satisfactorio, se llenaron las casillas de acuerdo a lo requerido por el sistema y la función cumplió su fin.

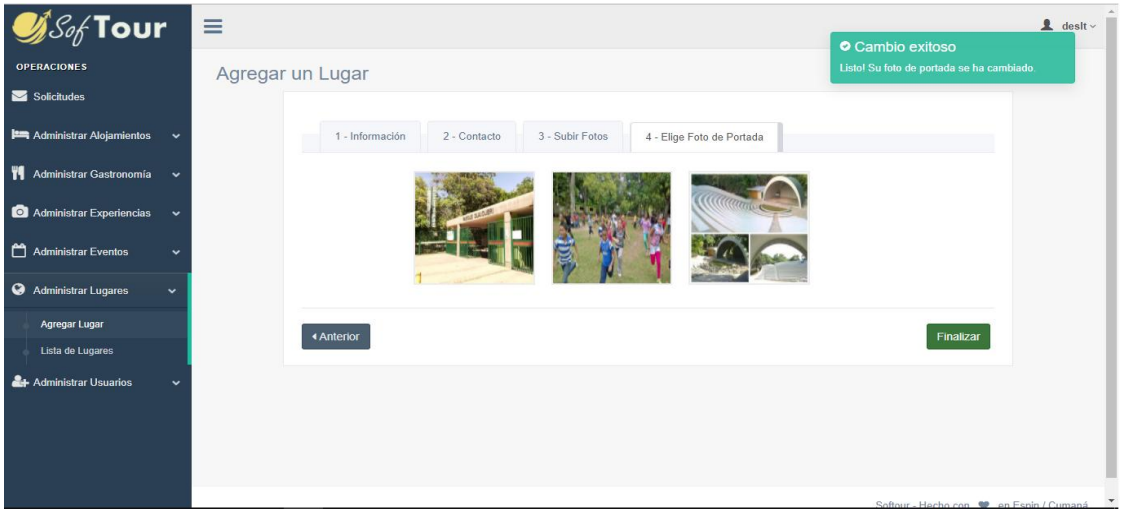


Figura 96. Caso de prueba A1 para Agregar Lugar.

Para el caso A2 el registro no se pudo completar, desde el primer paso de agregar el lugar el sistema rechazo el nombre por contener signos de exclamación, documentándolo como no valido, comprobando que la validación se cumple.

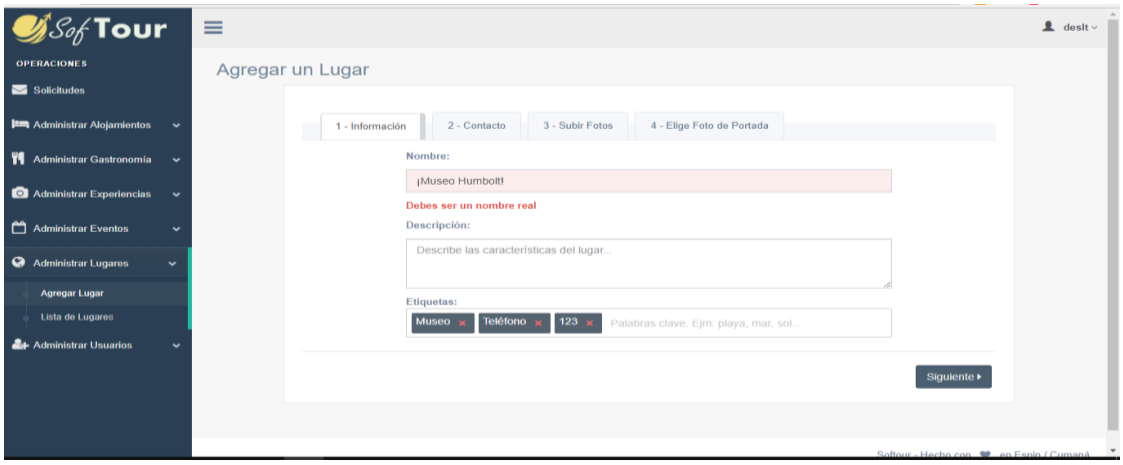


Figura 97. Caso de prueba A1 para Agregar Lugar.

Para el caso de pruebas A3 el registro no fue satisfactorio, inclusive, no se pudo avanzar del segundo paso de registro debido a que las validaciones del sistema

evitaron continuar ya que se introdujo un dato inválido en uno de los campos, ya que se trató de introducir letras en donde el campo solo permite números.

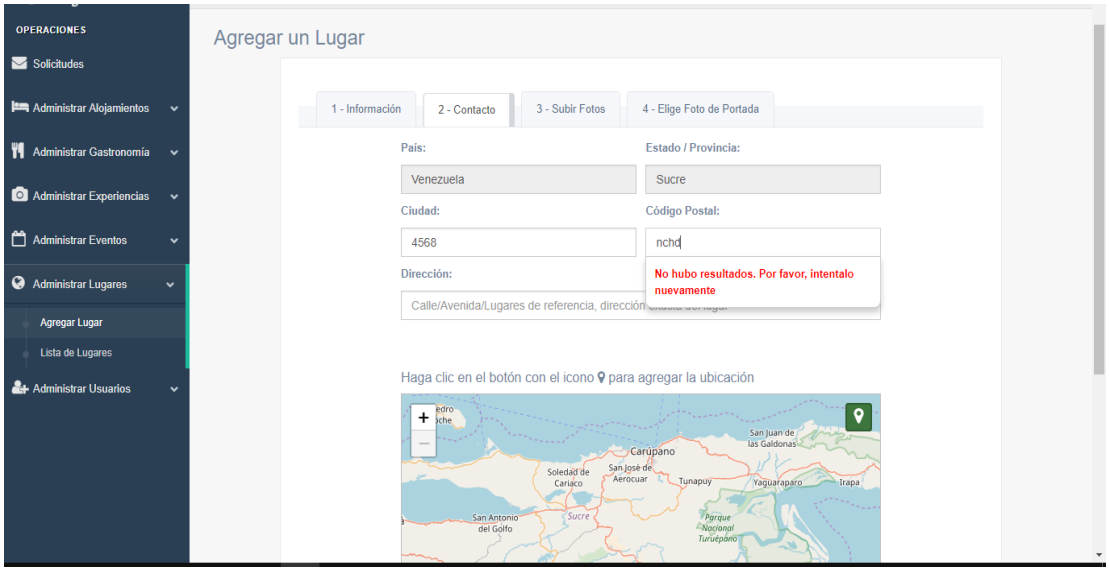


Figura 98. Caso de prueba A1 para Agregar Lugar.

Conclusión de la prueba

El módulo demostró estar validado en todos sus campos, evitando que se llenaran campos con datos inválidos y así evitar que la base de datos fuese llenada con información errónea, pero se realizó la observación al equipo de desarrollo del caso A2, ya que pueden existir nombres de lugares que puedan llevar algún número en su nombre.

Prueba de interfaz y usabilidad

Cabe señalar que desde el comienzo del desarrollo del software una de las ideas principales era diseñar una interfaz que considerara los principios de usabilidad, para así asegurar el éxito del sistema.

A continuación, se muestra una captura de pantalla de la función “Agregar Lugar” al cual se le realizó un análisis detallado para corroborar que cumpliera

con los principios heurísticos de usabilidad.

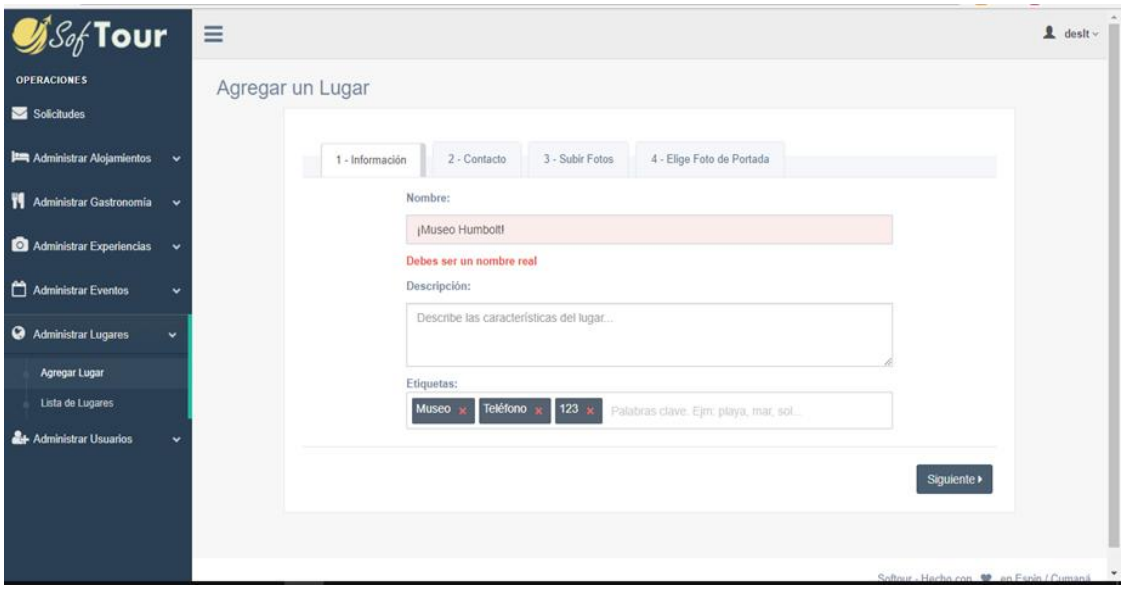


Figura 99. Caso de prueba A1 para Agregar Lugar.

Como se puede constatar en la imagen, el diseño es minimalista, utiliza palabras sencillas que se adaptan al ámbito, muestra metáforas para el reconocimiento de funciones, enumera los pasos a seguir para mantener al usuario informado en qué punto se encuentra y mediante sombreado indica en que área del panel está ubicado, además de señalizar en tonalidad roja cuando se encuentra un error en los datos.

Caso de Uso “Modificar Evento” por el usuario prestador de servicio, definiéndolo como la segunda iteración de la segunda versión.

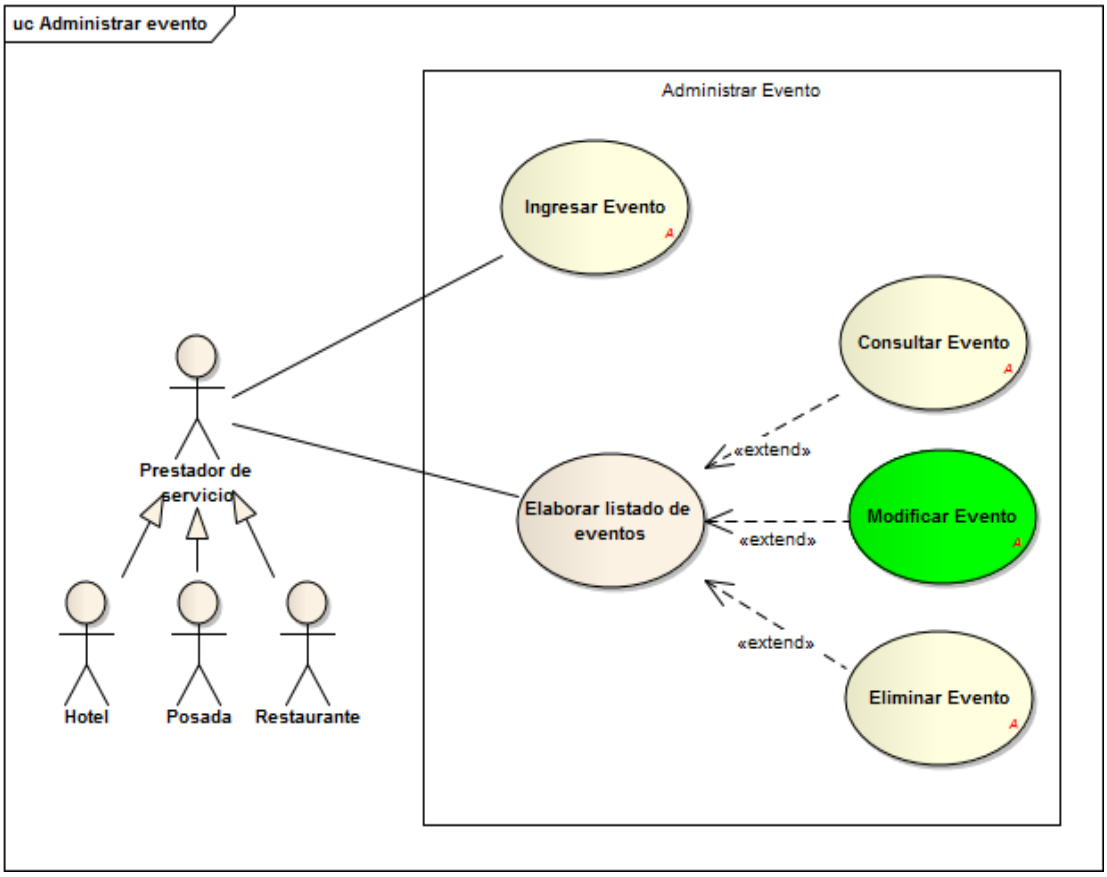


Figura 100. Caso de uso: Modificar Evento por prestador de servicio.

Tabla 14. Descripción Caso de uso Modificar Evento por usuario prestador de servicio.

Nombre del Caso de uso	Modificar Evento	
Versión	1.0 25/10/2017	
Actor	Prestador de Servicios.	
Descripción del Casos de Uso	El caso de uso consiste en modificar los atributos del evento de manera satisfactoria.	
Precondición	Para modificar un Evento, el mismo debe estar registrado en el sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Prestador de Servicios selecciona la opción "Modificar Evento".
	2	El sistema ejecuta la búsqueda del evento, mostrando un formulario con los atributos respectivos.
	3	El Prestador de Servicios modifica los atributos del evento, selecciona aceptar.
	4	El sistema realiza las modificaciones en la BD y finaliza el caso de uso.
Postcondición	Se modifican los atributos del evento de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Prestador de Servicios decide no modificar los Eventos y selecciona cancelar el caso de uso (selecciona el botón cerrar el formulario o el botón cancelar).
	3.2.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. El evento debe estar registrado para que pueda ser modificado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

Diagrama de clases para funcionalidad Modificar Evento por usuario Prestador

de Servicio.

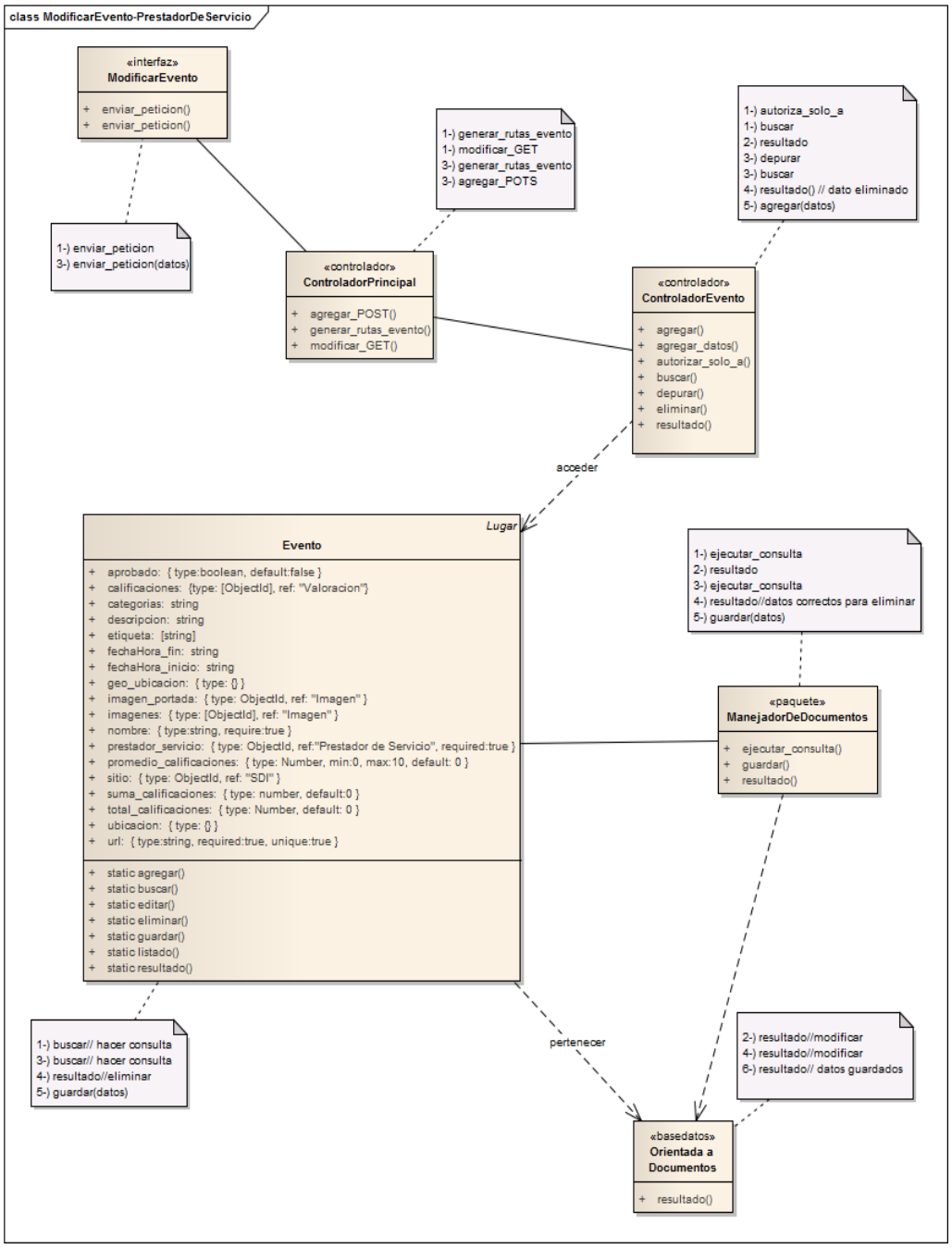


Figura 101. Diagrama de Clases: Modificar Evento por usuario Prestador de Servicios.

Diagrama de Secuencia para funcionalidad Modificar Evento por usuario Prestador de Servicio.

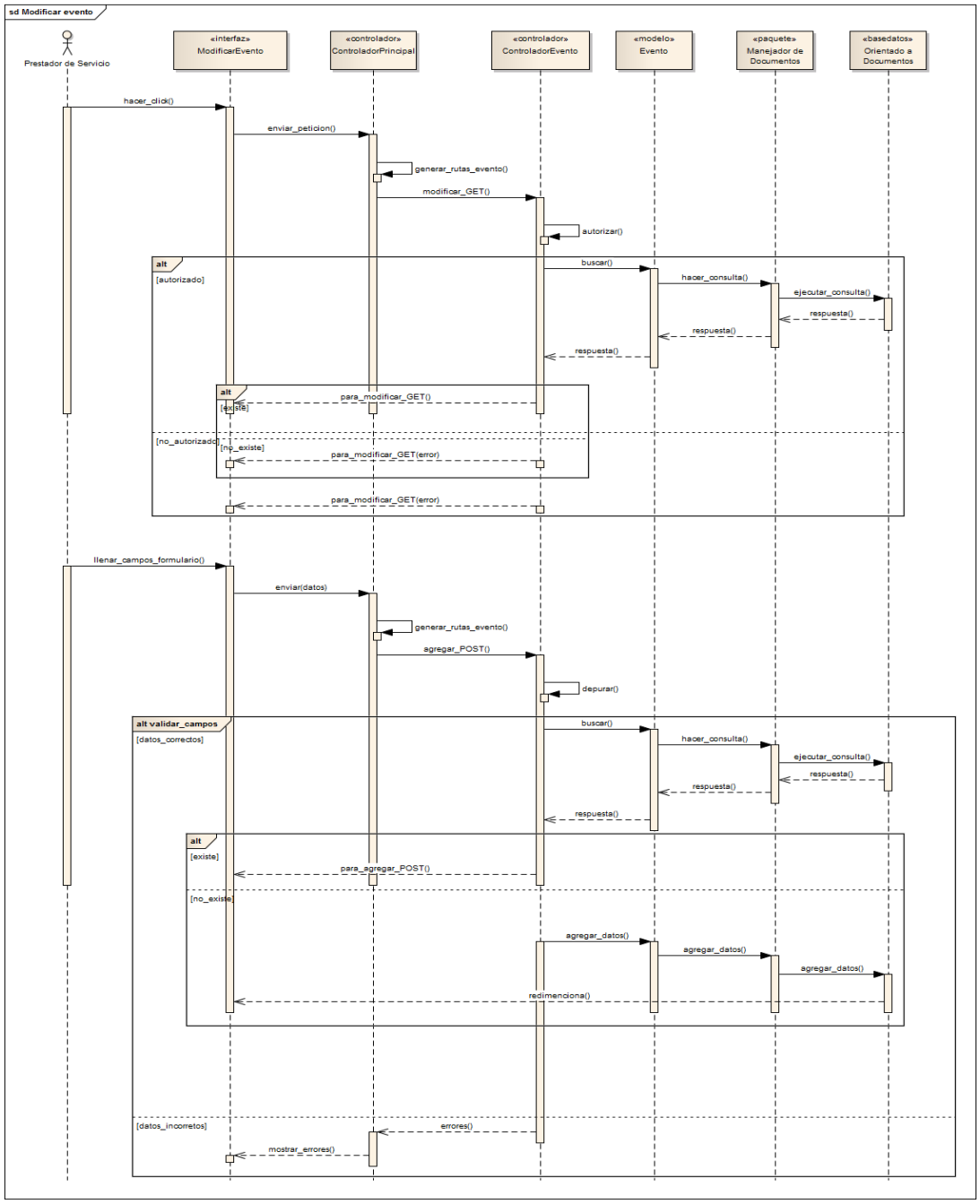


Figura 102. Diagrama de secuencia: Modificar Evento por usuario Prestador de Servicio.

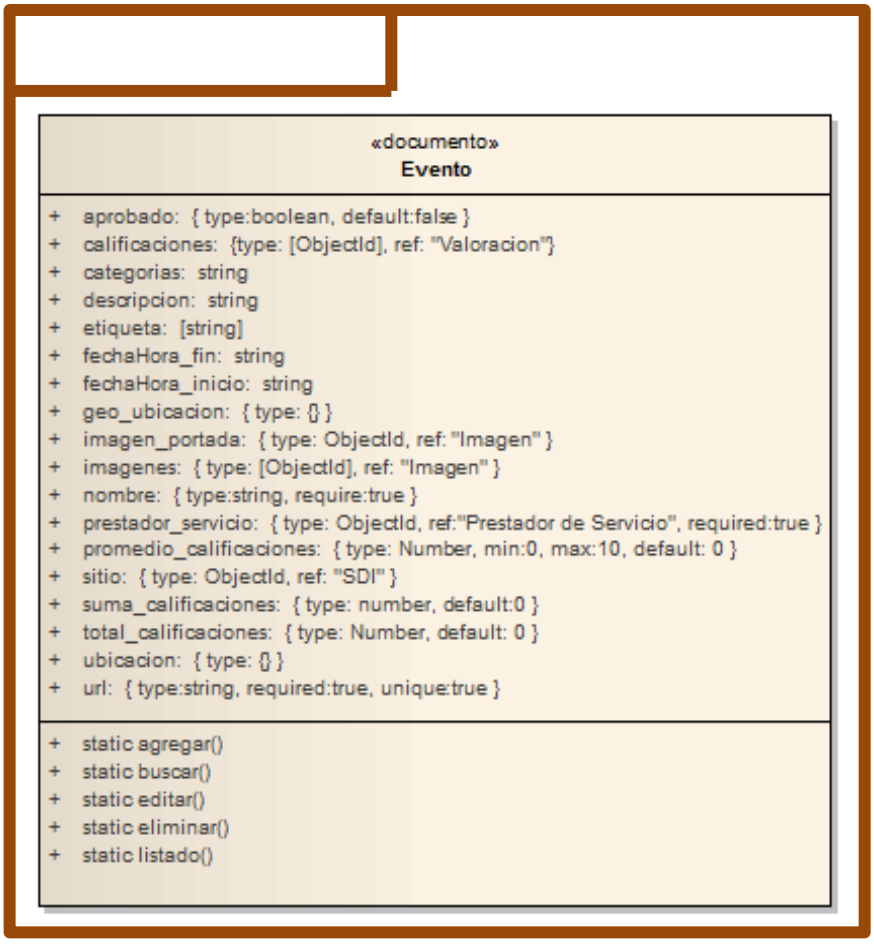


Figura 103. Imagen Colección de Evento: Modificar evento por usuario Prestador de Servicio.

A continuación, se muestran los modelos de plantillas para el proceso de modificar evento.

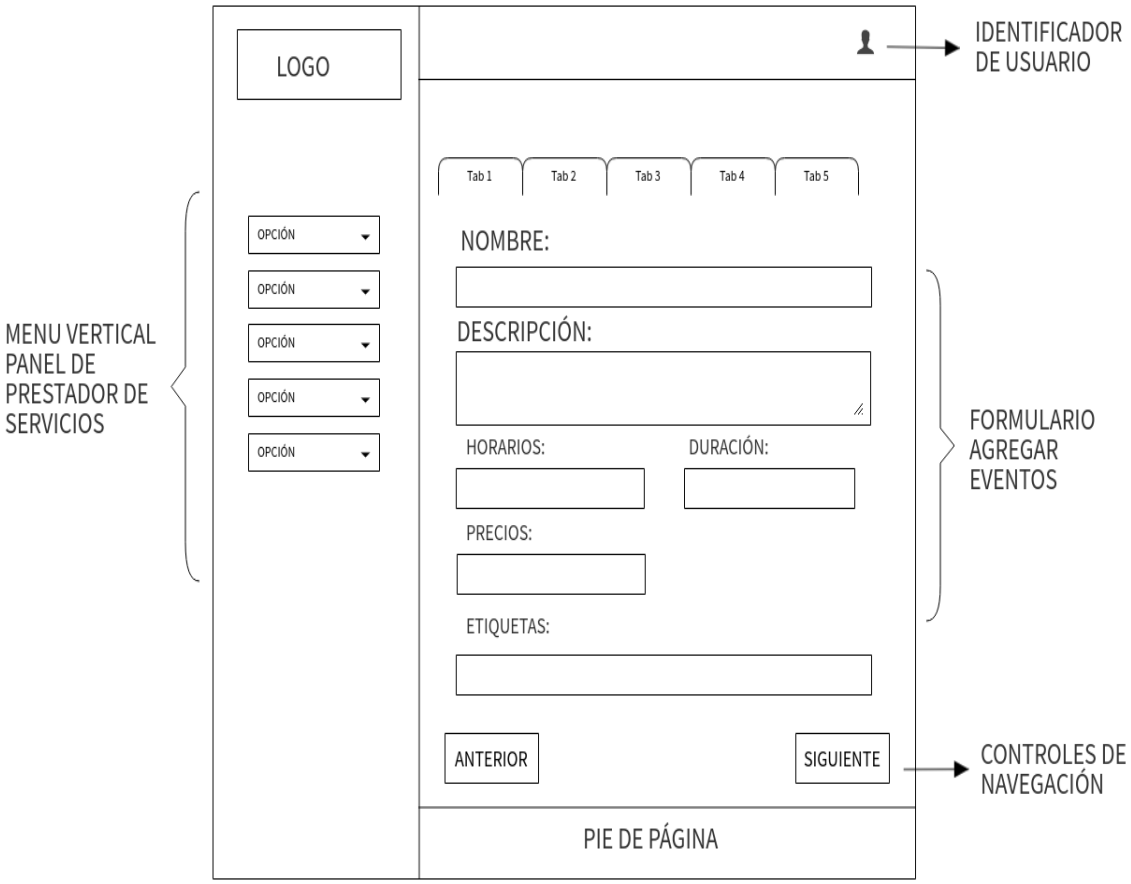


Figura 104. Plantilla de interfaz 1: Modificar Evento por usuario prestador de servicio.

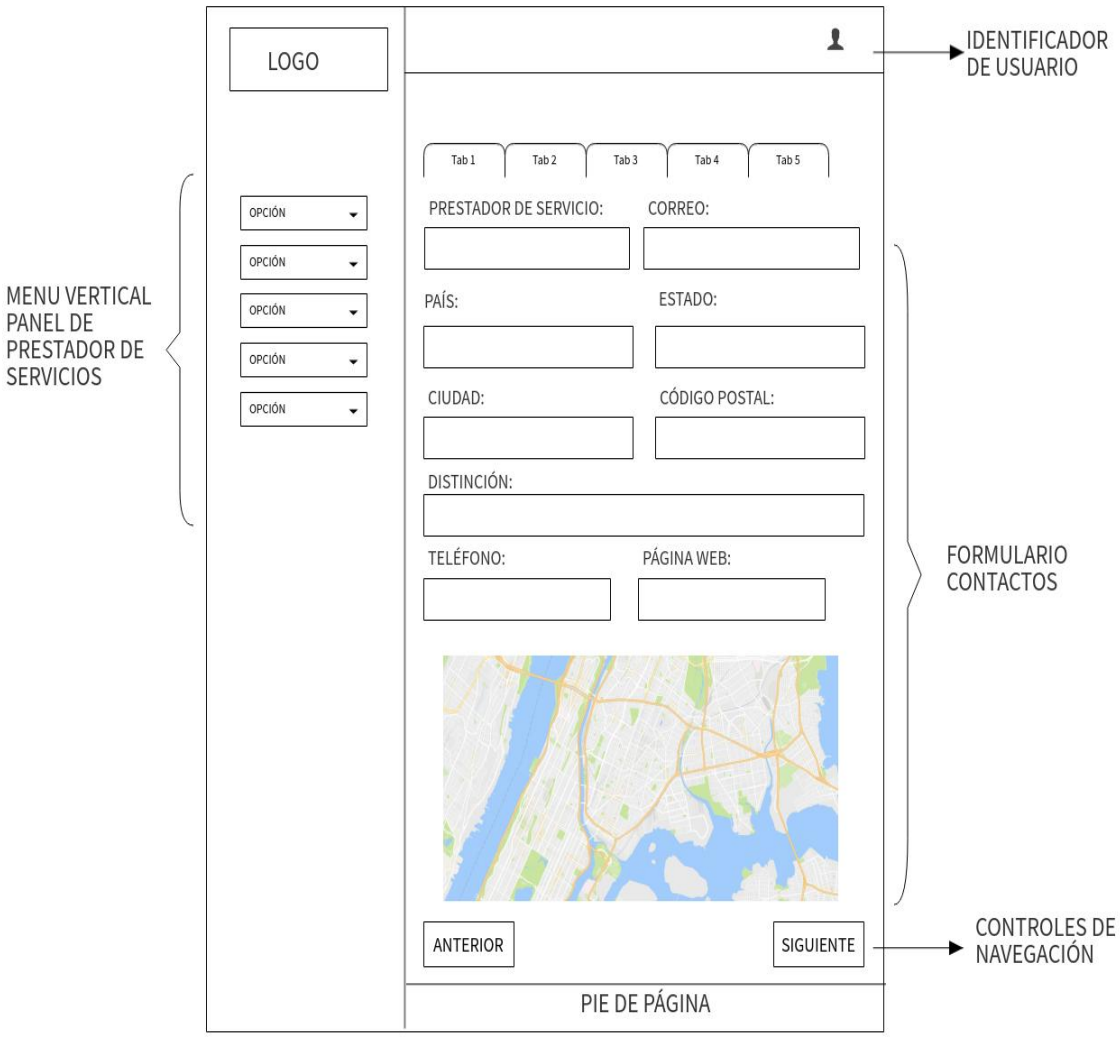


Figura 105. Plantilla de interfaz 2: Modificar Evento por usuario Prestador de Servicio.

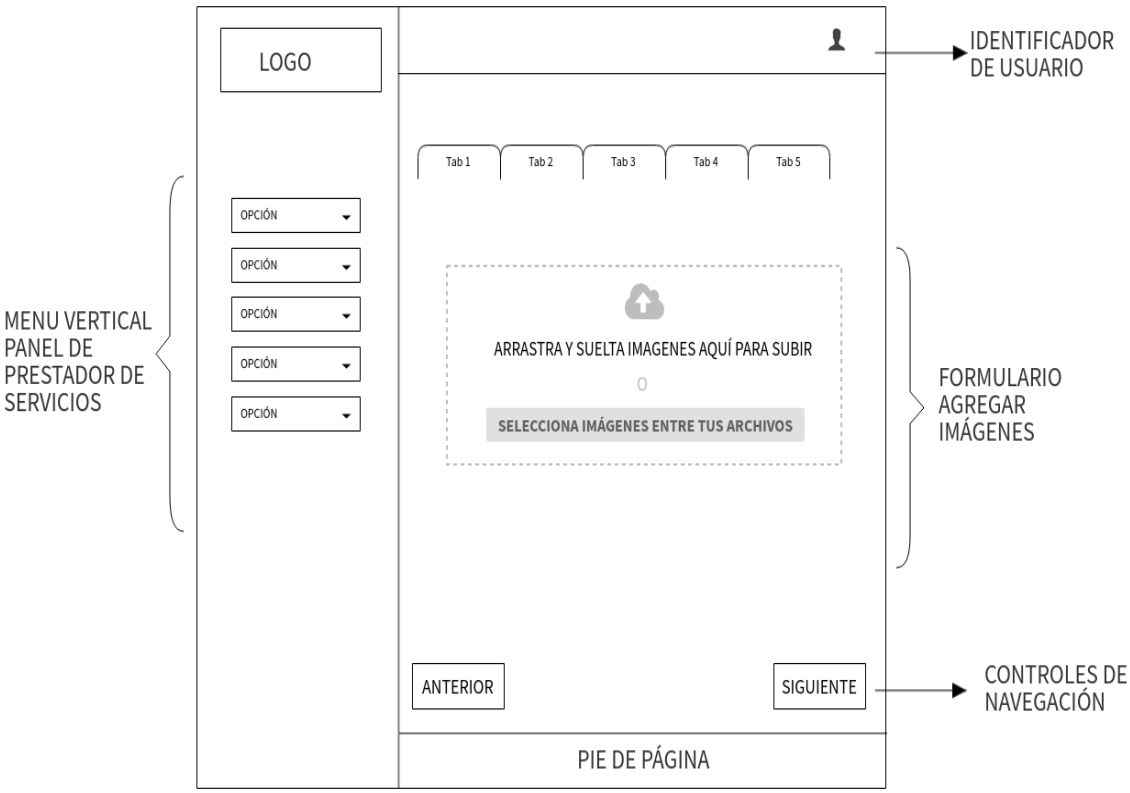


Figura 106. Plantilla de interfaz 3: Modificar Evento por usuario Prestador de Servicio.

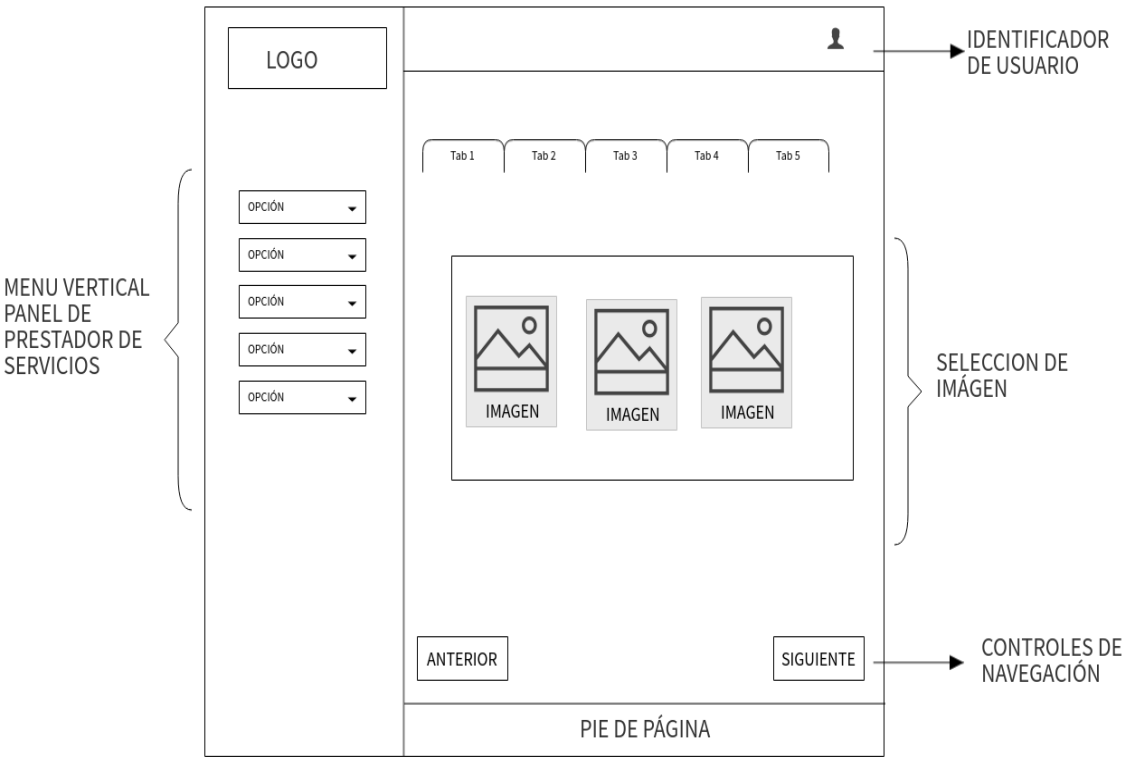


Figura 107.Plantilla de interfaz 3: Modificar Evento por usuario Prestador de Servicio.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Modificar Evento y el sistema responde mostrando el formulario correspondiente.

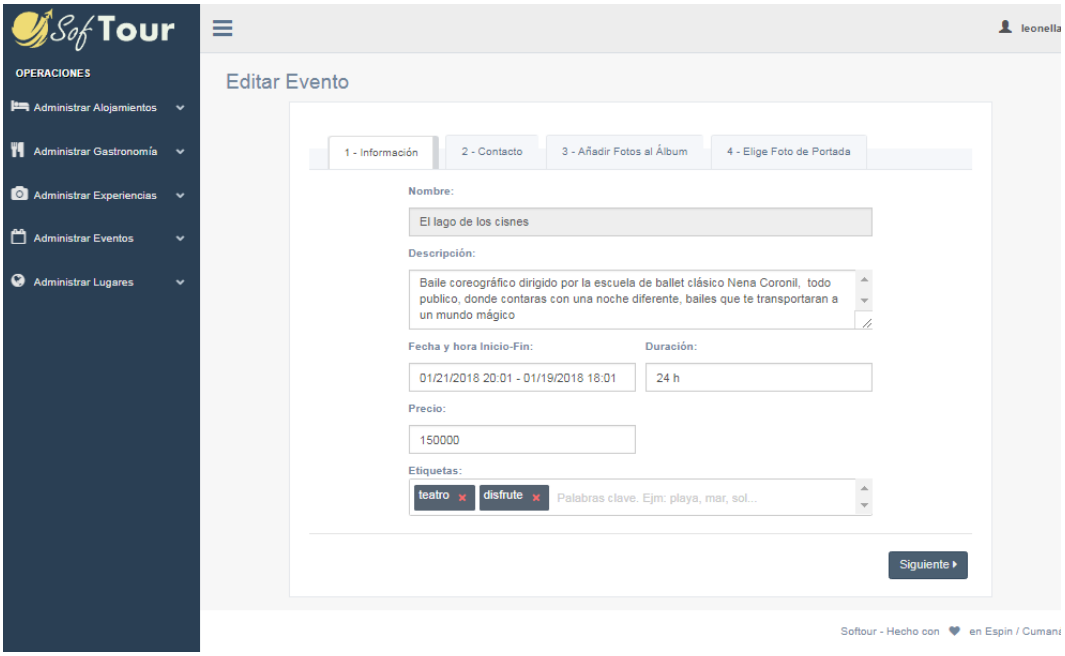


Figura 108. Interfaz 1 de proceso de Modificar Evento por usuario Prestador de Servicio.

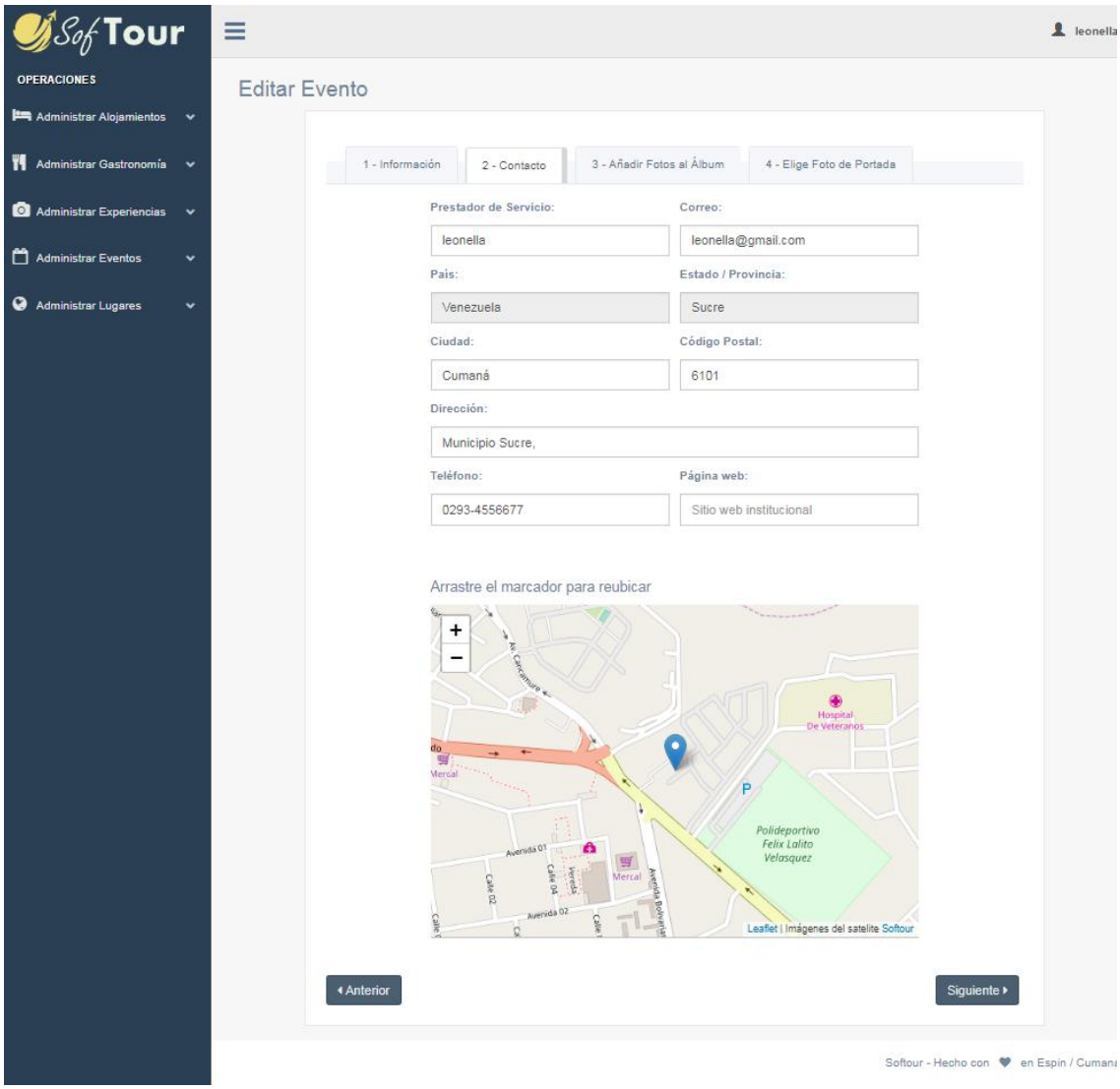


Figura 109. Interfaz 2 de proceso de Modificar Evento por usuario Prestador de Servicio.

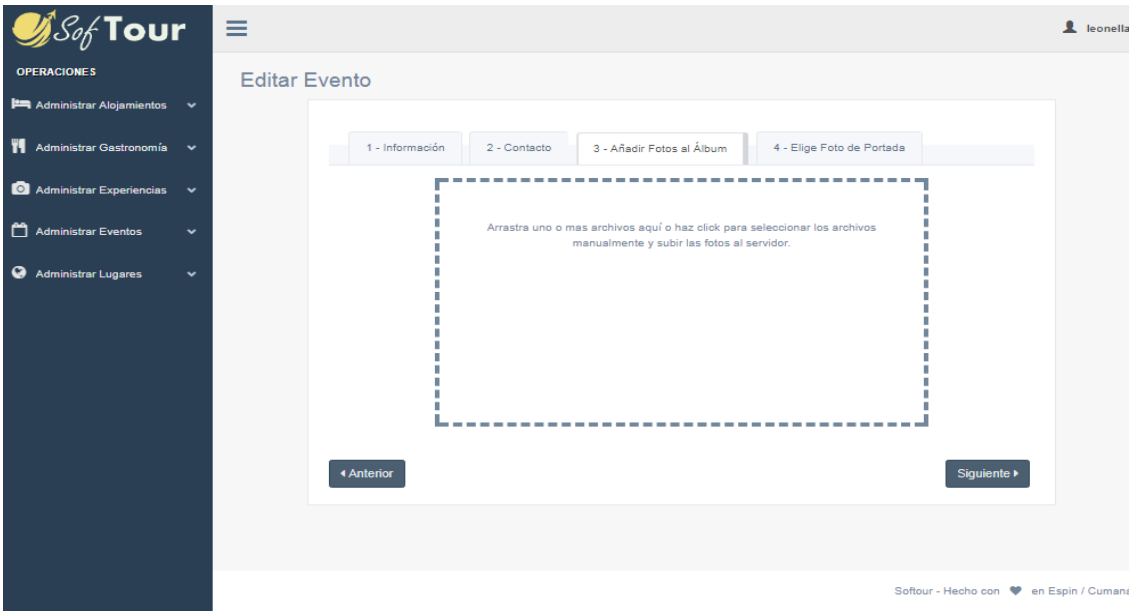


Figura 110. Interfaz 3 de proceso de Modificar Evento por usuario Prestador de Servicio.

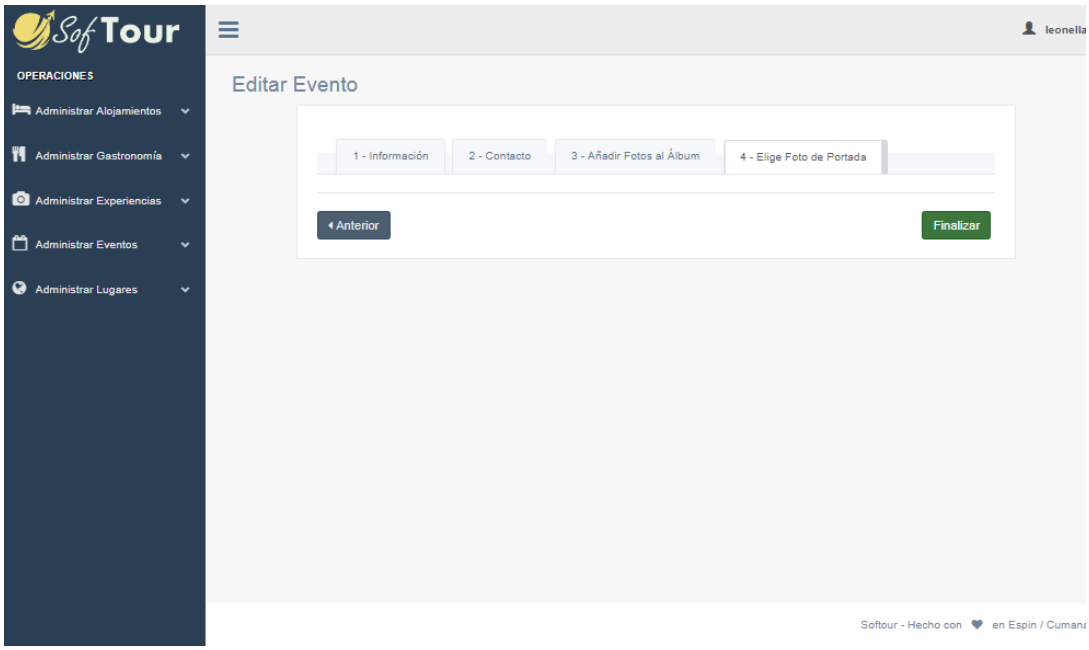


Figura 111. Interfaz 4 de proceso de Modificar Evento por usuario Prestador de Servicio.

Código desarrollado para la funcionalidad Modificar Alojamiento por usuario Prestador de Servicio.

```
modificar(req, res, next) {
    this._validar.campos(req.body, true, (error, resultados) => {
        if(error) return next(error);
        res.locals.datos = resultados;

        if(res.locals.datos.error){
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_modificar(req, res);
        }

        else {
            this._Evento.buscar({"url": req.params.url.toLowerCase()}, (error, buscado) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
                    this._respuesta.para_modificar(req, res);
                }

                if(!buscado) return next(new this._error_NoEncontrado());
                res.locals.datos.value.geo_ubicacion = {
                    "type": "Point",
                    "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
                };
                res.locals.datos.value.ubicacion = {
                    "direccion": res.locals.datos.value.direccion,
                    "pais": res.locals.datos.value.pais,
                    "estado": res.locals.datos.value.estado,
                    "ciudad": res.locals.datos.value.ciudad,
                    "codigo_postal": res.locals.datos.value.codigo_postal
                };

                res.locals.datos.value.url = req.params.url.toLowerCase();
                this._Evento.modificar(res.locals.datos.value, (error, modificado) => {
                    if(error) {
                        res.locals.datos.error = error;
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_modificar(req, res);
                    }
                    if(!modificado) return next(new this._error_NoEncontrado());

                    res.locals.datos = {"value": modificado};
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_modificar(req, res);
                });
            });
        }
    });
}
```

Figura 112. Código desarrollado para el proceso Modificar Evento por usuario Prestador de Servicio.

Pruebas Modificar Evento

Para llevar a cabo esta prueba se deben tener valores eventos agregados en sistema por ende se utilizaron valores añadidos en pruebas. En esta prueba se buscó validar que el usuario Prestador de servicio pudiese modificar información de un evento.

Para el caso de prueba se modificó el evento denominado “Amanecer llanero” con los siguientes datos

Se cambiaron los siguientes valores durante la prueba para validar la funcionalidad del módulo:

Fecha: 02/03/18.

Descripción: GRAN AMANECER LLANERO Ven y disfruta con toda la familia te esperamos Artistas: Jorge Guerrero, Fernando Tovar, Cristian Castillo, Manolo Guerrero, Freddy Ramos y Carlos Mèlendez. Con un gran espectáculo y homenaje de vida artística a Reynaldo Armas. amanecer llanero,cantantes,joropo,valle de la pascua.

Duración: 8 horas.

Ciudad: Cumanacoa.

Direccion: Calle las Flores

Para llevar a cabo esta prueba se seleccionó un evento previamente cargado en sistema y se procedió a modificar cierta información.

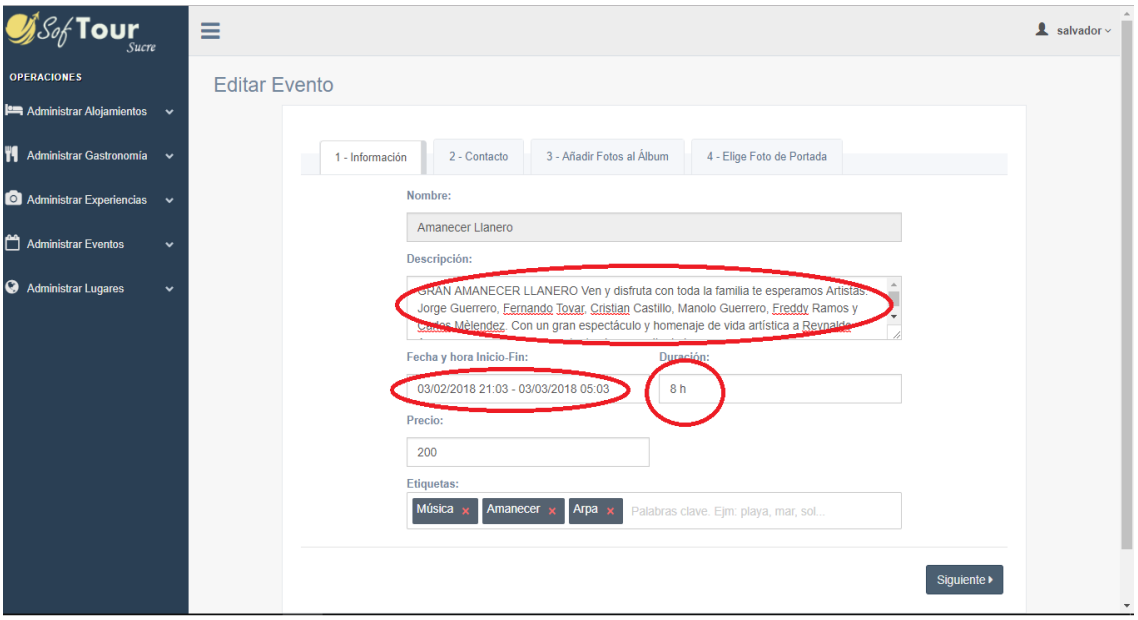


Figura 113. Modificación de información de un evento por Prestador de Servicio.

Se modificaron varios apartados del formulario buscando vulnerar o conseguir defectos en el sistema.

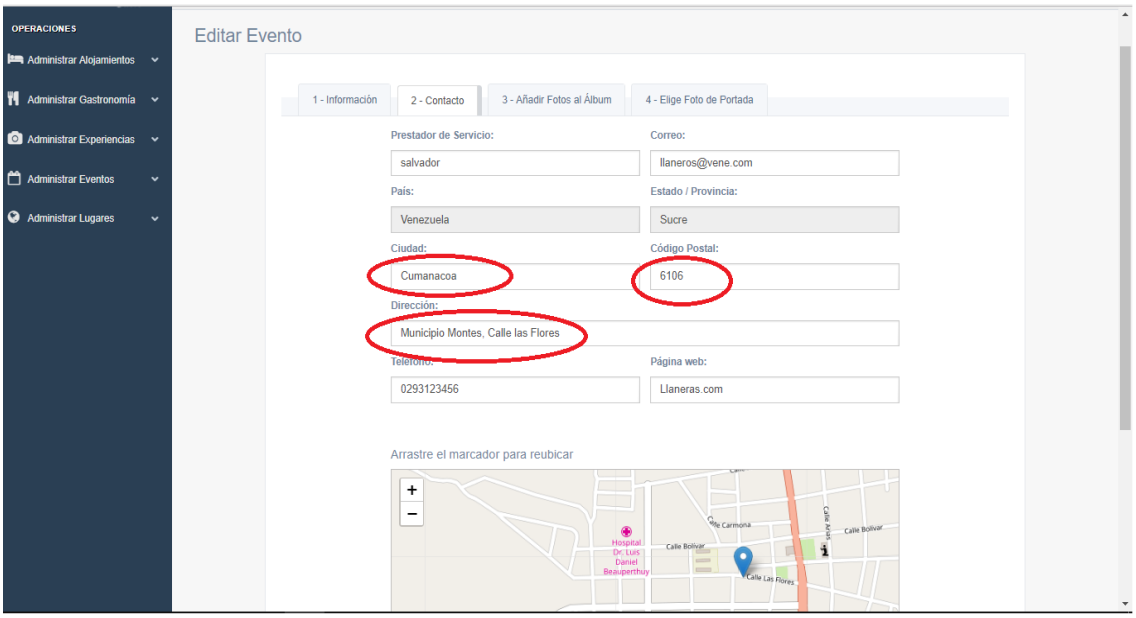


Figura 114. Prueba de Modificar Evento por parte del usuario Prestador de Servicio.

Luego de validar por sistema que los datos ingresados cumplen con los parámetros establecidos en el formulario se procede a guardar los cambios, lo cual se dio de forma exitosa.

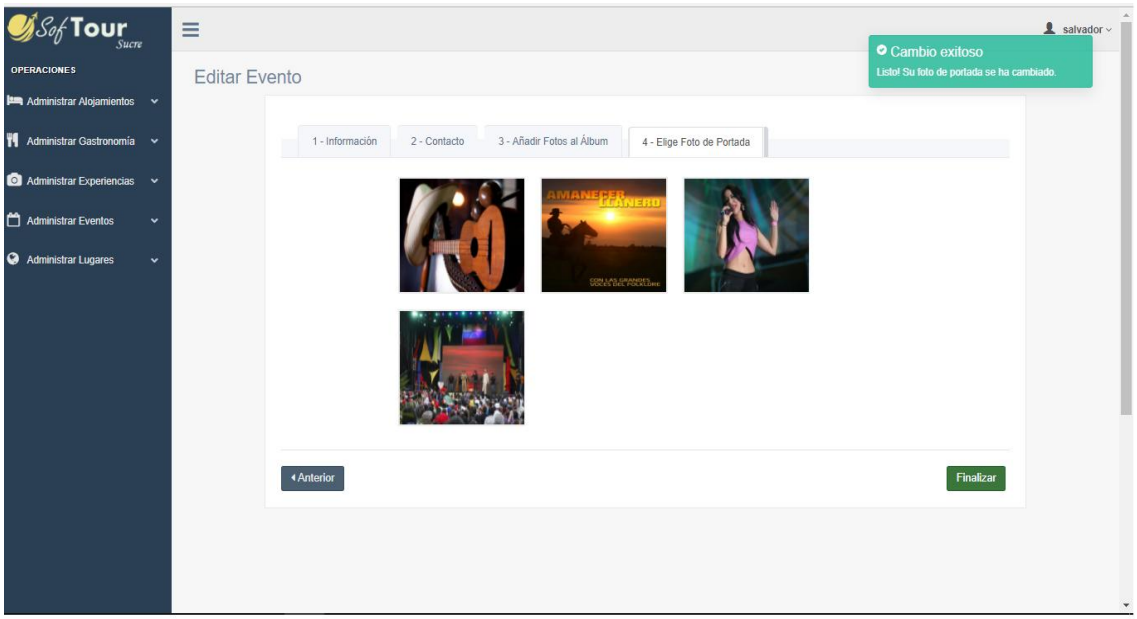


Figura 115. Prueba finalizada de Modificar Evento por usuario Prestador de Servicio.

Conclusión de la prueba

La modificación del evento se dio de forma exitosa, lo que indica que la función cumple con su finalidad.

Prueba de interfaz y usabilidad

Esta interfaz pertenece a interfaz de usuario Administrador y Prestador de Servicio, evaluado en pruebas anteriores y que cuenta con estándares de usabilidad.

Caso de Uso “Consultar Experiencia” por el usuario turista, definiéndolo como la tercera iteración de la tercera versión.

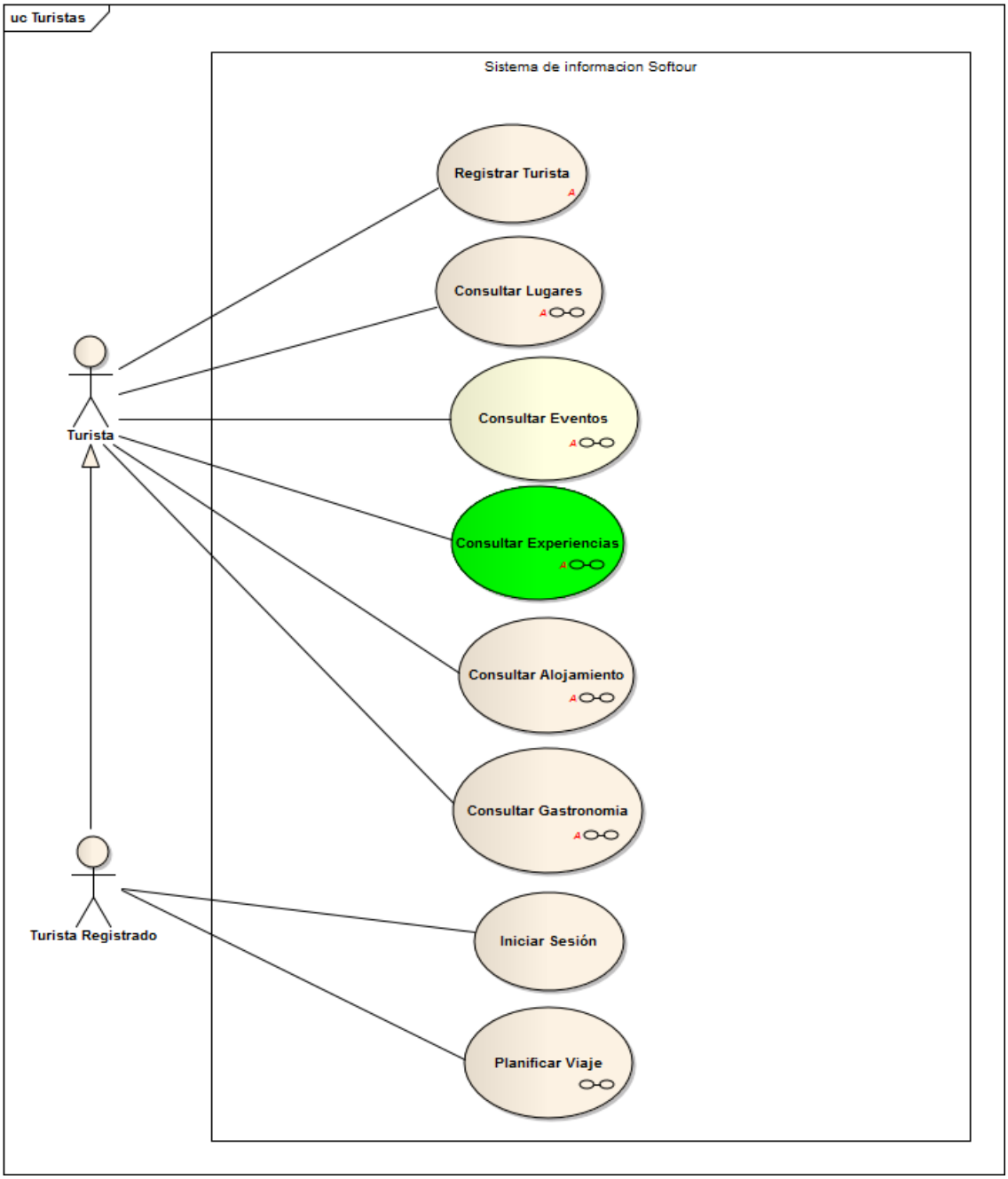


Figura 116.Caso de uso: Consultar Alojamiento por usuario Turista.

Tabla 15. Descripción Caso de uso Consultar Alojamiento por usuario Turista.

Nombre del Caso de uso	Consultar Experiencias	
Versión	1.0 03/11/2017	
Actor	Turista.	
Descripción del Caso de Uso	El caso de uso consiste en consultar las experiencias desde una perspectiva visual, organizada y llamativa. Permitiendo enlazar el caso de uso con patrones de búsqueda específicos relacionados con lo que desea el turista.	
Precondición	Ninguna	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El caso de uso comienza cuando el Turista hace clic en el menú en la categoría “Experiencia” o alguna subcategoría.
	2	El sistema despliega las experiencias disponibles y muestra los posibles filtros que aplican para refinar aún más la búsqueda, acordes a lo que desea el turista. Esto se realiza extendiendo <<extend>> la funcionalidad de la búsqueda con el caso de uso: a.- Administrar patrones de búsqueda de experiencias.
	3	El Turista visualiza los alojamientos disponibles que se muestran y selecciona uno de ellos para ver el detalle del mismo.
	4	El sistema muestra el detalle de la experiencia seleccionada.
	5	El Turista visualiza los atributos de la experiencia.
	6	El sistema cierra el caso de uso y retorna el control.
Postcondición	Se consulta la experiencia desde la perspectiva del turista, satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Turista decide no consultar la experiencia y selecciona cancelar el caso de uso (selecciona el botón cerrar o el botón cancelar).
	4.1.	El sistema finaliza el caso de uso.
Reglas del negocio	1. Las consultas de las experiencias que incluyan filtros o patrones de búsquedas deben considerarse extensiones de este caso de uso. 2. Al seleccionar una experiencia, el turista podrá visualizar mayor detalle del mismo.	
Inclusiones	Ninguna	
Extensiones	1. Administrar patrones de búsqueda de experiencias	

Diagrama de clases para funcionalidad Consultar Experiencia por usuario Turista.

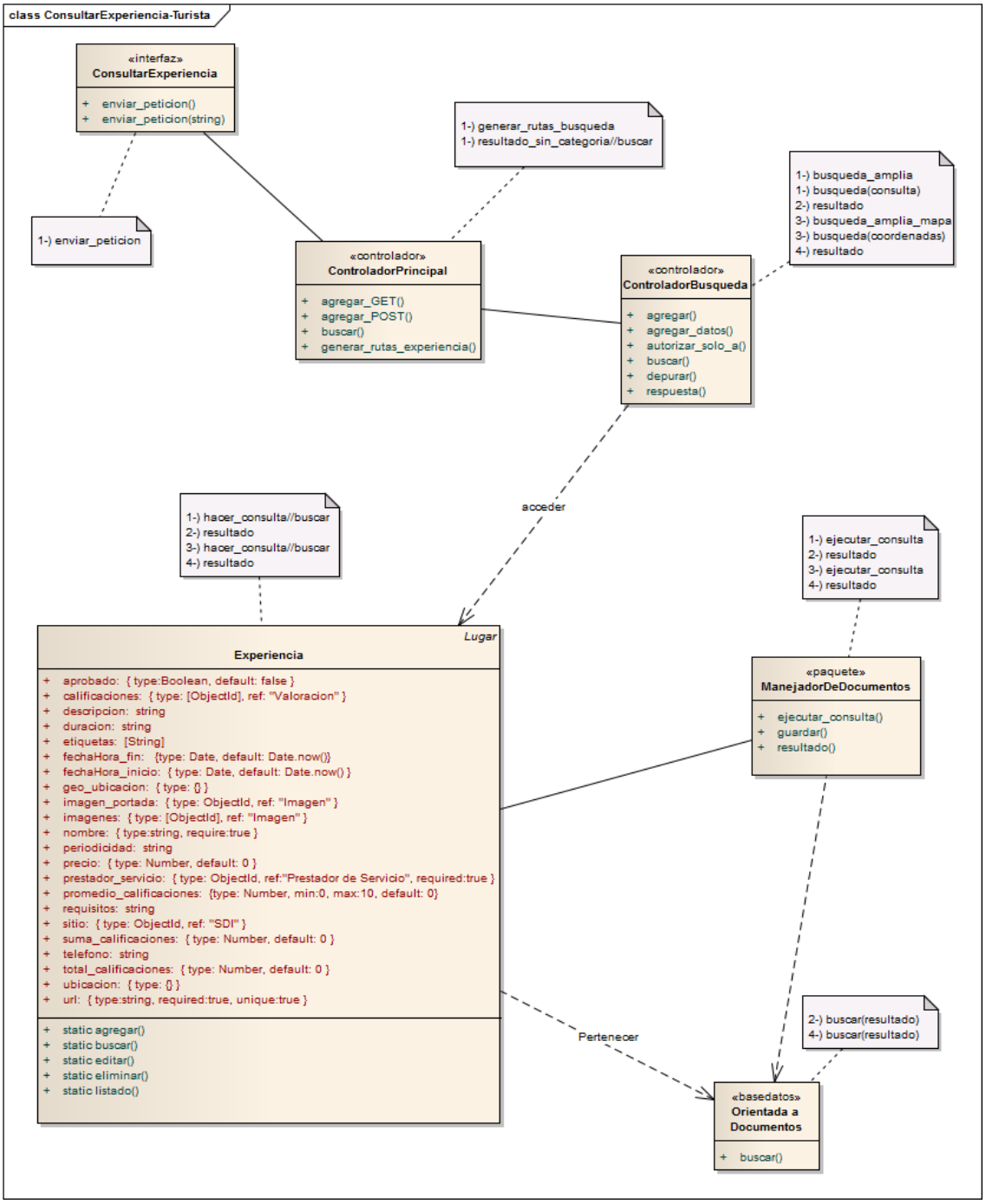


Figura 117. Diagrama de Clases: Consultar Experiencia por Usuario Turista.

Diagrama de Secuencia para funcionalidad Consultar Experiencia por usuario Turista.

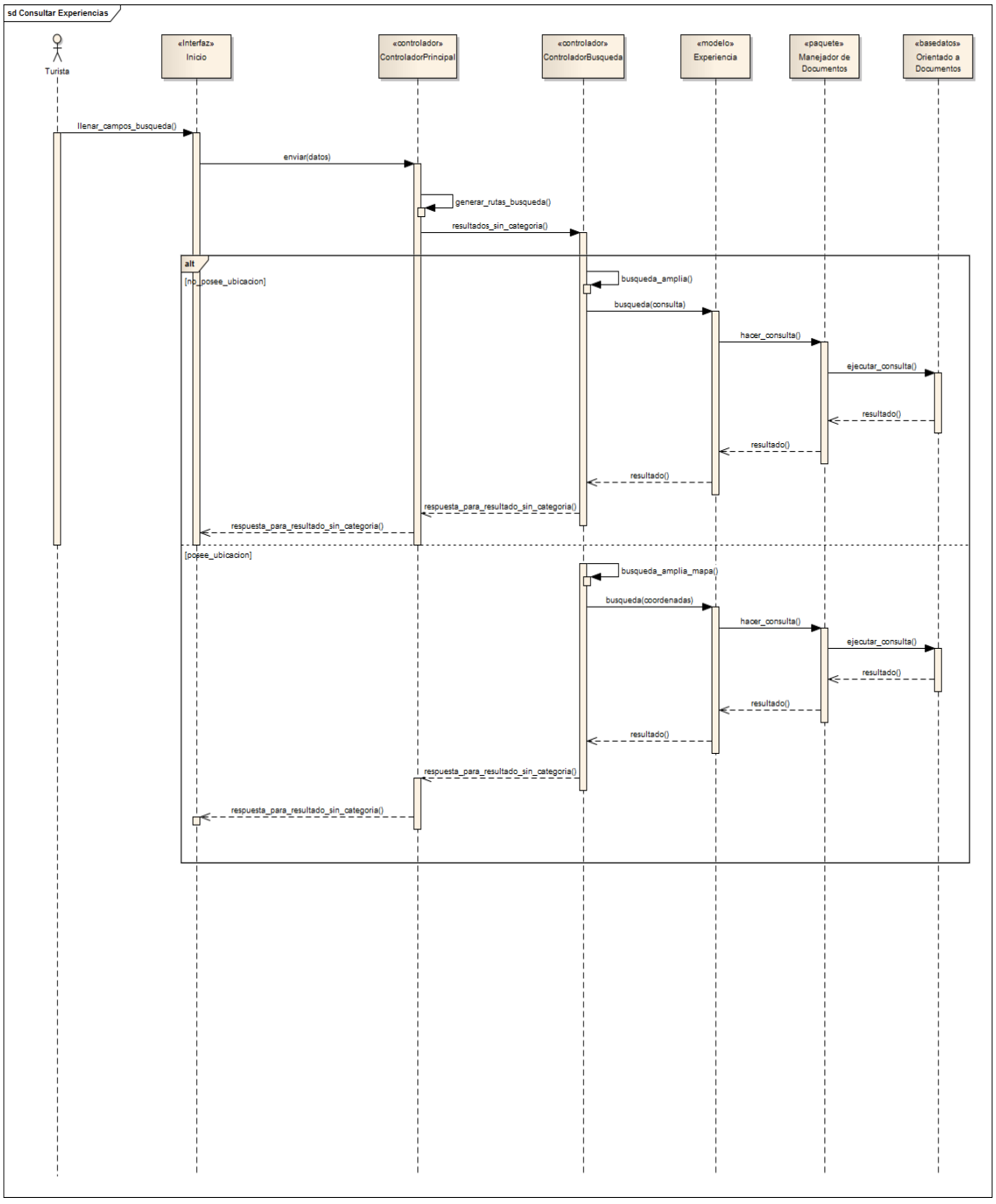


Figura 118. Diagrama de secuencia: Consultar Experiencia por usuario Turista.

A continuación, se muestran los modelos de plantillas para el proceso de consultar experiencia por usuario turista.

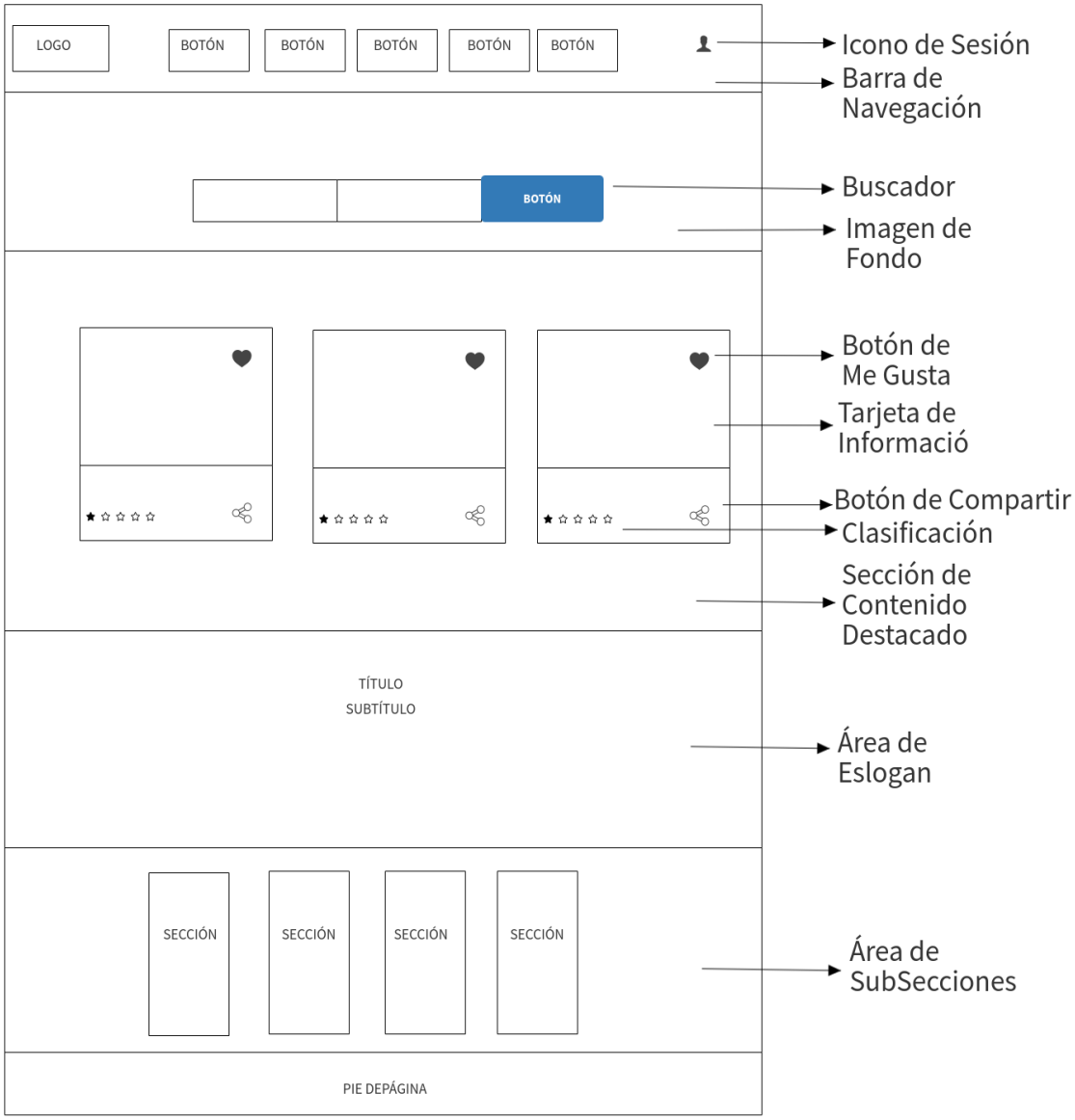


Figura 119. Plantilla de interfaz general: Consultar Experiencia por usuario Turista.

Figura 120. Plantilla de interfaz especifica: Consultar Experiencia por usuario Prestador de Servicio.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Experiencia, y luego selecciona una experiencia en específica y el sistema responde mostrando la interfaz con las especificaciones.

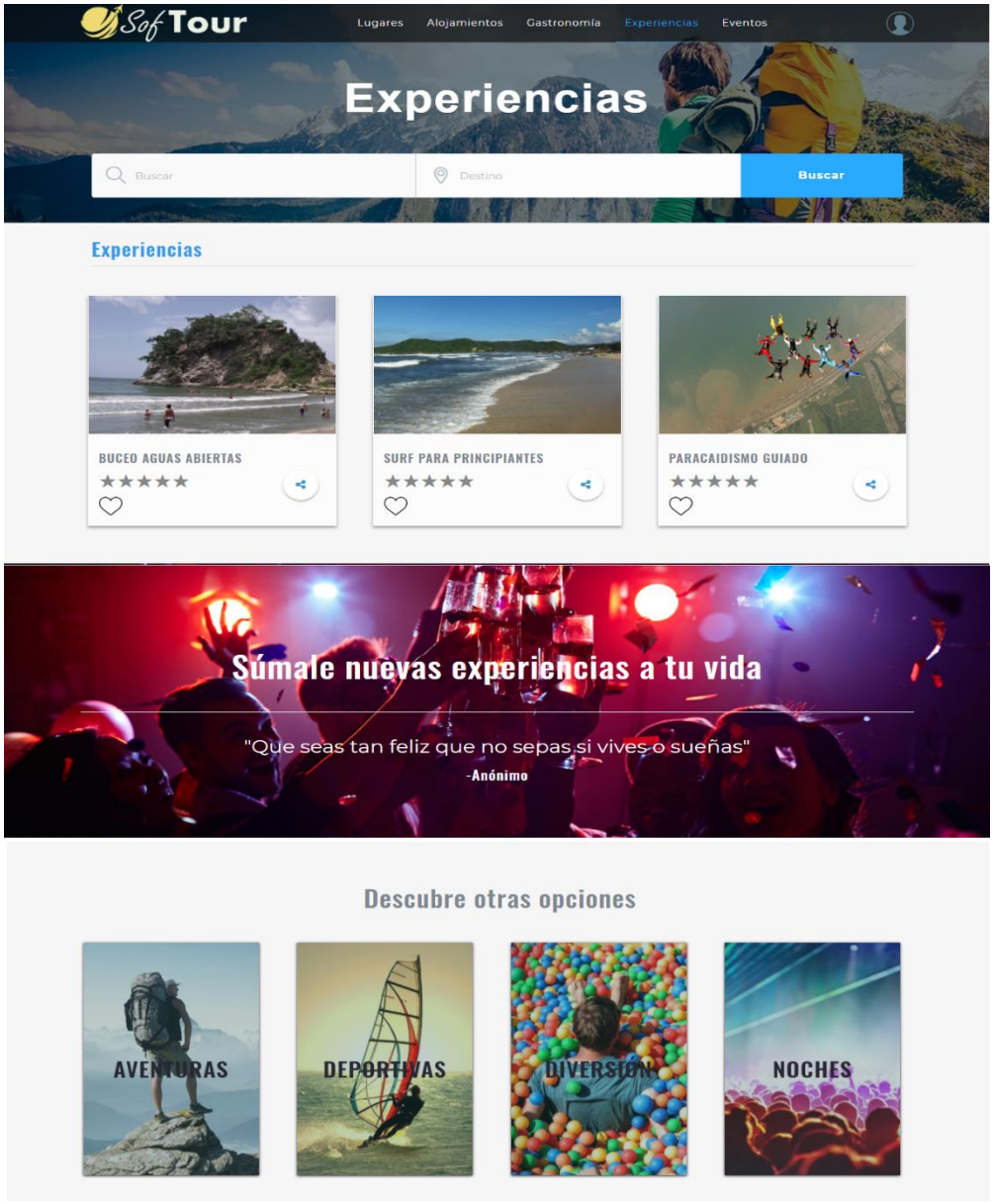


Figura 121.Interfaz general de proceso de Consultar Experiencia por usuario Turista

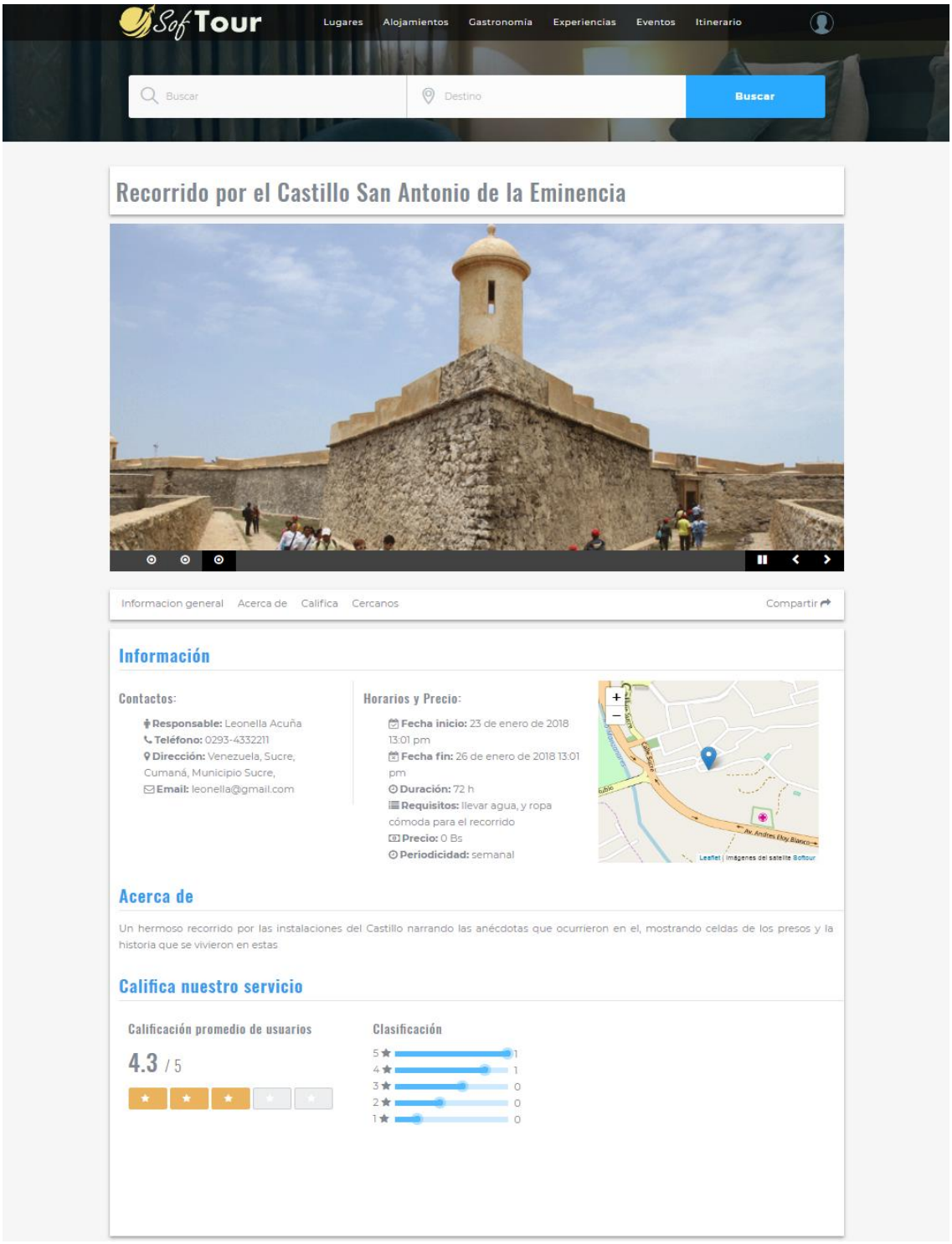


Figura 122. Interfaz especifica de proceso de Consultar Experiencia por usuario Turista

Código desarrollado para la funcionalidad Consultar Experiencia por usuario Turista.

```
consultar(req, res, next) {
    this._Experiencia.buscar_consulta({"url": req.params.url.toLowerCase()}, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": buscado};
        if (req.isAuthenticated())
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_consultar(req, res);
    })
}
```

Figura 123. Código desarrollado para el proceso Consultar Experiencia por usuario Prestador de Servicio.

Prueba Consultar Experiencia

La función a validar es la selección y muestra de la sección experiencias desde el perfil turista.

A continuación, se muestra secuencias de imágenes durante la prueba:

Figura 124. Prueba consultar experiencias por Turista

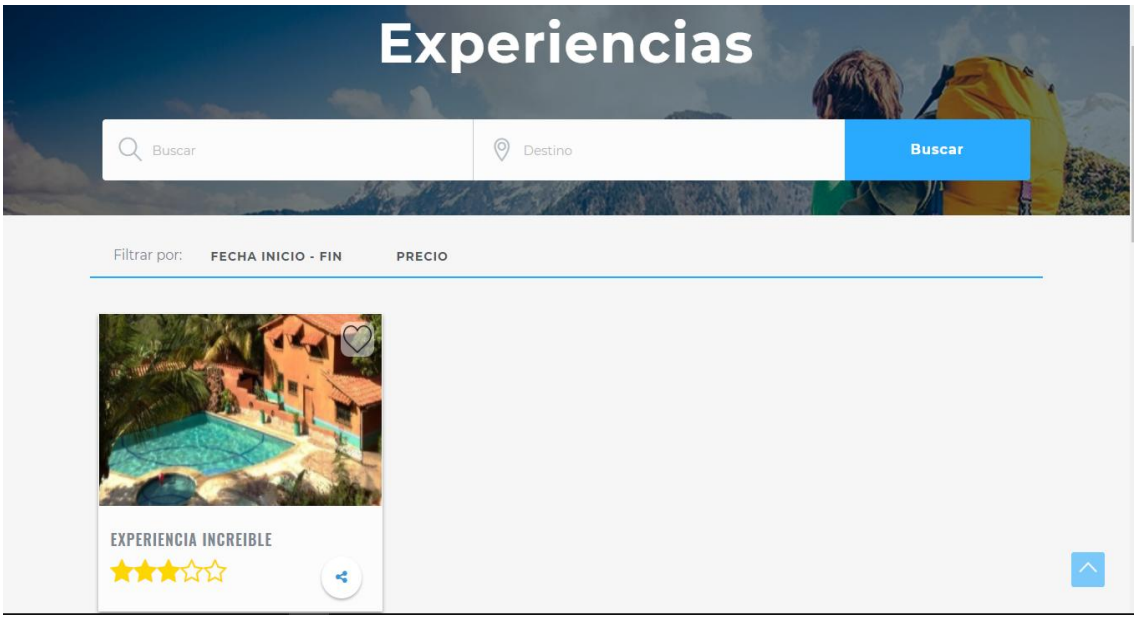


Figura 125. Prueba Consultar Experiencia por usuario Turista.

Conclusión de la prueba

La función se ejecuta de forma correcta.

Prueba de interfaz y usabilidad

Durante encuesta realizada previamente se determinó que la interfaz cumple con estándares de usabilidad.

CONCLUSIONES

El desarrollo de la aplicación Softour permite el incentivo, la promoción y apoyo al sector turismo, dando a conocer diferentes zonas o regiones en donde desee ser implementado, resaltando lugares, eventos y experiencias que se puedan visitar o realizar durante un viaje, ya sea por placer o por trabajo, brindando una información confiable y que aporte datos como lo son horarios, ubicación, servicios, brindando imágenes reales y actualizadas de lo que se desee visitar. Además de proporcionar la opción de compartir y calificar para así mantener una retroalimentación entre el software y sus usuarios.

Basándose en la simplificación de las tareas para las personas con el uso de la tecnología y de aplicaciones usables y predecibles, Softour suministra un apartado para realizar un itinerario de actividades, que le permite a los usuarios planificar su viaje y visitas a las diferentes actividades que proporcione la región donde se utilice el sistema para explotar el turismo, esto busca simplificar y servir de herramienta, programando el viaje y ofreciendo mejores opciones que la competencia.

El desarrollo de un proyecto debe estar guiado por una normativa que de estructura y directriz a los elementos que intervienen de forma directa en la elaboración del proyecto, si esto no se contempla la pérdida de tiempo y trabajo pueden ser incalculables, por ello el equipo de desarrollo tomó las previsiones necesarias utilizando la metodología *WATCH*, específicamente la instanciación *Blue*.

Blue WATCH como marco metodológico proporcionó una base conceptual que unificó y relacionó los conceptos, procesos, productos y roles que caracterizan al desarrollo de software de alta calidad, a través de una estructura de trabajo que facilitó la elaboración de métodos de desarrollo de software que se adaptaron a las necesidades y características particulares de la aplicación Softour y a su equipo de desarrollo.

Blue WATCH sirvió de recurso instruccional para la enseñanza-aprendizaje de la Ingeniería del Software y, particularmente, de los procesos, productos y roles que distinguen al desarrollo de software de alta calidad, lo que ayudo a disminuir el tiempo de la curva de aprendizaje que presentaba el equipo de desarrollo.

Blue WATCH aporto una serie de beneficios valiosos para el proyecto y el equipo de desarrollo, las cuales se resumen en: la mejora significativa de la calidad del software, mejora y estandarización de los procesos de desarrollo del software, unificación de la documentación que se generó en cada uno de los procesos, aseguramiento de la calidad de la documentación requerida para darle un mejor mantenimiento al software en un futuro, reducción de los costos de desarrollo de la aplicación ocasionado por una reducción del trabajo debido al desarrollo no controlado de software, reducción de los tiempos de entrega del software debido a una mejor planificación y control del proyecto, mayor ventaja competitiva ocasionada por un uso más efectivo y eficiente de las tecnologías de información y comunicaciones.

La aplicación Softour fue planteado como sistema de información web ante retos impuestos por economías emergentes y el desarrollo de nuevas tecnologías, en la cual *Blue Watch* resulto de gran ayuda porque sugirió métodos, herramientas y procedimientos para hacer un trabajo menos pesado y seguro, permitió dividir el trabajo en áreas, en donde el producto pasaba por modificaciones en un ciclo iterativo incremental que lo iba mejorando progresivamente hasta lograr el objetivo planteado.

Implementar un trabajo iterativo incremental le ofreció al equipo desarrollo la ventaja de cambiar y modificar elementos del sistema de acuerdo a lo planteado en los requisitos, además de ajustar cambios que surgieron durante el desarrollo de los cursos especiales de grado, obteniendo un producto acabado luego de varias iteraciones.

El Lenguaje Unificado de Modelado (UML) utilizado en las fases del desarrollo permitió modelar, construir y documentar los elementos del portal vertical de manera sencilla y clara, a través de diagramas que permitieron establecer los requerimientos y las necesidades de los usuarios finales. Se utilizó UML para diseñar y explicar la arquitectura y funcionamiento del software, mediante el modelado de negocios, el diseño arquitectónico del funcionamiento del sistema y de los elementos que intervendrían en la integración de componentes.

Para realizar los diagramas que especifica UML se utilizó el software Enterprise Architect, que ofrece un conjunto de herramientas para simplificar y acelerar el proceso de diseño y creación de diagramas.

El lenguaje de enmarcado HTML y CSS permitió un diseño de la interfaz del software que siguiera los estilos y formatos disponibles en la palestra de diseño, respetando las reglas de heurísticas de usabilidad planteadas por Nielsen.

Se utilizaron elementos de *framework Bootstrap* para elaborar las plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, integrando herramientas compatibles para obtener un acabado que cumplas con ciertos estándares de calidad de ingeniería de software en el diseño de la interfaz.

El lenguaje de programación JavaScript facilitó el desarrollo de los ejes funcionales de la aplicación, proporcionando un sistema dinámico, con ejecución en una gran variedad de dispositivos sin mayores inconvenientes, ajustándose a la versatilidad de uso planteado en el alcance planteado obtener con el software SofTour.

El manejador de base de datos MongoDB, no relacional, proporcionó mayor libertad y comodidad al equipo de desarrollo, permitiendo optimizar consultas con una gran cantidad de colecciones de datos, lo cual fue ideal para el manejo de datos implementados en el área de inteligencia de negocios, además

proporciona una base de datos escalable brindando la posibilidad de incrementos futuros del software.

GitHub permitió controlar los incrementos y versiones, adaptando el trabajo en grupo a la ejecución de la metodología *Blue Watch*, esta herramienta permitió mantener al equipo al tanto de las actualizaciones que hacían los diferentes integrantes a la aplicación. Alojar el sistema en los servicios prestados por GitHub sirvió de ventaja en momentos que los desarrolladores trabajaron a distancia, manteniendo un constante crecimiento en paralelo al trabajo de cada integrante.

La introducción de buenas prácticas para llevar a cabo el trabajo en equipo, con asignaciones de responsabilidades y tiempos de ejecución permitió la líder del proyecto y colaboradores establecer una planificación dinámico y constante, incorporando recomendaciones del marco de trabajo SCRUM, como lo son reuniones diarias para saber el avance de cada integrante según lo contemplado en la metodología, las entregas formales de avances, los retrasos e incumplimientos, entre otros. Usar las recomendaciones de SCRUM aseguró que cada integrante tomara la responsabilidad de llevar a cabo su cuota de trabajo establecido para cierto intervalo de tiempo, obteniendo así un manejo óptimo de los tiempos de entrega de los incrementos del producto.

El uso de la herramienta Trello fue importante para gestionar y controlar el trabajo y responsabilidades que debía cumplir cada integrante del equipo, indicando el tiempo en el cual debía culminar las actividades asignadas, además de que cumpliera con diferentes estados para obtener un acabado que llenará las expectativas del equipo. Trello contribuyó a minimizar los riesgos considerando que estaban involucrados siete (7) desarrolladores con realidades distintas, por lo que no era fácil mantener una comunicación constante y un control de las diferentes etapas y productos que necesitaba el proyecto.

De igual forma se estableció el uso de Slack para mantener una comunicación

formal entre los desarrolladores del software y la persona que solicito su creación, de esta manera la comunicación se mantuvo de forma directa y jerarquizada por temas. Utilizando ambas herramientas en conjunto se logró una comunicación fluida y constante, obteniendo productos óptimos a tiempo.

La metodología *WACTH*, en su instanciación *Blue WATCH*, permitió llevar a cabo los procesos técnicos, gerenciales y de soporte por medio del equipo de trabajo en donde cada integrante desempeño diversos roles, posibilitando el desarrollo del software propuesto (Montilva *et al*., 2014) garantizando la calidad y el buen desempeño del sistema de información bajo ambiente web para el incentivo, promoción y apoyo al sector turismo, logrando una experiencia invaluable en el manejo de prácticas, estándares, herramientas, procesos, subprocesos, actividades y tareas que permitieron asegurar el cumplimiento de los requisitos establecidos.

Implementar BBM y UML brindo la facilidad de elaborar documentos que complementaron los productos resultantes de las iteraciones.

Desarrollar la aplicación bajo el uso de las herramientas Express.js, Node.js, MongoDB, Mongoose, JQuery y Bootstrap, implicó un reto y un cambio de paradigma para el equipo de desarrollo, creándose la necesidad de realizar estudios e investigaciones para levantar el conocimiento necesario para trabajar bajo este enfoque, el equipo pudo adaptarse, levantar el conocimiento y llevar adelante el desarrollo utilizando estas herramientas novedosas y robustas para el desarrollo de aplicaciones web, dejando un aprendizaje valioso para cada uno de los integrantes, mejorando sus cualidades y preparándolos para asumir el crecimiento como profesionales.

RECOMENDACIONES

Antes de colocar Softour disponible en la web se debe realizar un llenado inicial de la base de datos del sistema que establezca metas mínimas de contenido almacenado para que presente una gama de opciones considerable de lugares, eventos, experiencias, alojamientos y gastronomía, a los usuarios turistas haciéndolo más atractivo por presentar variedad de contenido.

Establecer planes de publicidad y mercadeo para dar a conocer la aplicación, generando el incentivo y participación de la mayor cantidad de prestadores de servicios de la zona, haciendo una propuesta de ganar/ganar en donde la región se beneficie del turismo y el turista se beneficie de la región.

Programar respaldos periódicos de la base de datos y de los archivos del sitio Softour, para mantenerlos disponibles en caso de que ocurra una falla de cualquiera tipo, siendo indispensable contar con este respaldo antes de cualquier actualización o cambio de gran impacto, teniendo la posibilidad de revertirlo.

La aplicación web Softour ha pasado por diferentes etapas de desarrollo, pruebas y su pronta puesta en operación, en este proceso se activaron cuentas de usuarios que ya no forman parte del mismo, por lo que el futuro administrador del sitio debe tener presente desactivar, eliminar o limpiar la base de datos cuando esto sea necesario.

Se debe poner en práctica mecanismos o algoritmos para la creación y renovación de claves de usuario de Softour para que sean lo suficientemente seguras e impredecibles.

BIBLIOGRAFÍA

Airbnb. *Empresa y plataforma de software dedicada a la oferta de alojamientos turísticos*. Sitio Web: https://es.wikipedia.org/wiki/Airbnb (05/11/2017).

Alarcón, J. (2012). *Fundamentos de Javascript Y Ajax - Para Desarrolladores Y Diseñadores Web*. Vigo: Krasis Consulting, S.L..

Amaya, J. (2010). *Sistemas de información gerenciales: Hardware, software, redes, Internet, diseño*. Bogota: ECOE Ediciones.

Booch, G., Rumbaugh, J. y Jacobson, I. (2006). El Lenguaje Unificado de Modelado: Guía del Usuario. Madrid: Pearson Educación.

Carrillo, A. (2014). *La intranet social*. Barcelona: Editorial UOC.

Cliente-servidor (s.f.). En Wikipedia. Recuperado el 13 de noviembre de 2017 de: https://es.wikipedia.org/wiki/Cliente-servidor.

Cohen, D. (2000). *Sistemas de información gerencial.* México: Mc Graw-Hill.

Conociendo Github, ¿Qué es GitHub? Sitio Web: http://conociendogithub.readthedocs.io/en/latest/data/introduccion/ (14.11.2017).

De Gregorio, M. (2014). *Impacto de las TICS en el sector turístico. Universidad de Valladolit, Escuela de Ciencias Empresariales y del Trabajo de Soria. Soria, España.* Sitio Web: https://uvadoc.uva.es/bitstream/10324/8459/1/TFG-O% 20435.pdf (12.11.2017).

De Pablos, C., López, J., Martín, S., Medina, S., Montero, A. y Nájera, J. (2006). *Dirección y gestión de los sistemas de información en la empresa: una visión integradora*. Madrid: ESIC Editorial.

Díaz, A. y Sastre, A. (2011). *El disfrute de la naturaleza: aproximaciones a su estudio desde la gestión.* Oviedo: Septem Ediciones.

Duque, A. (2005). *Turismo y Competitividad en Venezuela: Una Aproximación a la Situación Actual Venezolana*. Sitio Web: http://www.saber.ula.ve/bitstream/ 123456789/25091/2/articulo4.pdf

Ferrari, G., Montero, J., Mondéjar, J. y Vargas, M. (2013). *Investigaciones, métodos y análisis del turismo*. Oviedo: Septem Ediciones.

Flavián, C. y Fandos, C. (2011). *Turismo Gastronómico: Estrategias de Marketing y Experiencias de Éxitos*. Zaragoza: Prensas Universitarias de Zaragoza.

Genbetadev. (2014). *MongoDB: qué es, cómo funciona y cuándo podemos usarlo (o no). Sitio Web:* https://www.genbetadev.com/bases-de-datos/mongodb-que-es-como-funciona-y-cuando-podemos-usarlo-o-no (14.11.2017).

Gerardino, F. (2013). *Sistema De Información Web Para El Control De Los Procesos Que Realiza La Comisión De Trabajos De Grado De La Licenciatura En Informática, Del Núcleo De Sucre De La Universidad De Oriente*. Cumaná, Venezuela.

Getbootstrap. (s.f.). Bootstrap es el marco de HTML, CSS y JS más popular para desarrollar primeros proyectos receptivos y móviles en la web. Sitio Web: http://getbootstrap.com/docs/3.3/ (14.11.2017).

Git, (s.f.). En Wikipedia. Recuperado el 14 de noviembre de 2017 de: https://es.wikipedia.org/wiki/Git.

González, R. (2014). *Impacto de la Data Warehouse e Inteligencia de Negocios en el desempeño de las empresas: investigación empírica en Perú, como país en vías de desarrollo. Universitat Ramón Llull, Departamento de política de empresa, dirección recursos humanos y sistemas de información*. Barcelona, España. Sitio Web: http://www.tesisenred.net/bitstream/handle/10803/85876/GONZALES_Tesis%20Doctoral_FV.pdf (11.11.2017).

Graygrids. (2015). *Gentelella - Plantilla de administración de Bootstrap gratis*. Sitio Web: https://graygrids.com/item/gentelella-free-bootstrap-admin-template/ (15.11.2017).

Hurtado, M. y Dean, I. (2013). *Sistema de información gerencial para la gestión documental*. Universidad de Cartagena. Facultad de Ingeniería. Programa de Ingeniería de Sistemas. Cartagena de Indias, Colombia. Sitio Web: http://190.242.62.234:8080/jspui/bitstream/11227/2917/1/Sistema%20de%20informaci%C3%B3n%20gerencial%20.pdf (11.11.2017).

Interviajes. (s.f.). *El turismo. Conceptos y definiciones e importancia actual*. Sitio Web: http://www.interviajes.com.py/admin/img/pdfs/turismo1c-1q4.pdf (12.11.2017).

Jacobson, I. (2004). *El Proceso Unificado de Desarrollo de Software*. Madrid: Pearson Educación.

Jacobson, I., Booch, G. y Rumbaugh, J. (2000). *El Proceso de Desarrollo Unificado.* Madrid: Pearson Educación.

jQuery. (s.f.). En Wikipedia. Recuperado el 14 de noviembre de 2017 de: https://es. wikipedia.org/wiki/JQuery.

Kruchten, P. (1995). *Planos arquitectónicos - El modelo 4 +1 Vista de la Arquitectura de Software. IEEE Software, Instituto de Ingenieros Eléctricos y Electrónicos*. Sitio Web: https://pdfs.semanticscholar.org/efb3/bd19565a682 c278e1ce03f8b0ad4b5232f07.pdf (14/11/2017).

Landeros, R. (2009). *Herramientas CASE: apuntes*. Madrid: El CID Editor.

Larman, C. (2003). *UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado*. México: Prentice Hall.

Laudon, K. (2004). *Sistemas de Información Gerencial: Administración de la Empresa Digital*. México: Prentice Hall.

Magadan, M. y Rivas, J. (2010). *El Patrimonio Cultural como Atractor Turístico*. Oviedo: Septem Ediciones.

Martínez, A. (2012).*Turismo 2.0: Iniciando el Despegue*. Oviedo: Septem Ediciones.

Millán, M. Morales, E. y Pérez, L. (2011). *El Turismo Religioso en la Comunidad Gallega: Estudio del Camino de Santiago*. Oviedo: Septem Ediciones.

MongoDB, (s.f.). En Wikipedia. Recuperado el 14 de noviembre de 2017 de: https://es.wikipedia.org/wiki/MongoDB.

Mongoosejs, (2011). *Elegante modelado de objetos Mongodb para Node.js*.

Sitios Web: http://mongoosejs.com/ (14.11.2017).

Montero, J., Fernández, G., Higueras, M. y García, M. (2010). *El Turismo Idiomático: Una Panorámica.* Oviedo: Septem Ediciones.

Montilva, J. y Barrios, J. (2014). *El Marco Metodológico Blue Watch para el Desarrollo de Software.* Mérida: Proyecto METHODIUS Financiado por FONACIT.

Montilva, J., Barrios, J. (2013). *White_Watch: Un Método para el Desarrollo de Proyectos Pequeños de Software.* Mérida: Proyecto METHODIUS Financiado por FONACIT.

Montilva, J., Barrios, J. y Rivero, M. (2008). *Gray Watch*: Método de Desarrollo de Software para Aplicaciones Empresariales Mérida: Proyecto METHODIUS Financiado por FONACIT.

Node.js. (s.f.). Node.js. En Wikipedia. Recuperado el 15 de noviembre de 2017 de: https://es.wikipedia.org/wiki/Node.js.

Nodejs. (s.f.). Node.js. Sitio Web: https://nodejs.org/en/ (15.11.2017).

Npmjs, (2010). *Mongoose.* Sitio Web: https://www.npmjs.com/package/mongoose (14.11.2017).

Oviedo, E. (2015). *Lógica de programación orientada a objetos*. Bogota: ECOE Ediciones.

Powell, T. (2001). *HTML 4. Manual de referencia*. Madrid: McGraw Hill.

Pressman, R. (2005). *Ingeniería del Software. Un enfoque práctico.* México: McGraw-Hill/Interamericana Editores S.A. de C.V.

Protocolo seguro de transferencia de hipertexto (s.f.). En Wikipedia. Recuperado el 14 de noviembre de 2017 de: https://es.wikipedia.org/wiki/Protocolo_seguro_de_transferencia_de_hipertexto.

Rumbaugh, J., Booch, G. y Jacobson, I. (2007). *El Lenguaje Unificado de Modelado: Manual de Referencia*. Madrid: Pearson Educación.

Sánchez, C. (2004). *ONess: un proyecto* Open Source *para el negocio textil mayorista desarrollado con tecnologías* Open Source *innovadoras*. La Coruña: Universidade da Coruña Facultade de Informática, Departamento de Tecnoloxías da Información e as Comunicacións.

Sancho, A. (1998). *Unidad 1 El turismo como actividad economica. En OMT Organizacion Mundial del Turismo. Introduccion al Turismo. OMT Organizacion Mundial del Turismo*. Sitio Web: https://es.wikipedia.org/wiki/Turismo#cite_note-2 (10/11/2017).

Schmuller, J. (2000). *Aprendiendo UML en 24 horas*. México: Prentice Hall.

Sistema de gestión de bases de datos (s.f.). En Wikipedia. Recuperado el 14 de noviembre de 2017 de: https://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos.

Sommerville, I. (2005). *Ingeniería de Software*. Madrid: Pearson Educación.

Sparx Systems, (2017). *Enterprise Architect - Herramienta de diseño UML*. Sitio Web: http://www.sparxsystems.com.ar/products/ea.html. (12.11.2017).

Sublime Text. (s.f.). En Wikipedia. Recuperado el 15 de noviembre de 2017 de: https://es.wikipedia.org/wiki/Sublime_Text.

Turismo de Negocios. (s.f.). En Wikipedia. Recuperado el 16 de diciembre de 2017 de: https://es.wikipedia.org/wiki/Turismo#Turismo_de_negocios.

Turismo Religioso. (s.f.). En Wikipedia. Recuperado el 16 de diciembre de 2017 de: https://es.wikipedia.org/wiki/Turismo_religioso.

Turismo Sustentable. (s.f.). En Wikipedia. Recuperado el 16 de diciembre de 2017 de: https://es.wikipedia.org/wiki/Turismo#Turismo_cultural.

Tamayo y Tamayo, M. **(**2001). *El Proceso de Investigación Científica*. Tercera edición. Ediciones Limusa. S.A. México.

Tiziana, L., Ferrari, G. y Secondi, L. (2010). *Un análisis CGE del impacto del turismo en el sistema económico de Cerdeña*. Oviedo: Septem Ediciones.

Valdes, L. (2002). *La re-evolución empresarial del siglo XXI*. Bagota: Editorial Norma.

Valle, M. (2009). *Plan De Negocio Para Facilitadora De Servicios Turísticos En Nicaragua*. Universidad De Chile Facultad De Ciencias Fisicas Y Matematicas Departamento De Ingenieria Industrial. Santiago de Chile, Chile. Sitio Web: http://repositorio.uchile.cl/tesis/uchile/2009/valle_m/sources/valle_m.pdf (12.11.2017).

Whitten, J., Bentley, L. y Barlow V. (2005). *Análisis y Diseño de Sistemas de Información*. México: McGraw-Hill/Interamericana Editores, S.A. de C.V.

APÉNDICES

INDICE

Página

Apéndice A. Planificación del tiempo y organigrama 222
Apéndice B. Organigrama del equipo de desarrollo del sistema. 225
Apéndice C. Ciclo de Versión Administrador: Iteración Ingresar Lugar. 226
Apéndice D. Ciclo de Versión Administrador: Iteración Listar Lugar. 241
Apéndice E. Ciclo de Versión Administrador: Iteración Consultar Lugar. 249
Apéndice F. Ciclo de Versión Administrador: Iteración Modificar Lugar. 258
Apéndice G. Ciclo de Versión Administrador: Iteración Eliminar Lugar. 272
Apéndice H. Ciclo de Versión Administrador: Iteración Agregar Evento. 281
Apéndice I. Ciclo de Versión Administrador: Iteración Listar Eventos. 302
Apéndice J. Ciclo de Versión Administrador: Iteración Consultar Evento. 310
Apéndice K. Ciclo de Versión Administrador: Iteración Modificar Evento. 320
Apéndice L. Ciclo de Versión Administrador: Iteración Eliminar Evento. 333
Apéndice M. Ciclo de Versión Administrador: Iteración Ingresar Experiencia. 342
Apéndice N. Ciclo de Versión Administrador: Iteración Listar Experiencia. ... 362
Apéndice O. Ciclo de Versión Administrador: Iteración Consultar Experiencia. 370
Apéndice P. Ciclo de Versión Administrador: Iteración Modificar Experiencia. 380
Apéndice Q. Ciclo de Versión Administrador: Iteración Eliminar Experiencia. 395
Apéndice R. Ciclo de Versión Administrador: Iteración Ingresar Alojamiento. 405
Apéndice S. Ciclo de Versión Administrador: Iteración Listar Alojamiento. ... 429
Apéndice T. Ciclo de Versión Administrador: Iteración Consultar Alojamiento. 437
Apéndice U. Ciclo de Versión Administrador: Iteración Modificar Alojamiento. 447
Apéndice V. Ciclo de Versión Administrador: Iteración Eliminar Alojamiento. 463
Apéndice W. Ciclo de Versión Administrador: Iteración Ingresar Gastronomía. 471
Apéndice X. Ciclo de Versión Administrador: Iteración Modificar Gastronomía.

..... 486

Apéndice Y. Ciclo de Versión Administrador: Iteración Listar Gastronomía. . 501

Apéndice Z. Ciclo de Versión Administrador: Iteración Visualizar Gastronomía. 510

Apéndice AA. Ciclo de Versión Administrador: Iteración Eliminar Gastronomía 519

Apéndice BB. Ciclo de Versión Administrador: Iteración Ingresar Prestador de Servicios..... 528

Apéndice CC. Ciclo de Versión Administrador: Iteración Elaborar Listado de Prestador de Servicios. 542

Apéndice DD. Ciclo de Versión Administrador: Iteración Eliminar Prestador de Servicios..... 548

Apéndice EE. Ciclo de Versión Administrador: Iteración Elaborar Listado Turista. 556

Apéndice FF. Ciclo de Versión Administrador: Iteración Consultar Detalle Turista. 562

Apéndice GG. Ciclo de Versión Administrador: Iteración Eliminar Turista. 569

Apéndice HH. Ciclo de Versión Prestador de Servicios: Iteración Ingresar Lugar 577

Apéndice II. Ciclo de Versión Prestador de Servicios: Iteración Modificar Experiencia. 596

Apéndice JJ. Ciclo de Versión Prestador de Servicios: Iteración Listar Gastronomía 611

Apéndice KK. Ciclo de Versión Prestador de Servicios: Iteración Consultar Evento. 619

Apéndice LL. Ciclo de Versión Prestador de Servicios: Iteración Eliminar Alojamiento. 629

Apéndice NN. Consultar Evento por Turista 651

Apéndice OO. Consultar Experiencia por Turista. 662

Apéndice QQ. Consultar Lugar por Turista..... 685

Apéndice RR. Inicio Sesión Turista..... 701

Apéndice SS. Regsitrar Turista. 709

Apéndice TT. Plantilla de Volére para la documentación de requisitos y descripción de casos..... 717

Apéndice VV. Diagrama de datos. 735

Apéndice WW. Diagrama de componentes. 736
Apéndice XX. Diagrama de reglas del negocio. 738

Apéndice A. Planificación del tiempo y organigrama

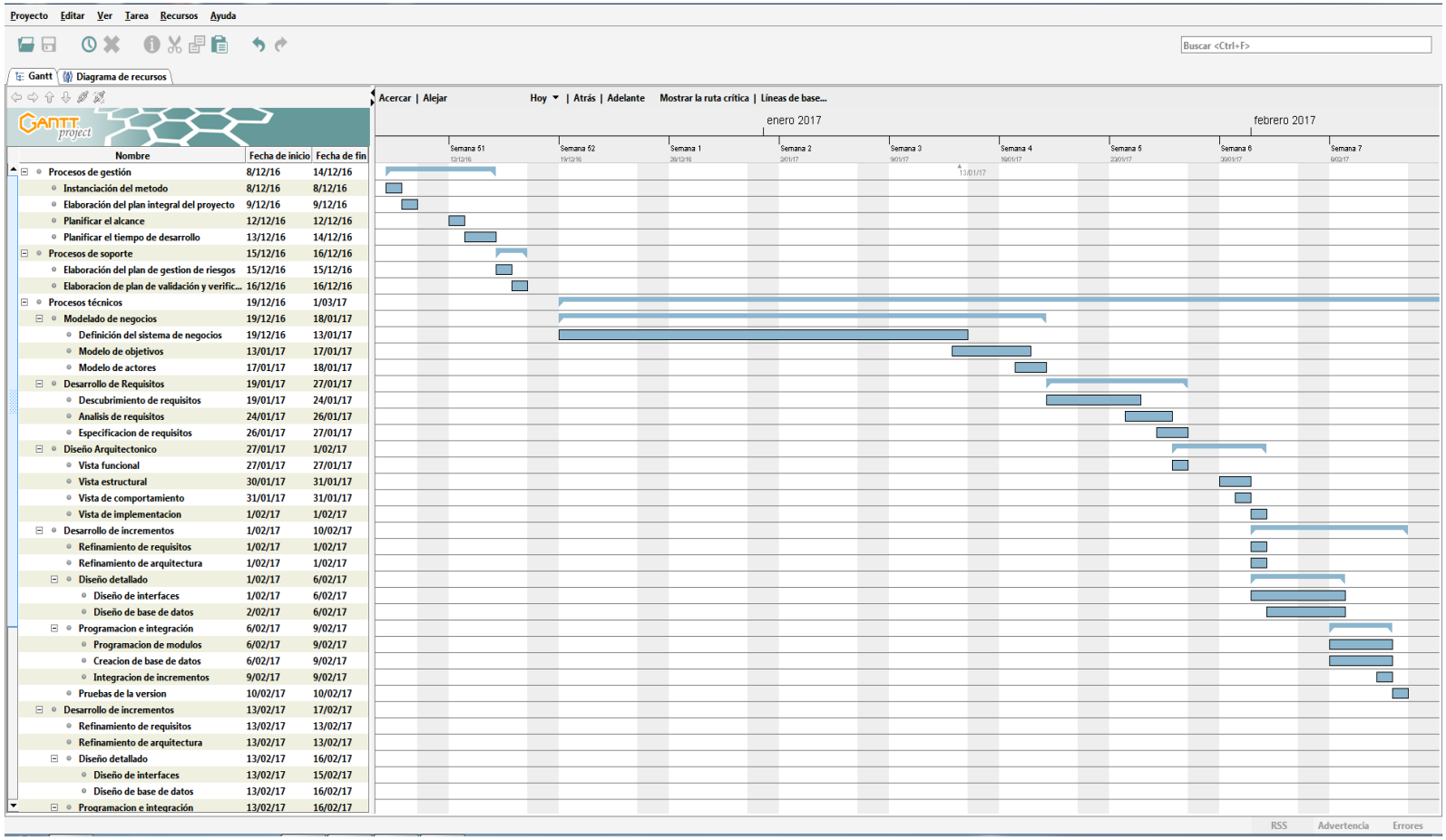


Figura A.1. Planificación del tiempo.

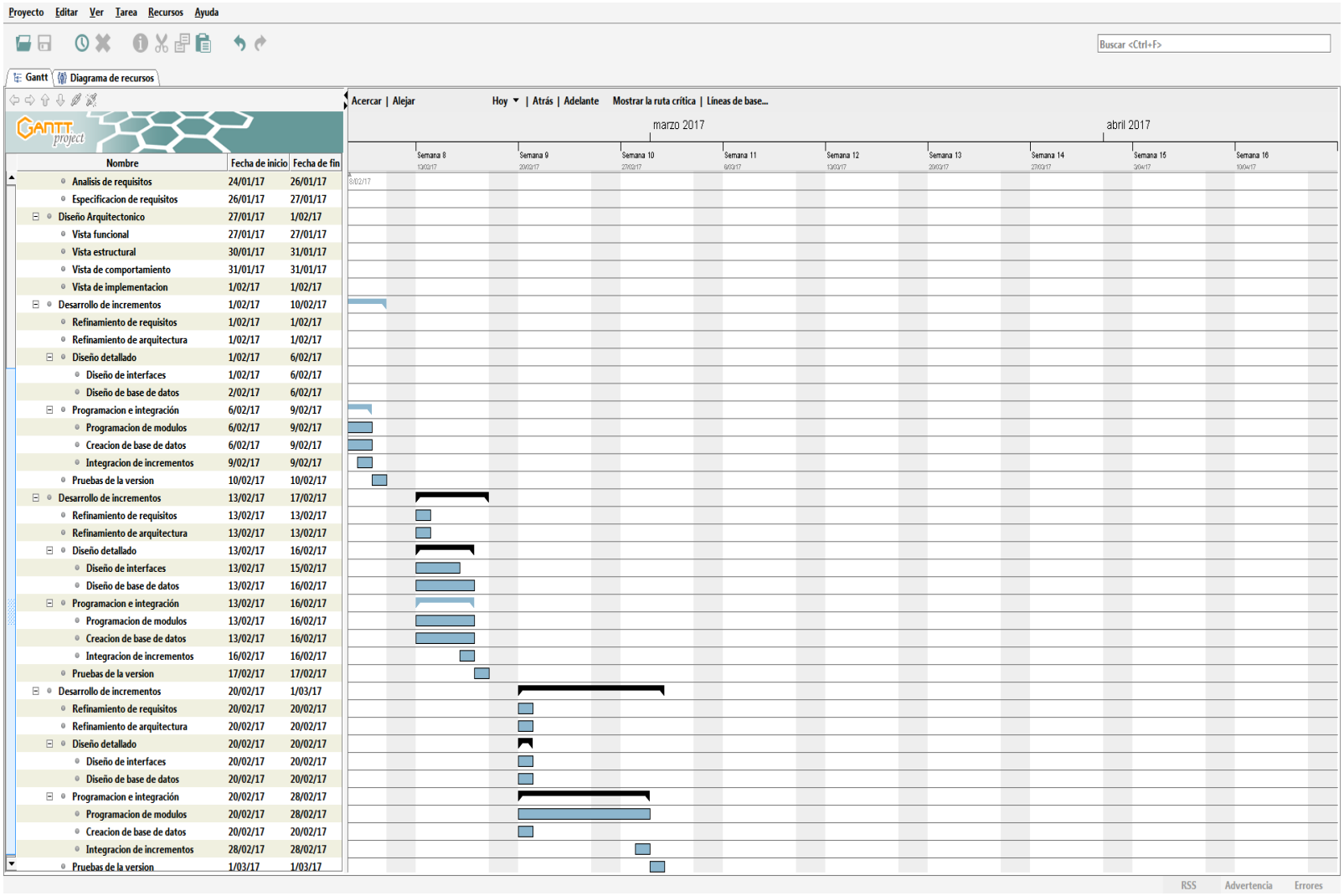


Figura A.2. Planificación del tiempo.

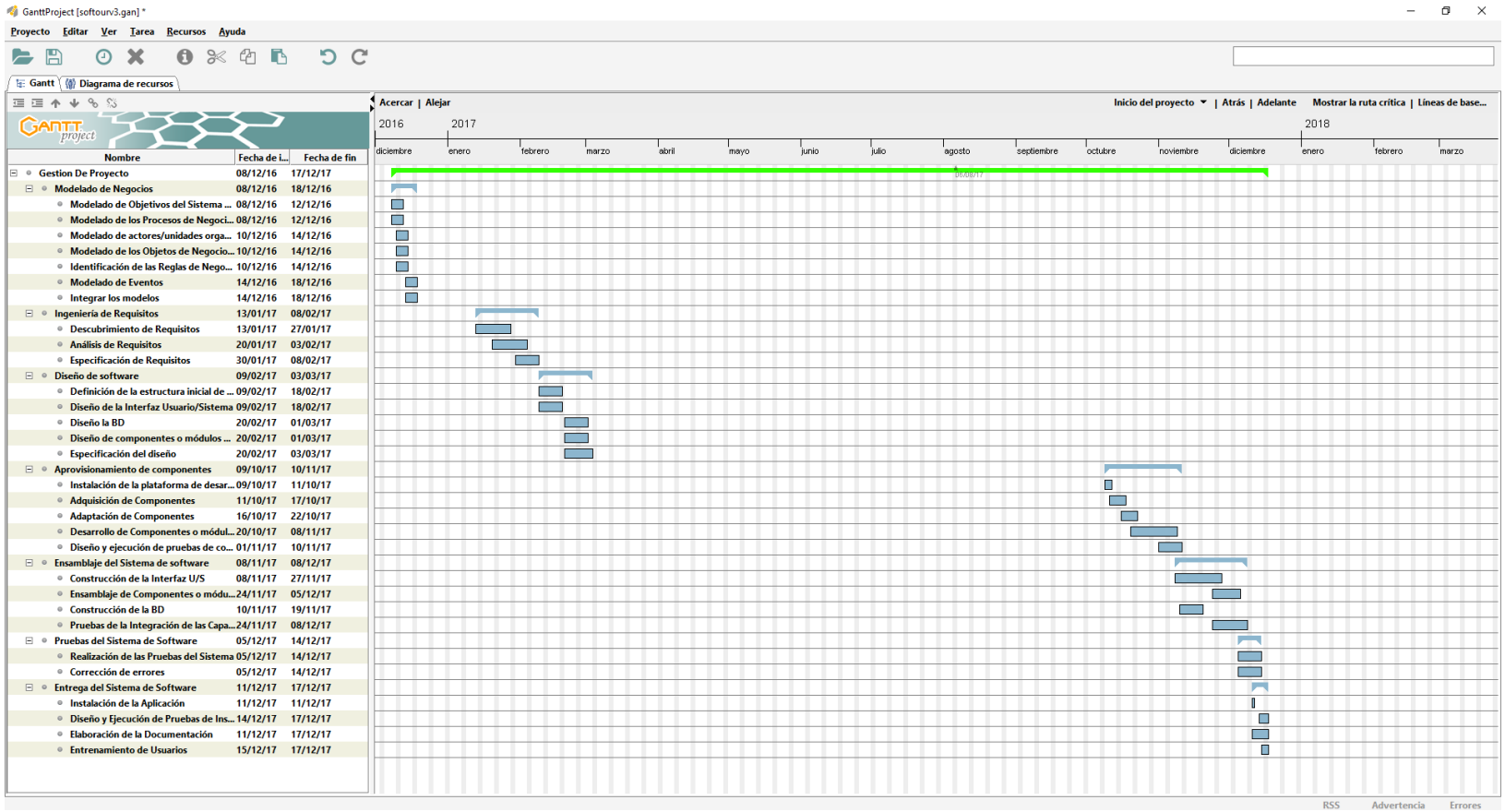


Figura A.3. Planificación del tiempo.

Apéndice B. Organigrama del equipo de desarrollo del sistema.

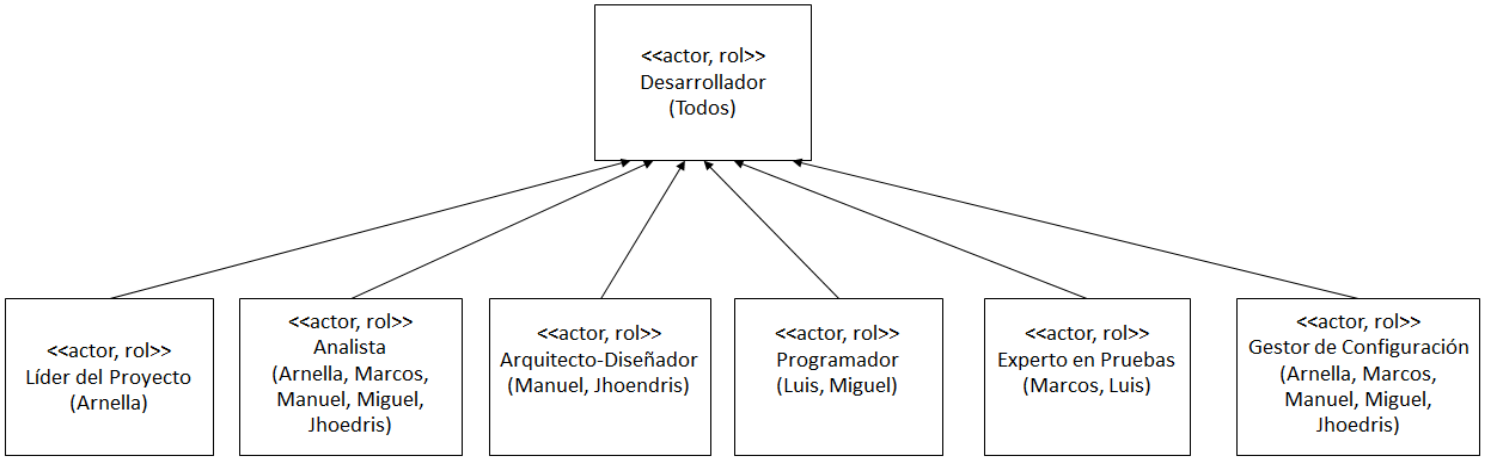


Figura B.1. Organigrama aplicado por el equipo durante el desarrollo del sistema.

Apéndice C. Ciclo de Versión Administrador: Iteración Ingresar Lugar.

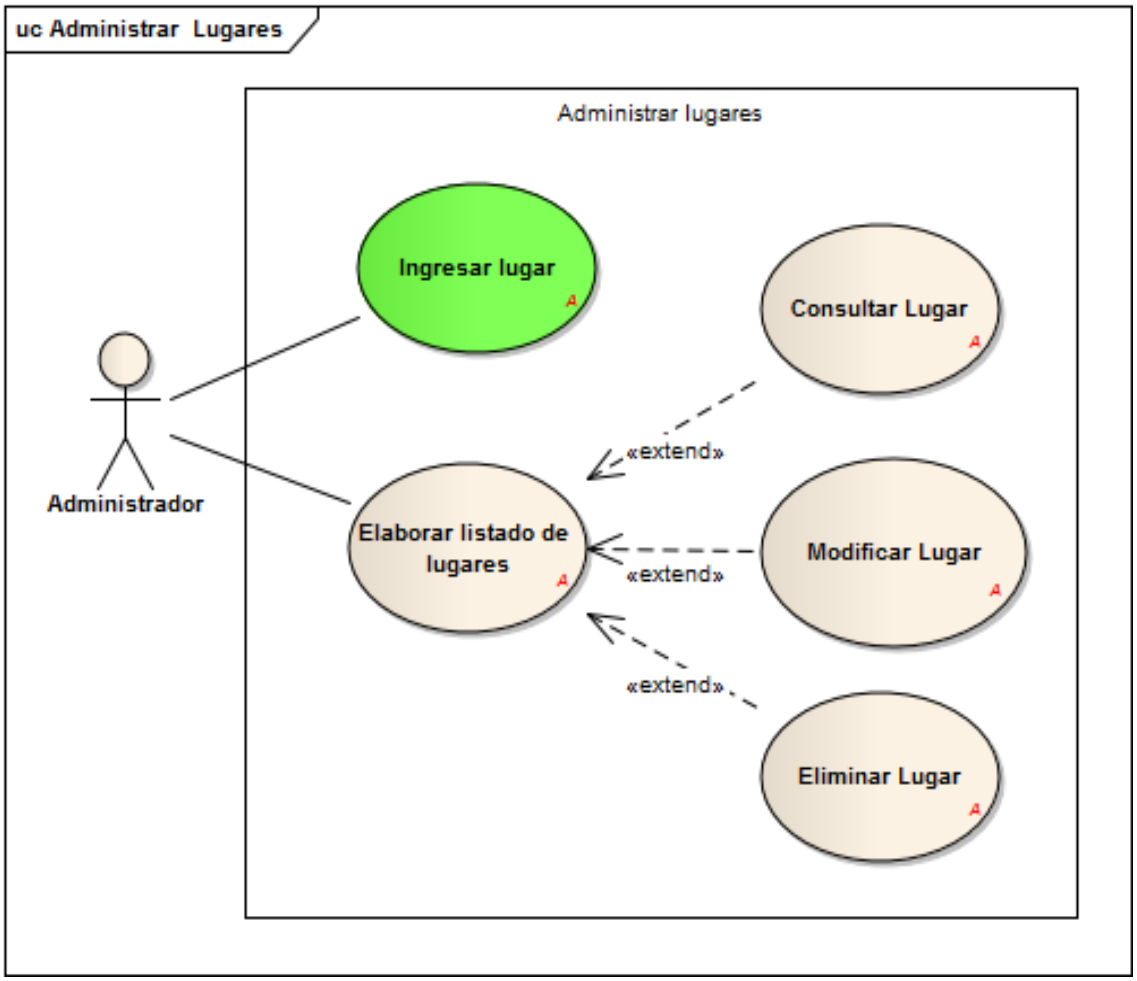


Figura C. 1. Caso de uso: Ingresar Lugar administrador.

Tabla C. 1. Descripción Caso de uso Ingresar Lugar por usuario Administrador.

Nombre del Caso de uso	Ingresar lugar.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en ingresar los atributos del lugar de manera satisfactoria.	
Precondición	Para ingresar un lugar, no debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Ingresar lugar”.
	2	El sistema muestra el formulario correspondiente.
	3	El Administrador ingresar los atributos del lugar.
	4	El sistema almacena los atributos del lugar y finaliza el caso de uso.
Postcondición	Se ingresa los atributos del lugar en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no ingresar el lugar y selecciona cancelar el caso de uso, seleccionando otra opción del sistema.
	3.2.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. Para que un lugar pueda ser ingresado no debe estar registrado en la base de datos del sistema.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

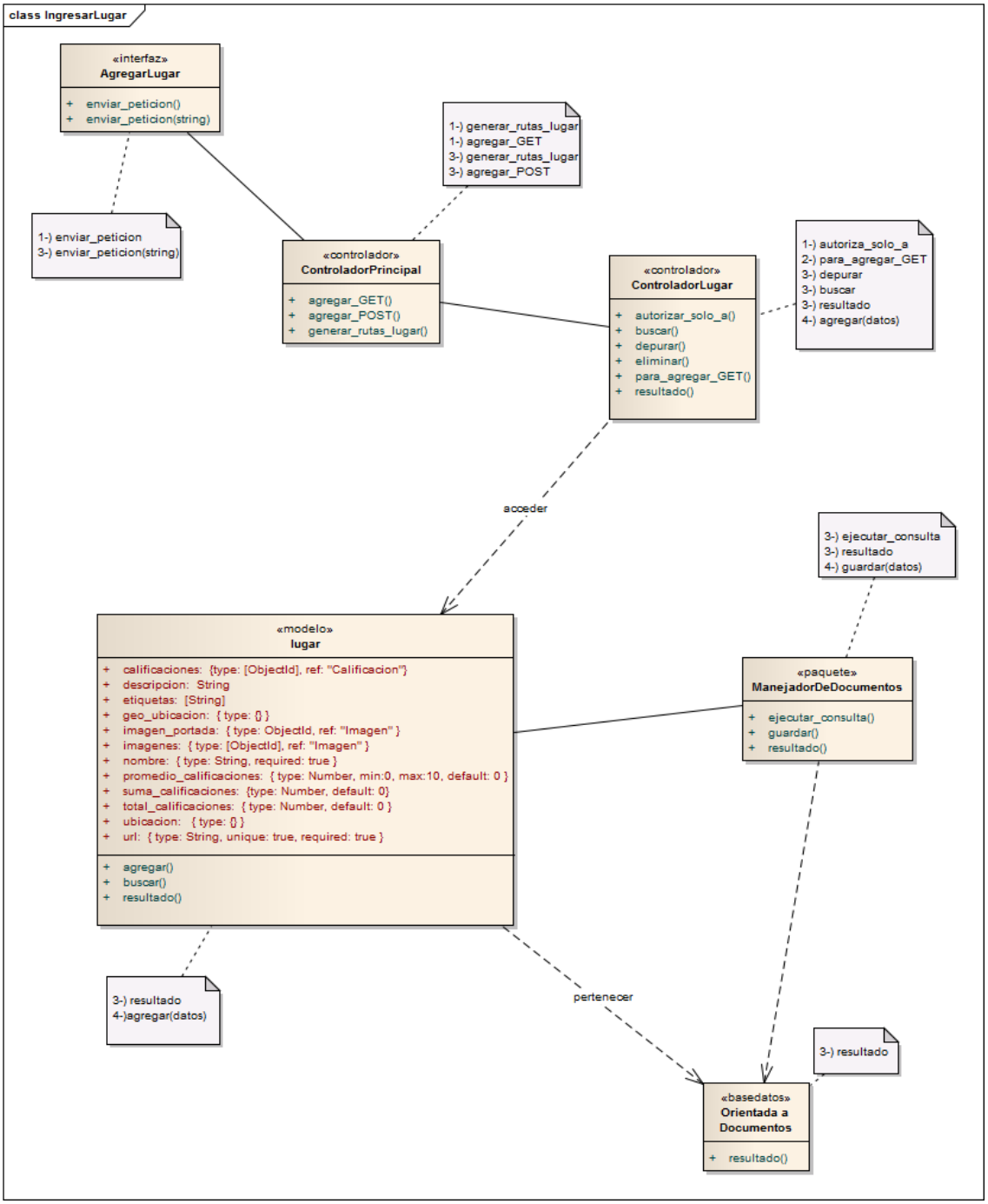


Figura C. 2. Diagrama de Clases: Ingresar Lugar por usuario Administrador.

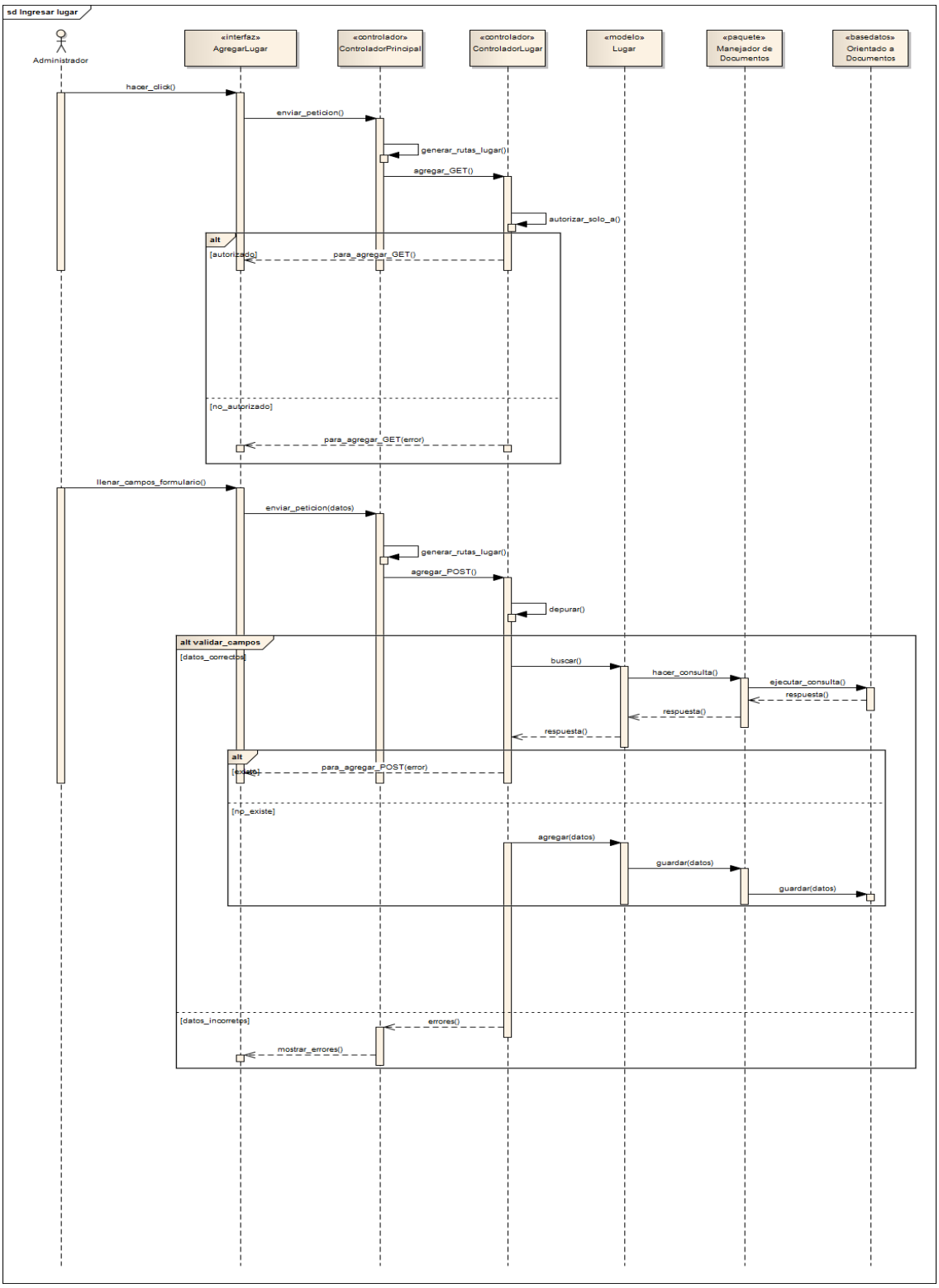


Figura C. 3. Diagrama de secuencia: Ingresar Evento por usuario Administrador.

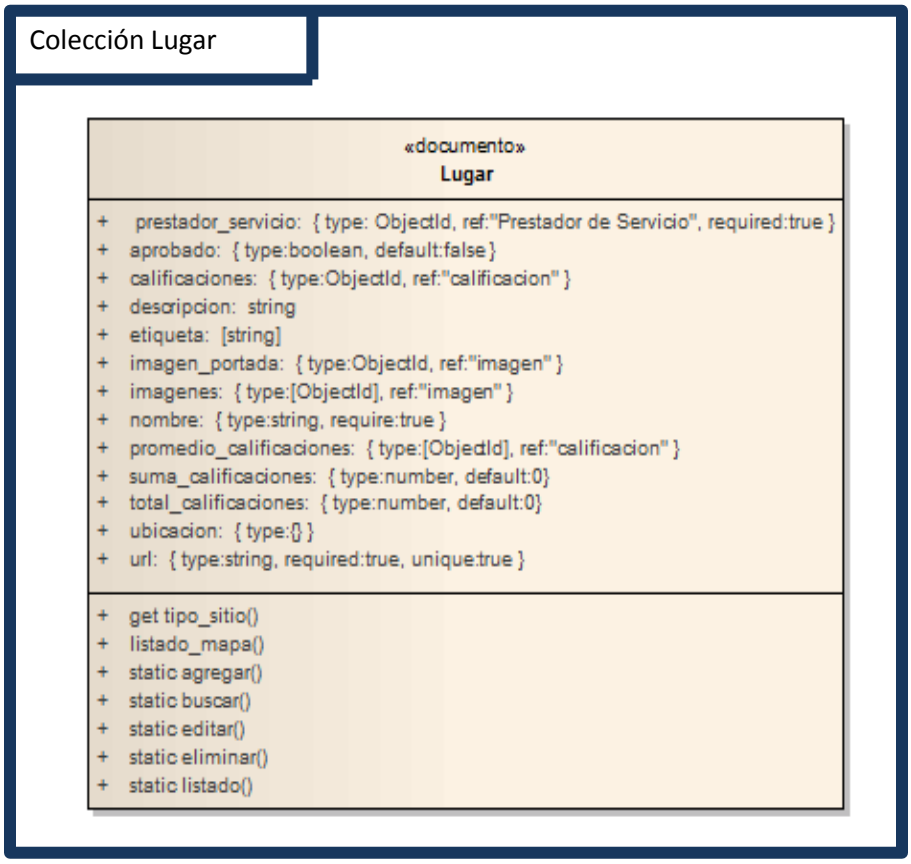


Figura C. 4. Colección de lugares: ingresar lugar por usuario Administrador.

A continuación, se muestran los modelos de plantillas para el proceso de ingresar lugar.

Figura C. 5. Plantilla de interfaz 1: Ingresar Lugar por usuario Administrador.

Figura C. 6. Plantilla de interfaz 2: Ingresar Lugar por usuario Administrador.

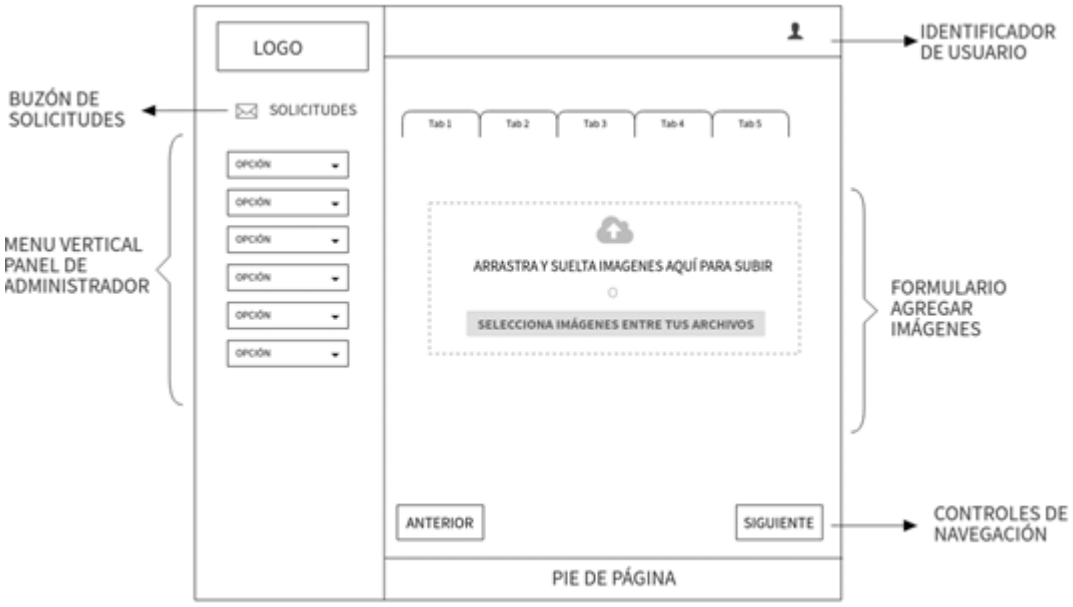


Figura C. 7. Plantilla de interfaz 3: Ingresar Lugar por usuario Administrador.

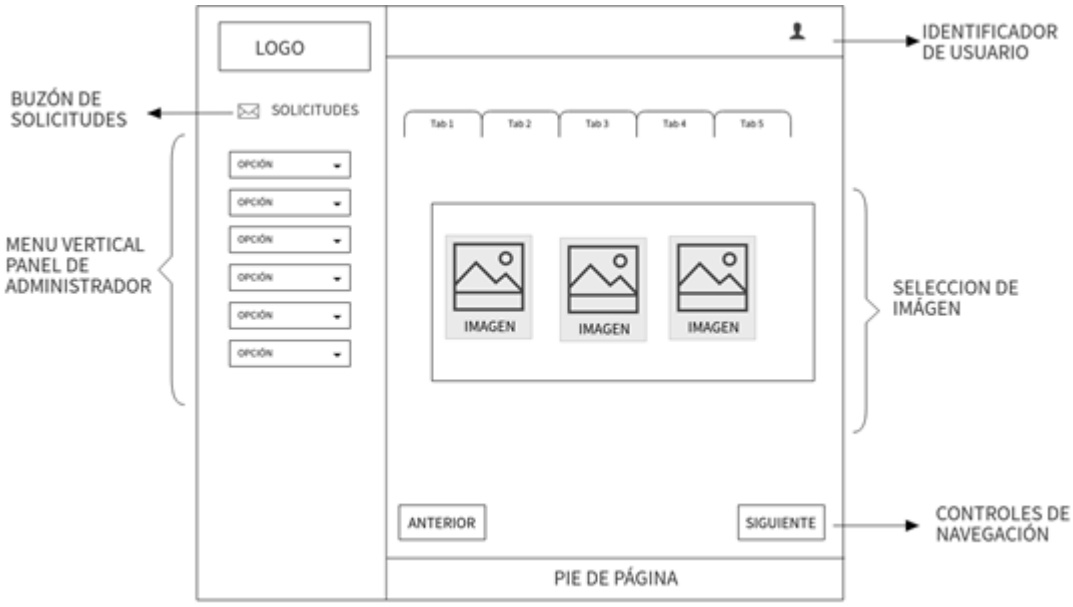


Figura C. 8. Plantilla de interfaz 4: Ingresar Lugar por usuario Administrador.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Agregar Evento y el sistema responde mostrando el formulario correspondiente.

Figura C. 9. Interfaz 1 de proceso de Agregar Lugar por usuario Administrador.

El administrador ingresa los atributos del lugar, lo cual se visualiza en diferentes pestañas: Información, Contacto, Subir fotos y Elige foto de portada.

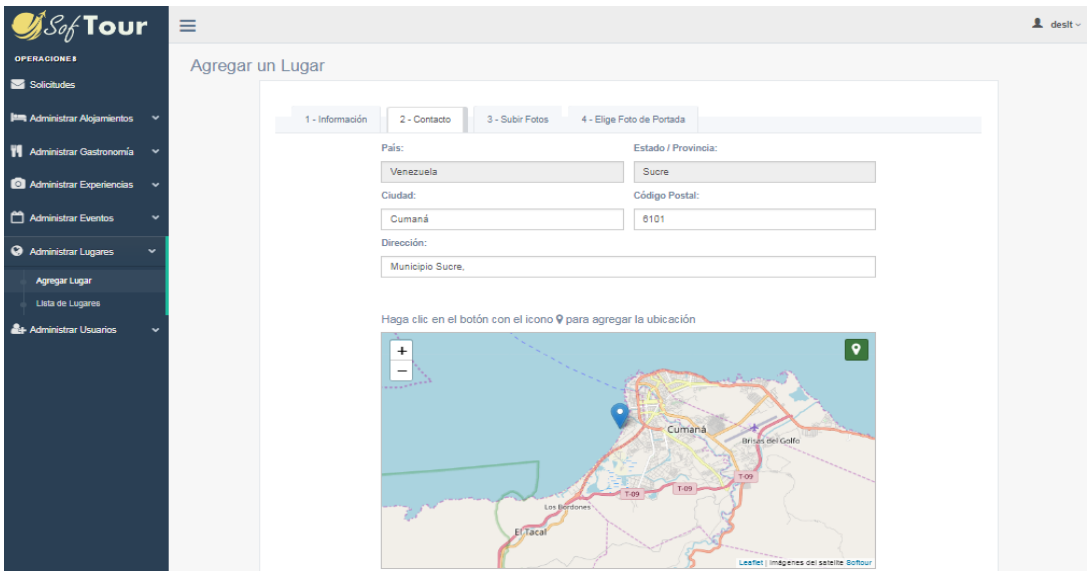


Figura C. 10. Interfaz 2 de proceso de Agregar Lugar por usuario Administrador.

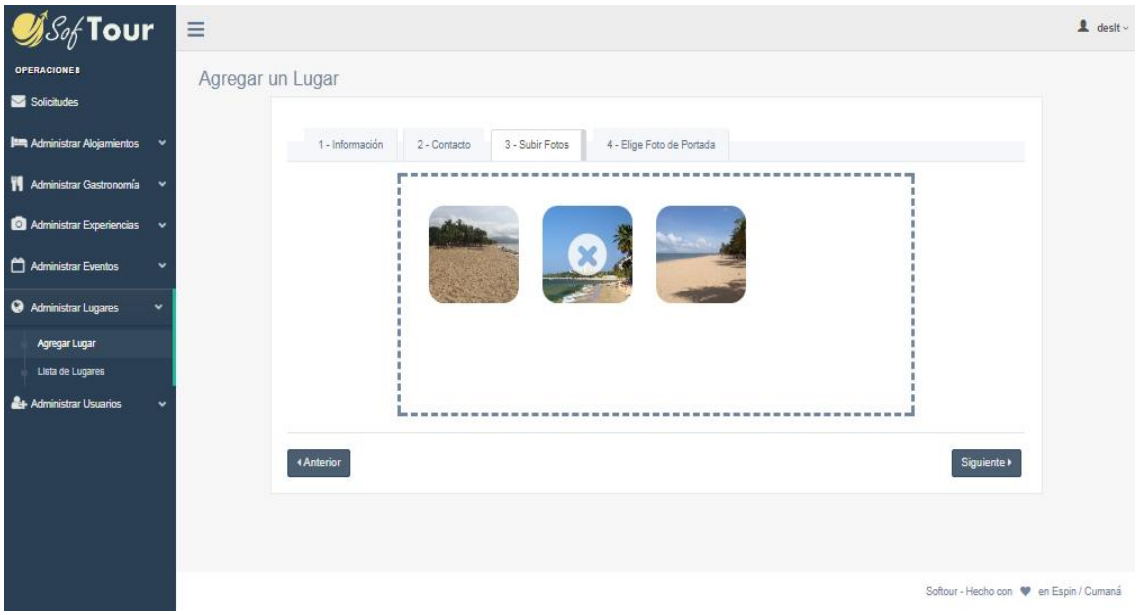


Figura C. 11. Interfaz 3 de proceso de Agregar Lugar por usuario Administrador.

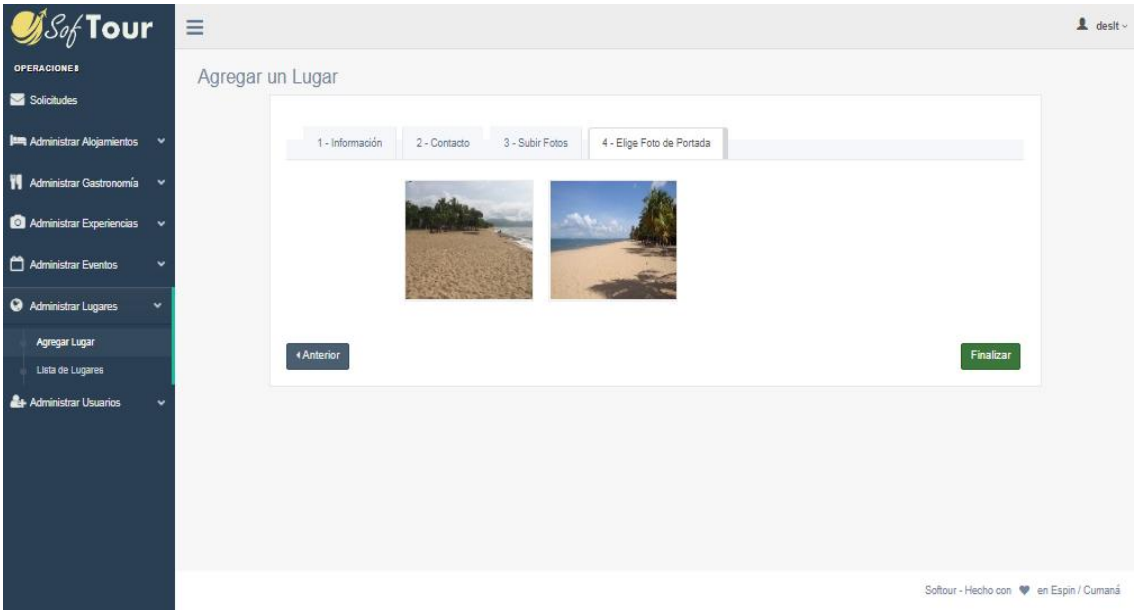


Figura C. 12. Interfaz 4 de proceso de Agregar Lugar por usuario Administrador.

Código desarrollado para la funcionalidad Agregar Evento por usuario Administrador.

```
ingresar_POST(req, res, next) {
    this._validar.campos(req.body, false, (error, resultados) => {
        if(error) next(error);
        res.locals.datos = resultados;

        if (res.locals.datos.error) {
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;

            this._respuesta.para_ingresar_POST(req, res);
        }
        else {
            res.locals.datos.value.geo_ubicacion = {
                "type": "Point",
                "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
            };
            res.locals.datos.value.ubicacion = {
                "direccion": res.locals.datos.value.direccion,
                "pais": res.locals.datos.value.pais,
                "estado": res.locals.datos.value.estado,
                "ciudad": res.locals.datos.value.ciudad,
                "codigo_postal": res.locals.datos.value.codigo_postal
            };

            this._Lugar.buscar({"url": res.locals.datos.value.url}, (error, lugar) => {
                if (lugar) {
                    res.locals.datos.error = { "details": [ {
                        "path": "nombre",
                        "message": 'Ya existe un lugar con este nombre.'
                    } ] };
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_ingresar_POST(req, res);
                }

                else {
                    let lugar = this._fabricar.lugarTipo(res.locals.datos.value.tipo_lugar);
                    if(req.session.passport.user.tipo_usr == "Administrador" && lugar.__t)
                        res.locals.datos.value.aprobado = true;

                    lugar.ingresar(res.locals.datos.value, (error, creado) => {
                        if(error) {
                            res.locals.datos.error = error;

                            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                            this._respuesta.para_ingresar_POST(req, res);
                        }

                        res.locals.datos = {"value": creado};
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_ingresar_POST(req, res);
                    });
                }
            });
        }
    });
}
```

Figura C. 13.Código desarrollado para el proceso Ingresar Lugar por usuario Administrador.

Prueba ingresar lugar

Casos de prueba para la evaluación de la función que realiza el usuario administrador etiquetada como ingresar lugar, se realizó de la siguiente manera:

La prueba se realizó de manera manual, el equipo de pruebas diseño los datos válidos y no válidos a ingresar y evaluó que el módulo cumpliera a cabalidad.

Se tomaron en cuenta las siguientes clases de equivalencia:

Tabla C. 2. Clases de equivalencia Ingresar Lugar por administrador

Condición de entrada	Tipo	Clases de equivalencias válidas	Clases de equivalencias no válidas
Nombre de lugar	Alfabético	1: Cadena de caracteres.	2: Campo en blanco 3:Símbolos
Descripción	Alfanumérico	4:Cadena de caracteres, números.	2: Campo en blanco 5:Cadena de caracteres menores a 150.
Etiquetas	Alfanumérico	4: Cadena de caracteres, números.	
País	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Estado/Provincia	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Ciudad	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Código postal	Numérico	6: Números	1: Cadena de caracteres 3: Símbolos 7:Letras
Dirección	Alfanumérico	1:Cadena de caracteres, números.	2: Campo en blanco
Fotos	Imagen	8: JPG, GIF, PNG	2: Campo en blanco
Elegir foto de perfil	Selección		

Tabla C. 3. Casos de prueba Ingresar Lugar por Administrador.

Condición de entrada	A1	A2	A3
Nombre de lugar	Parque	¡Museo Humbolt!	Monumento
Descripción	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.		Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Etiquetas	Parque; distracción; diversión	Museo; Teléfono; 123	Paseo; visitas;
País	Venezuela	México	
Estado/Provincia	Sucre		Amazonas
Ciudad	Cumaná		4568
Código postal	6101	5555	nchd
Dirección	Calle Mariño, sector 1		
Fotos	PNG, JPG	Doc, PDF	
Elegir foto de perfil	Selección	Selección	
Clases Equivalencia	1,4,6,8	1,2,3	1,2,4,5,7
Resultado	Registro satisfactorio	Registro no satisfactorio por datos no válidos para el sistema.	Registro no satisfactorio por datos no válidos para el sistema.

Para el caso A1 se obtuvo un registro satisfactorio, se llenaron las casillas de acuerdo a lo requerido por el sistema y la función cumplió su fin.

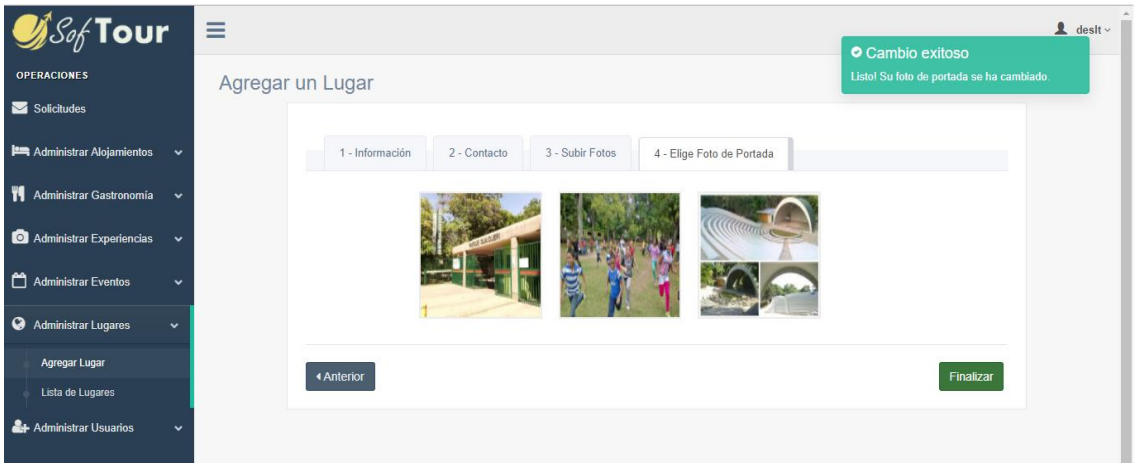


Figura C. 14. Inicio caso de prueba A1 Ingresar Lugar por usuario Administrador.

Para el caso A2 el registro no se pudo completar, desde el primer paso de agregar el lugar el sistema rechazo el nombre por contener signos de exclamación, documentándolo como no valido, comprobando que la validación se cumple.

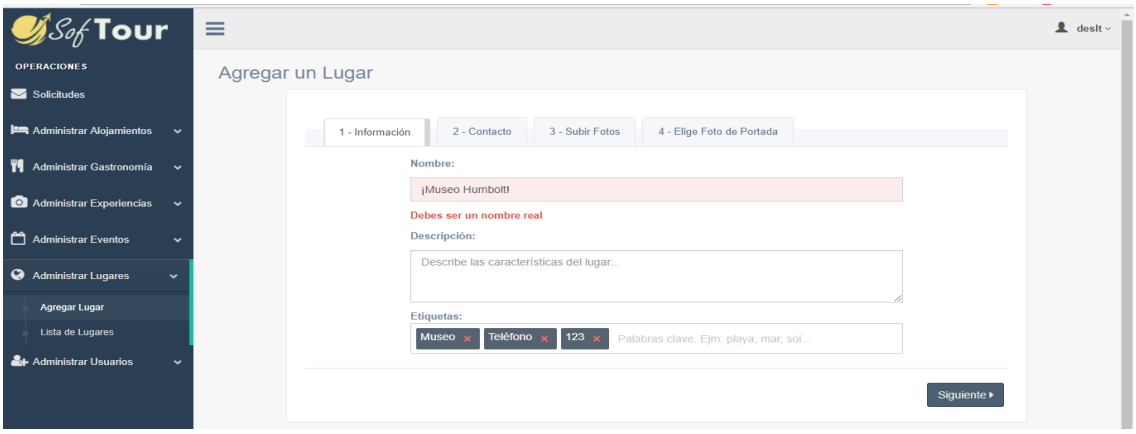


Figura C. 15. Inicio caso de prueba A2 Ingresar Lugar por usuario Administrador.

Para el caso de pruebas A3 el registro no fue satisfactorio, inclusive no se pudo avanzar del segundo paso de registro debido a que las validaciones del sistema evitaron continuar ya que se introducía un dato inválido en uno de los campos, se trató de introducir letras en donde el campo solo permite números.

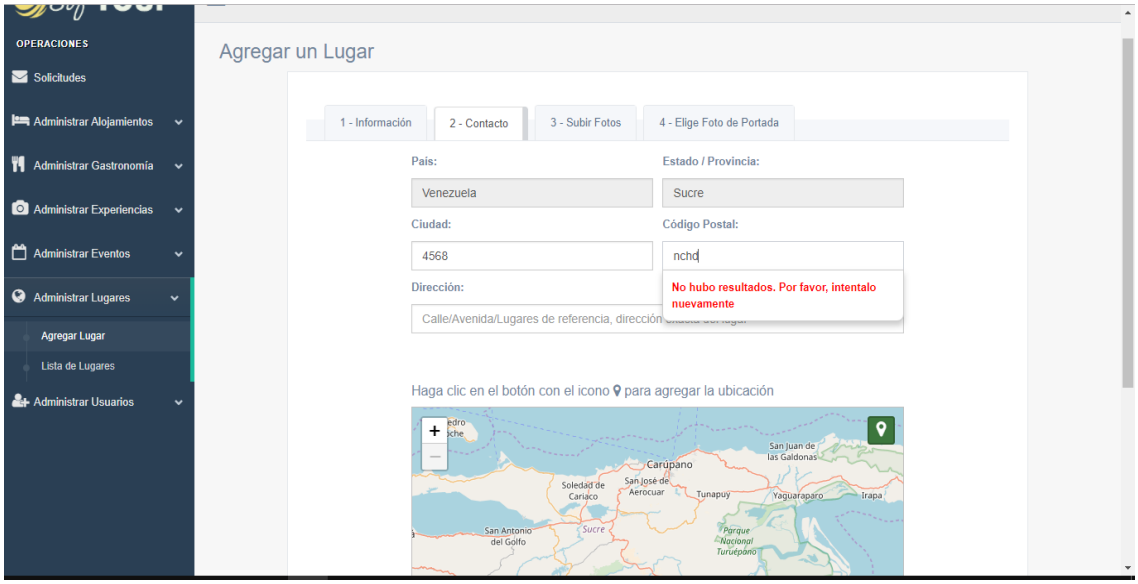


Figura C. 16.Inicio caso de prueba A3 Ingresar Lugar por usuario Administrador.

Conclusión de la prueba

El módulo demostró estar validado en todos sus campos, evitando que se llenaran campos con datos inválidos y así evitar que la base de datos sea llenada con información errónea, pero se realizó la observación al equipo de desarrollo del caso A2, ya que pueden existir nombre de lugares que puedan llevar algún número en su nombre.

Prueba de interfaz y usabilidad

Cabe señalar que desde el comienzo del desarrollo del software una de las ideas principales era diseñar una interfaz que se llevara de la mano con los principios de usabilidad, para así asegurar el éxito del sistema.

A continuación, se muestra una captura de pantalla d la función Agregar Lugar al cual se le realizó un análisis detallado para corroborar cumple con los principios heurísticos de usabilidad.

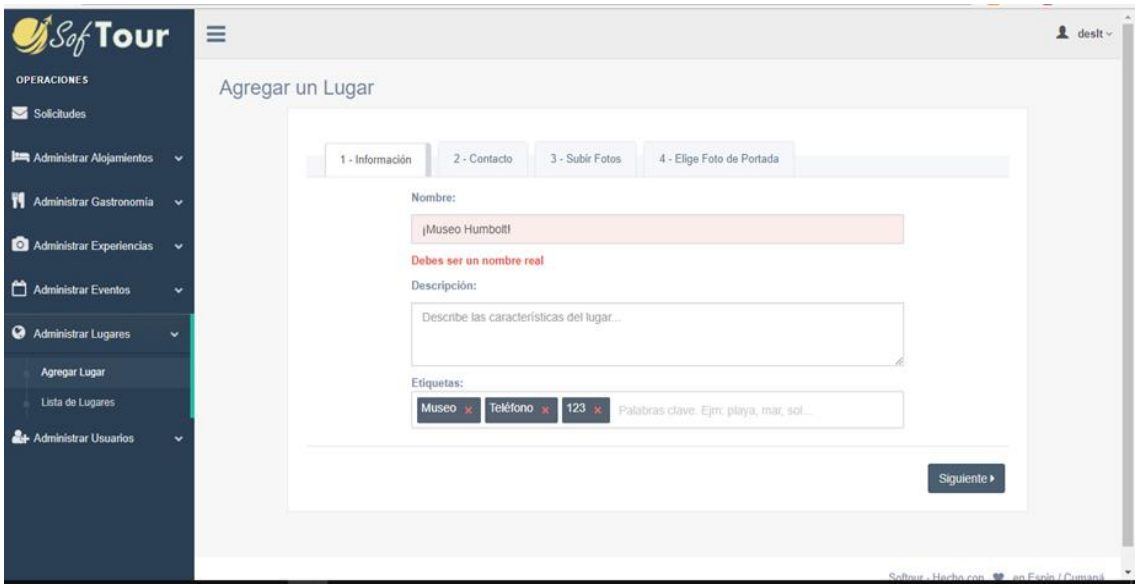


Figura C. 17.Ingresar Lugar por usuario Administrador.

Como se puede constatar en la imagen, el diseño es minimalista, utiliza palabras sencillas que se adaptan al ámbito, muestra metáforas para el reconocimiento de funciones, enumera los pasos a seguir para mantener al usuario informado en qué punto se encuentra y mediante sombreado indica en que área del panel está ubicado, además de señalizar en tonalidad roja cuando se encuentra un error en los datos.

Apéndice D. Ciclo de Versión Administrador: Iteración Listar Lugar.

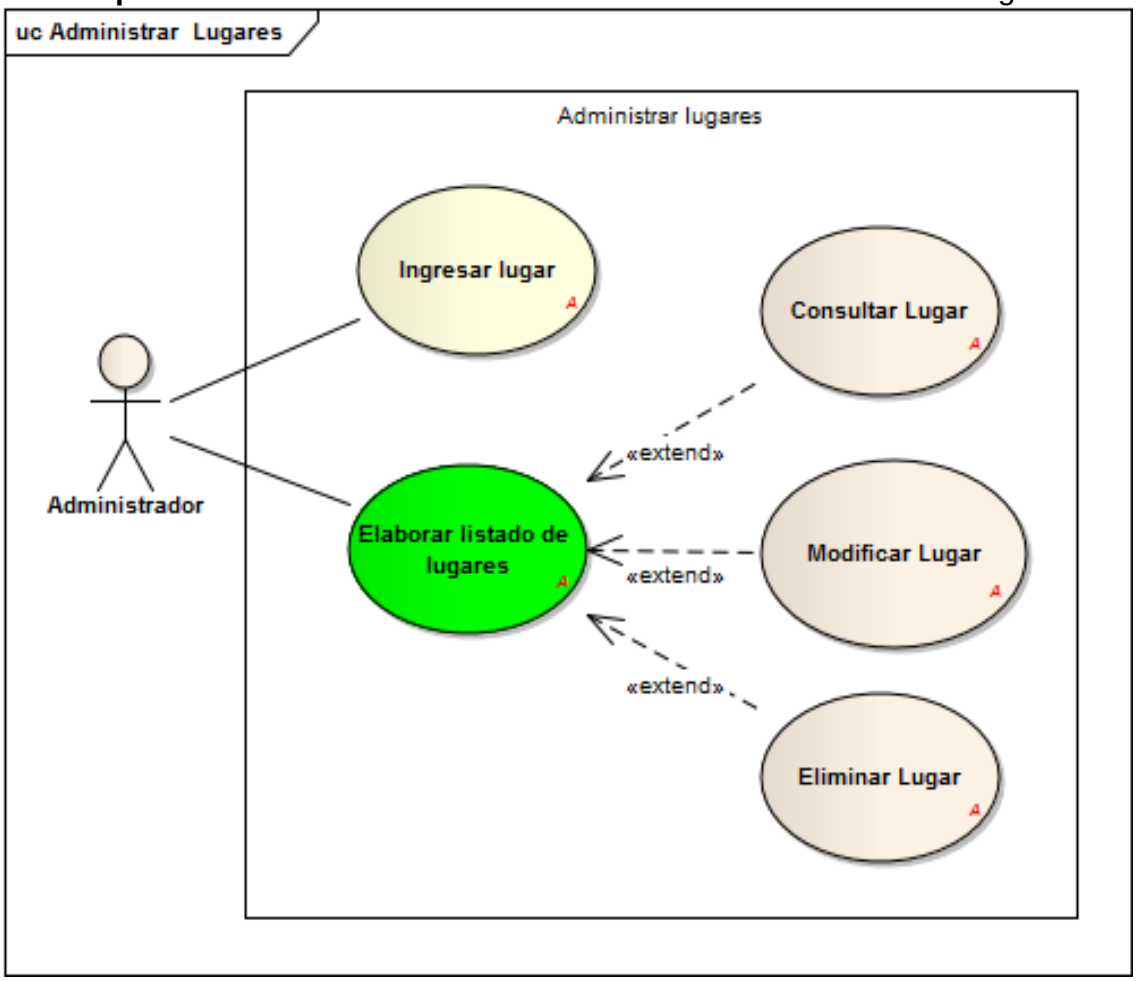


Figura D. 1.Caso de uso: Elaborar Listado de Lugar por Administrador.

Tabla D. 1. Descripción Caso de uso Elaborar Listado de Lugar por usuario Administrador.

Nombre del Caso de uso	Elaborar listado de lugar	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en elaborar el listado de los lugares que actualmente están registrados en el sistema.	
Precondición	Para elaborar el listado de los lugares, al menos debe existir uno registrado en la BD del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador de Servicios selecciona la opción “Listado de lugares”.
	2	El sistema realiza la búsqueda de los lugares registrados en la BD del sistema y muestra un listado de manera ordenada.
	3	El Administrador visualiza el listado de los lugares y selecciona aceptar.
	4	El sistema finaliza el caso de uso.
Postcondición	Se elaboró el listado de los lugares satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
	2.1.	El sistema, al realizar la búsqueda, no encuentra lugar registrado en la BD, muestra un mensaje “no existe lugar registrado en el sistema” y cierra el caso de uso.
	3.1.	El Administrador decide extender <<extend>> la operación sobre algún lugar específico que se ha mostrado en el listado a través del caso de uso: a.- Consultar lugar b.- Editar lugar. c.- Eliminar lugar.
Reglas del negocio	Debe existir al menos un lugar registrado para elaborar el listado. Una vez elaborado el listado de lugar, escoger entre tantos lugares en el listado para visualizar en detalle.	
Inclusiones	Ninguna.	
Extensiones	Visualizar Detalle de lugar. Editar lugares. Eliminar lugars.	

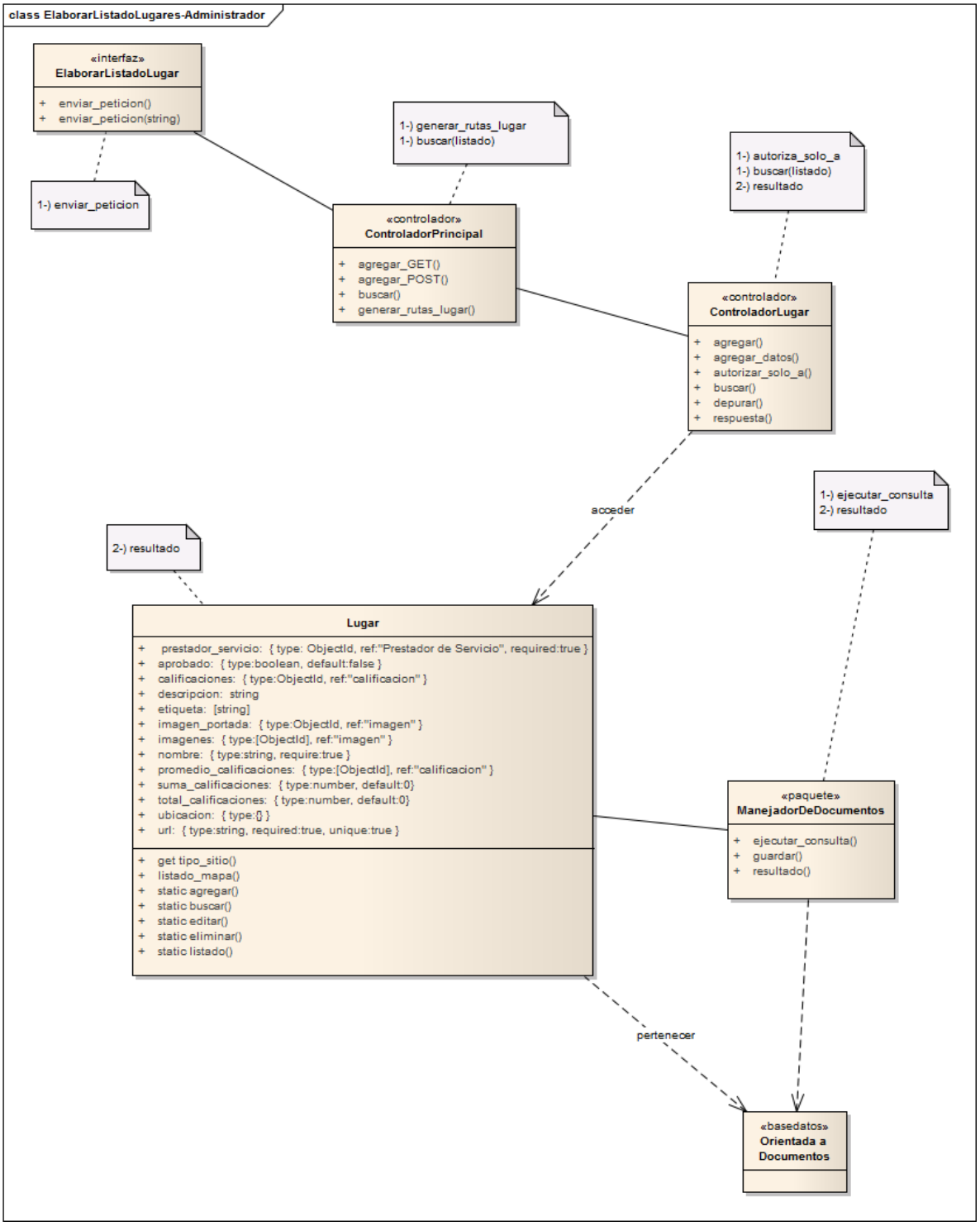


Figura D. 2. Diagrama de Clases: Elaborar Listado de Lugar por usuario Administrador

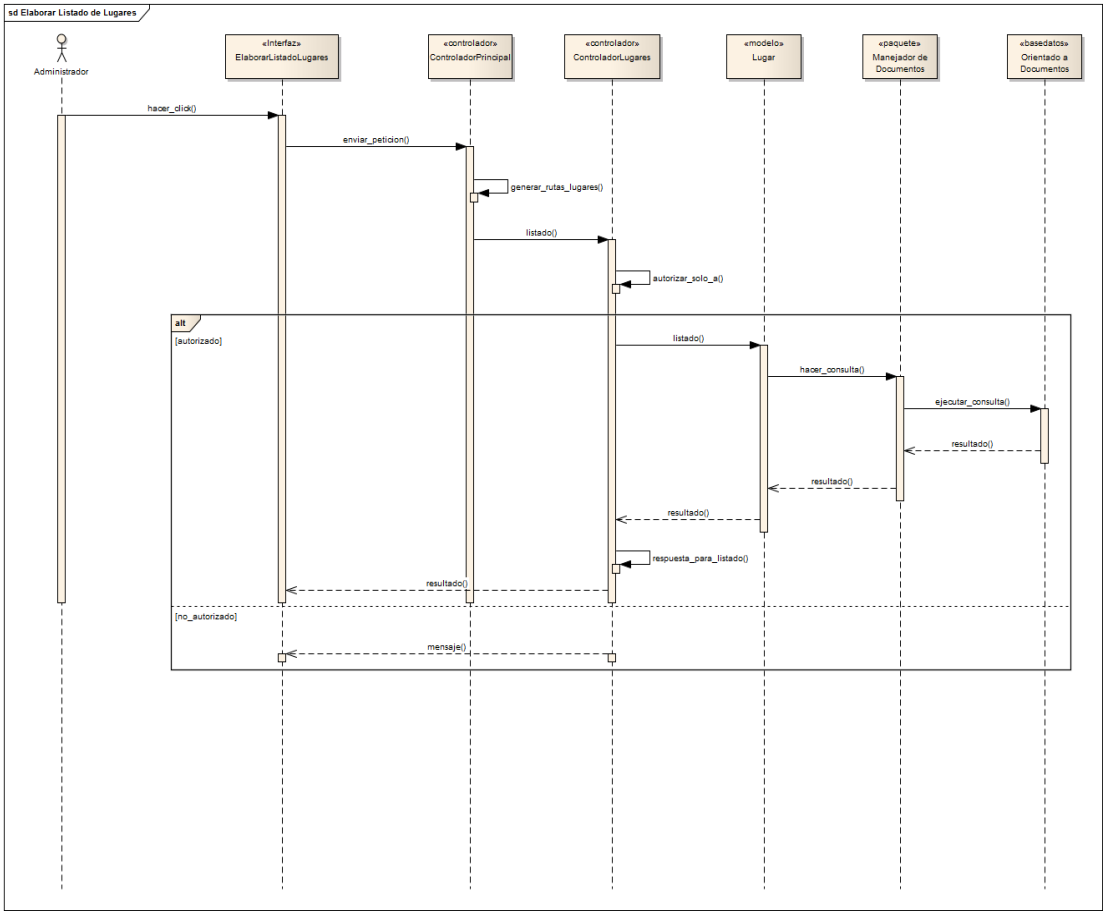


Figura D. 3. Diagrama de secuencia: Elaborar Listado de Lugar por usuario Administrador.

Figura D. 4 Colección de lugares: Listado de lugar por usuario Administrador.

A continuación, se muestra el modelo de plantilla para el proceso de listar lugar.

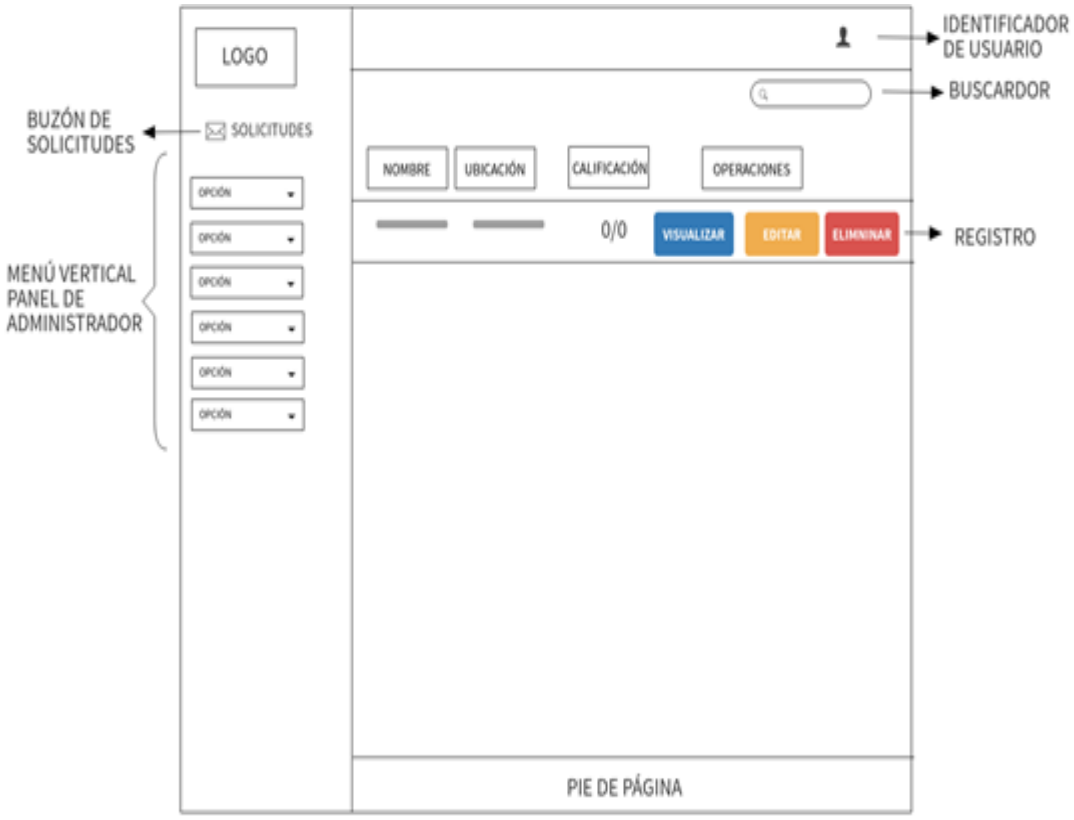


Figura D. 5. Plantilla de interfaz: Listar Lugar por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Lista de Lugar y el sistema responde mostrando la interfaz correspondiente.

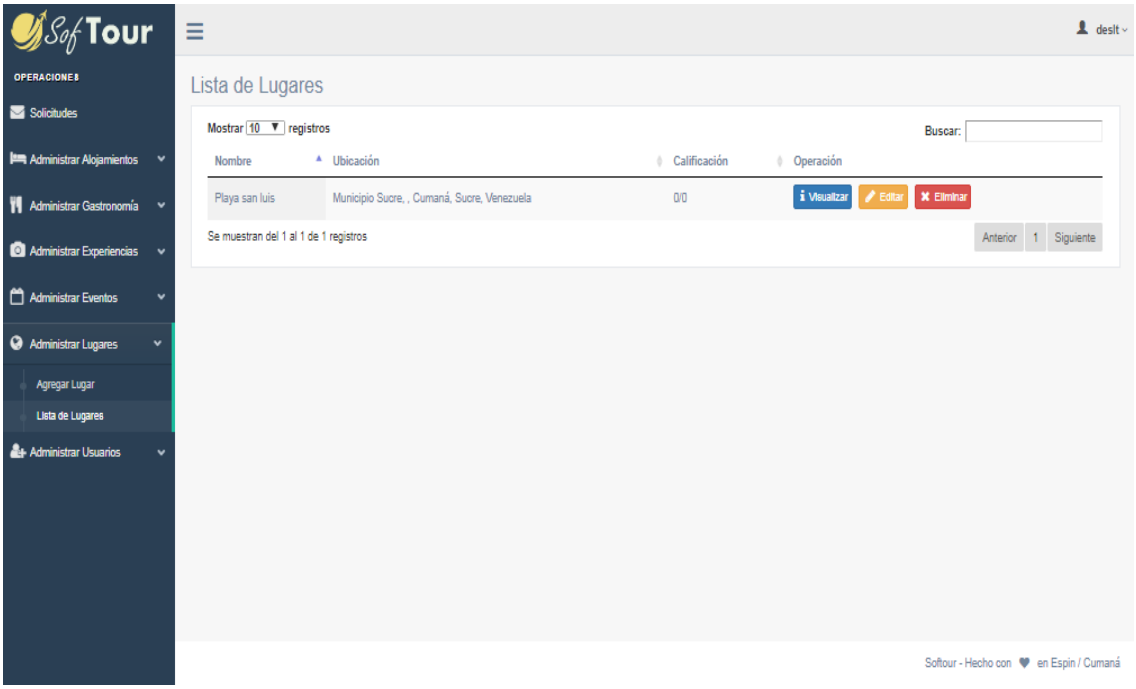


Figura D. 6. Interfaz Listado de Lugar por usuario Administrador.

Código desarrollado para la funcionalidad Listado de Lugar por usuario administrador

```
listado(req, res, next) {
    let query = {};

    if(req.session.passport.user.tipo_usr == "PrestadorServicio" && req.params.tipo_entidad != "lugar")
        query.prestador_servicio = req.session.passport.user.id;

    if(req.params.tipo_entidad == "lugar")
        query.__t = null;
    else{
        query.__t = req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1);
        query.aprobado = true;
    }

    this._Lugar.listado({"consulta": query}, (error, lista) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_listado(req, res);
        }

        res.locals.datos = {"value": lista};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_listado(req, res);
    })
}
```

Figura D. 7. Código desarrollado para el proceso Listar Lugar por usuario Administrador.

Apéndice E. Ciclo de Versión Administrador: Iteración Consultar Lugar.

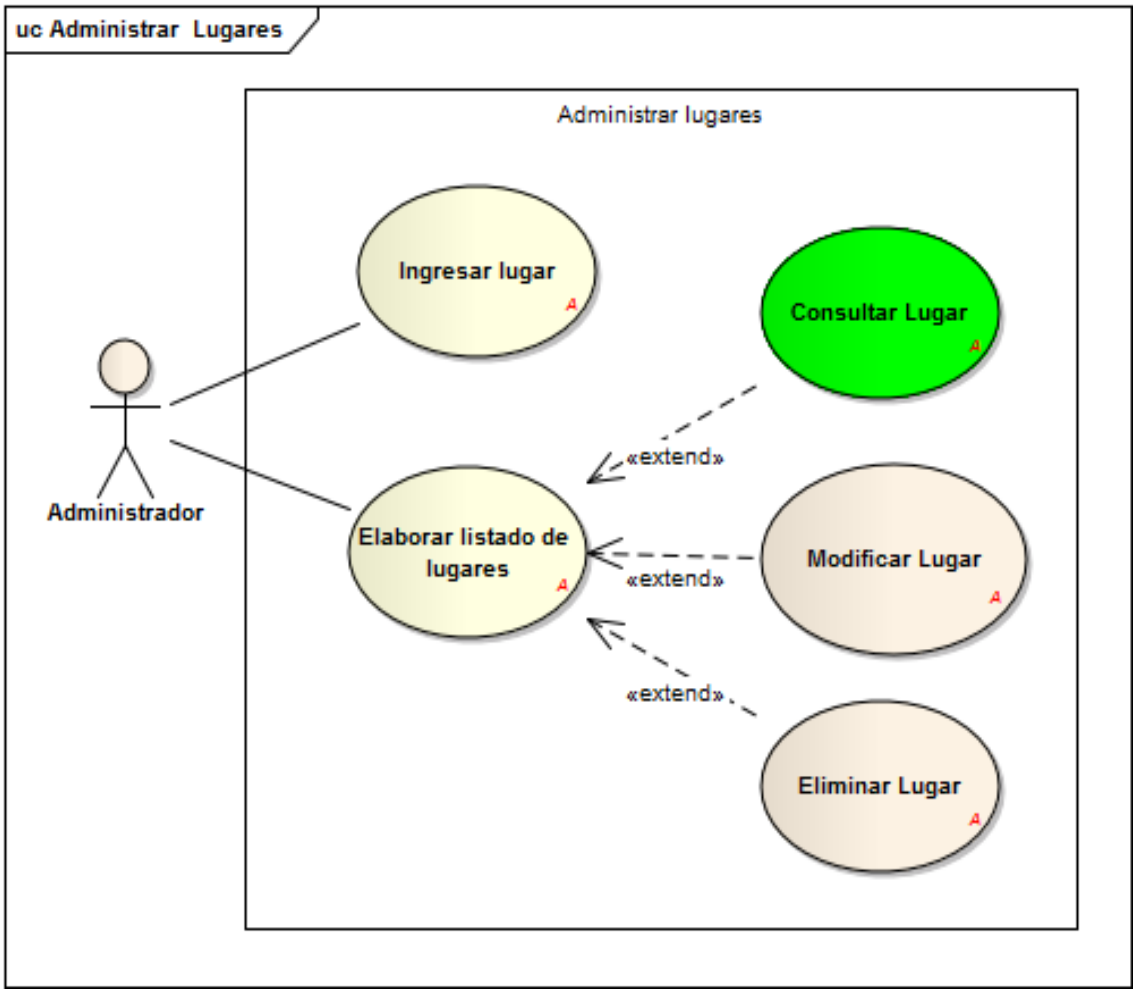


Figura E. 1. Caso de uso: Consultar Lugar por Administrador.

Tabla E. 1. Descripción Caso de uso Consultar Lugar por usuario Administrador.

Nombre del Caso de uso	Consultar lugar	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en consultar los atributos del lugar de manera satisfactoria.	
Precondición	Para consultar un lugar, éste debe estar registrado en el sistema.	
Secuencia **Normal de Eventos**	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Consultar”.
	2	El sistema ejecuta la búsqueda del lugar.
	3	El Administrador visualiza la consulta de los atributos del lugar.
	4	El sistema finaliza el caso de uso.
Postcondición	Se consultan los atributos del lugar de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no consultar el lugar y selecciona cancelar el caso de uso.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	El lugar debe estar registrado para que pueda ser consultado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

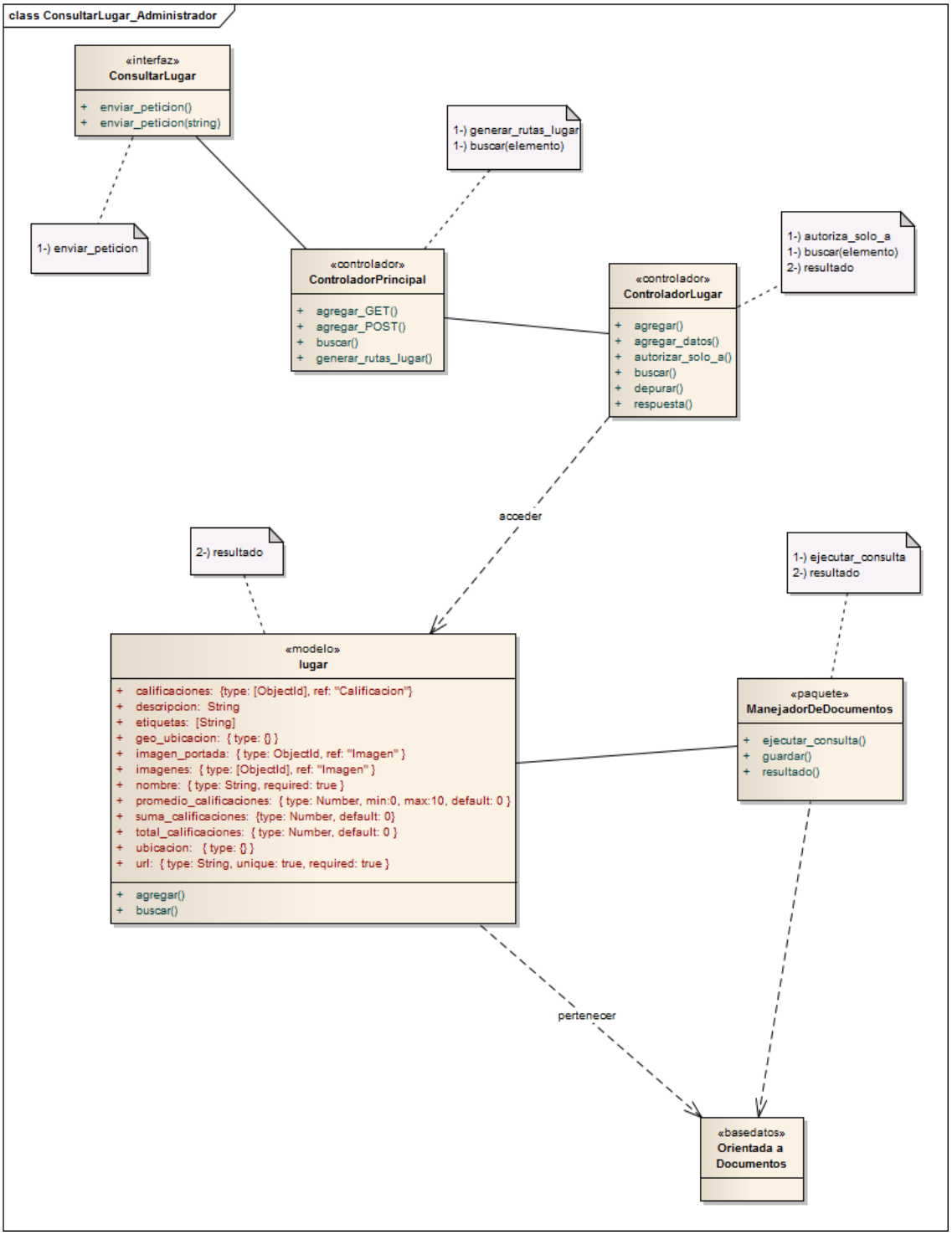


Figura E. 2.Diagrama de Clases: Consultar Lugar por usuario Administrador.

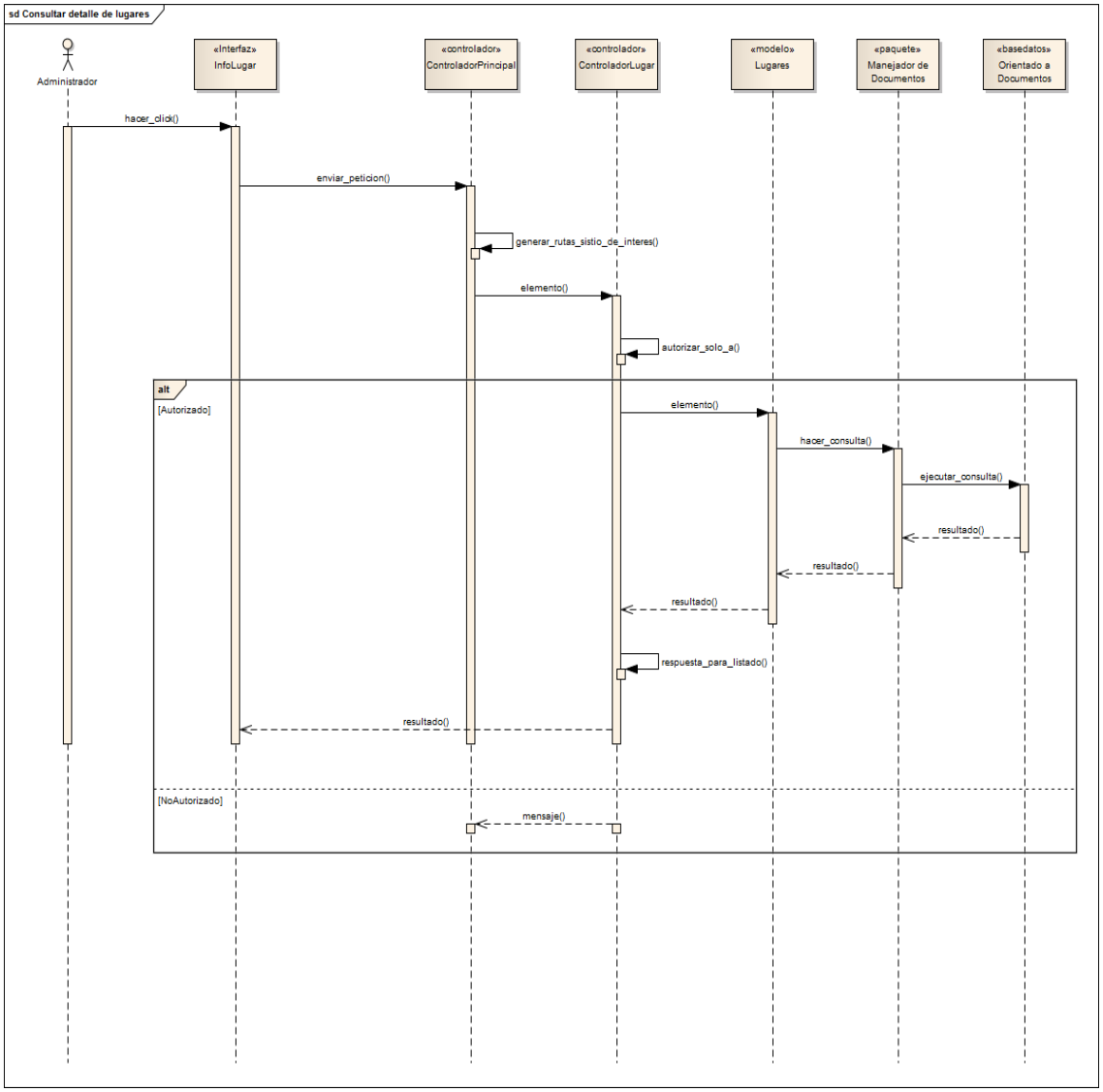


Figura E. 3.Diagrama de secuencia: Consultar Lugar por usuario Administrador.

A continuación, se muestra el modelo de plantilla para el proceso de consultar lugar.

Figura E. 4. Plantilla de interfaz: Consultar Lugar por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Consultar Lugar y el sistema responde mostrando la interfaz con las especificaciones

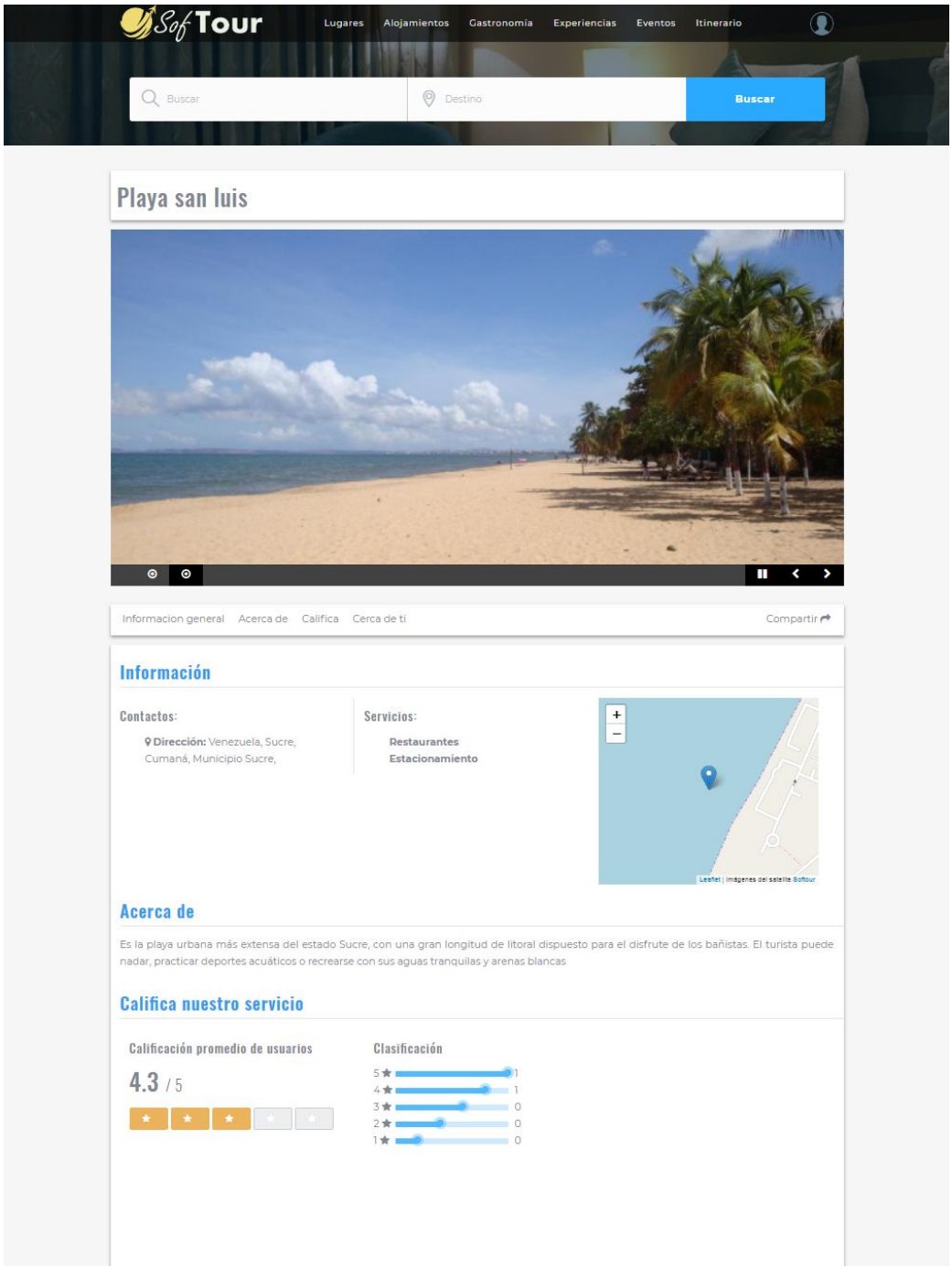


Figura E. 5. Interfaz de proceso de Consultar Lugar por usuario Administrador.

Código desarrollado para la funcionalidad Consultar Lugar por usuario administrador

```
consultar(req, res, next) {
    let parametros = {
        "__t": req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1),
        "url": req.params.url.toLowerCase()
    };
    if(parametros.__t)
        parametros.aprobado = true;

    this._Lugar.buscar_consulta(parametros, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        if(buscado.__t){
            this.CategoriaAlojamiento.find({"alojamiento": buscado._id}, (error, lista_categorias) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    if (req.isAuthenticated())
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_consultar_producto(req, res);
                }

                res.locals.datos = {
                    "value": buscado,
                    "lista_categorias": lista_categorias
                };

                if (req.isAuthenticated())
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                this._respuesta.para_consultar(req, res);
            });
        }
        else {
            res.locals.datos = { "value": buscado };

            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
    })
}
```

Figura E. 6. Código desarrollado para el proceso Consultar Lugar por usuario Administrador.

Prueba consultar lugares

Luego de validar el caso de uso Ingresar lugar, se procedió a verificar los casos de uso que depende de la creación de lugares, la prueba a realizar a el presente caso de uso se realiza consultando si cumplen o no su función sin el ingreso de datos, es decir, no se ingresaron datos ya que no lo requiere.

La prueba se realizó como se explica a continuación:

En primera instancia se toma en cuenta los valores agregados en la prueba anterior, ya que el ingreso de los datos generó una lista de lugares y a continuación se procedió a visualizar los lugares y consultar que la información

agregada se visualizada tenga congruencia con las que fue ingresada.

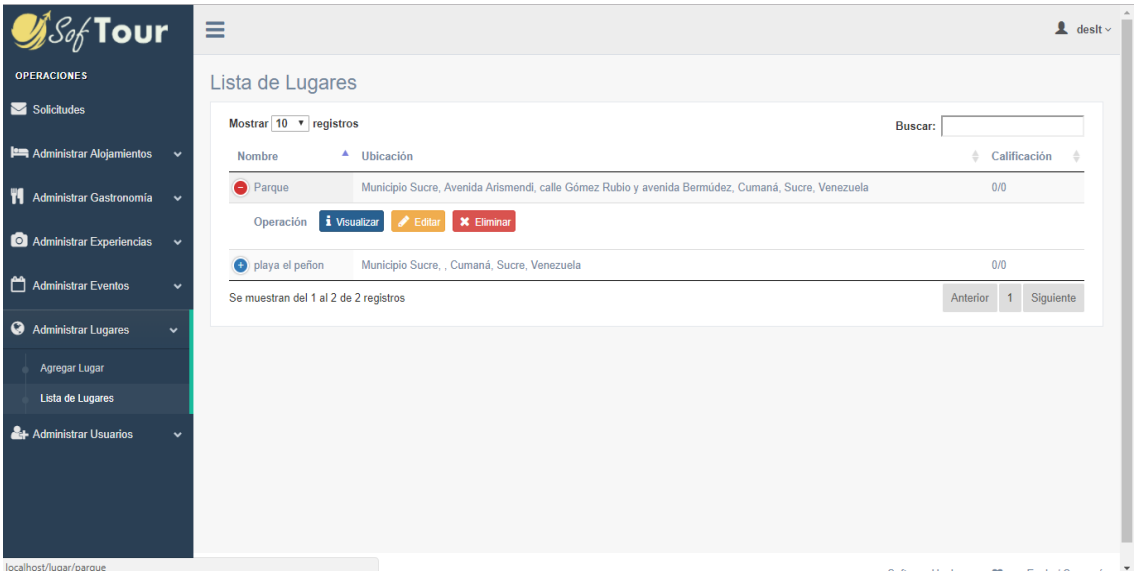


Figura E. 7. Prueba consultar lugar por Administrador.

Figura E. 8.Prueba Consultar Lugar por Administrador.

Conclusión de prueba

La función se cumple con total normalidad.

Prueba de interfaz y usabilidad

Manteniendo la técnica de análisis de usabilidad para la interfaz se procedió a verificar la interfaz que permite consultar la información de un lugar.

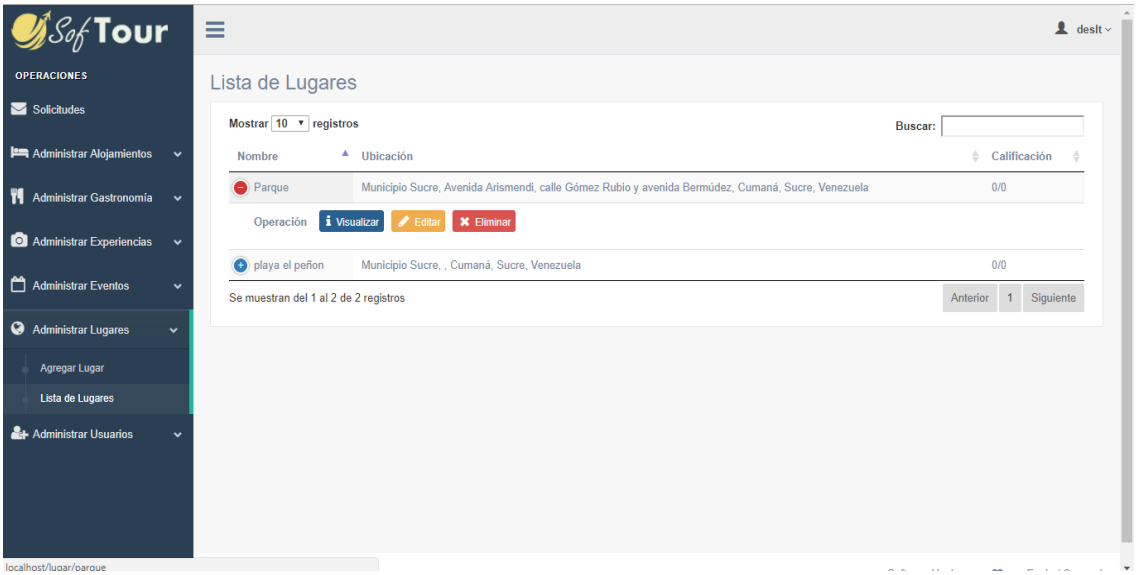


Figura E. 9.Interfaz Consultar Lugar por Administrador.

En este caso el diseño de interfaz mantiene consistencia con otras vistas del sistema previamente evaluadas, además del uso de metáforas para el reconocimiento de funciones y además se usaron colores distintos para separar las otras funciones que se encuentran con la función visualizar lugar.

Apéndice F. Ciclo de Versión Administrador: Iteración Modificar Lugar.

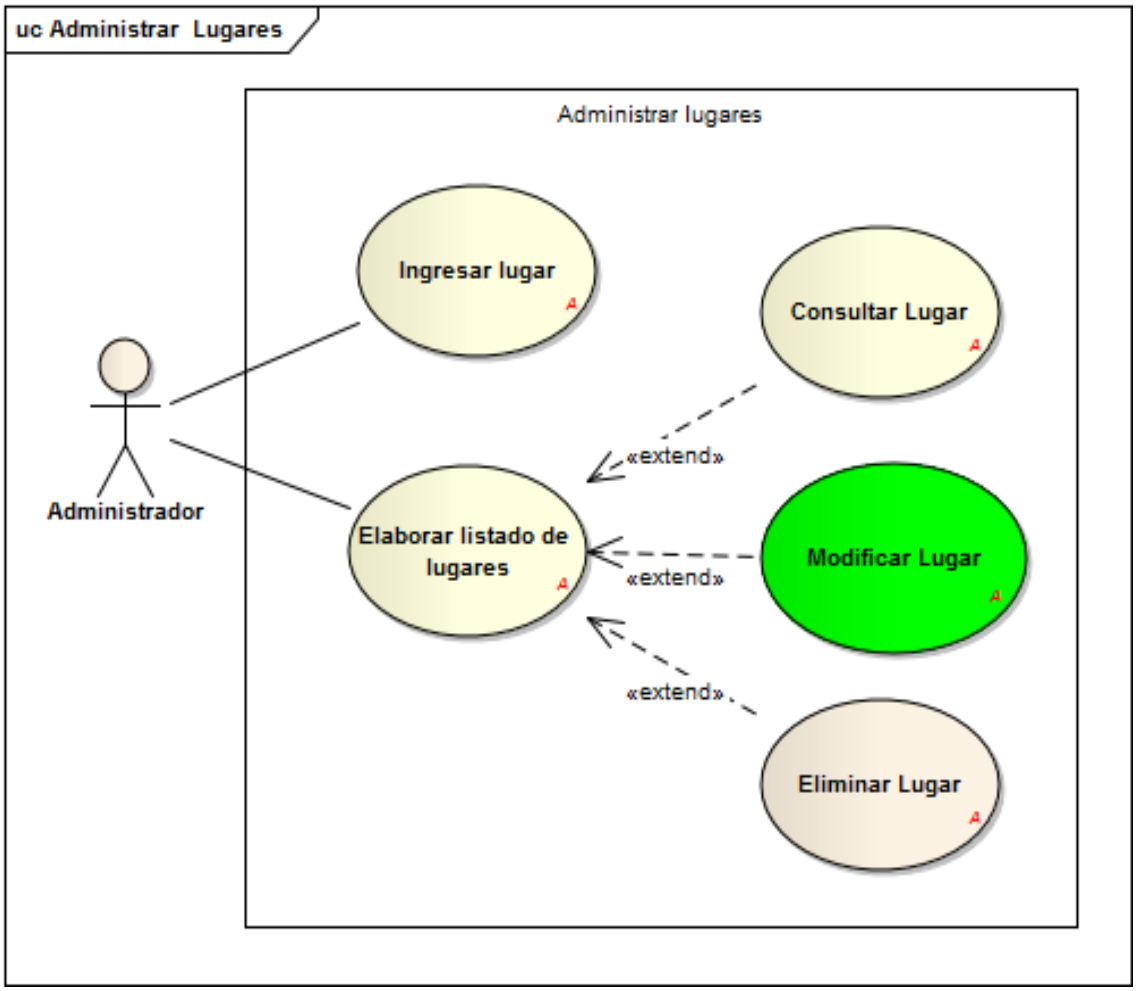


Figura F. 1. Caso de uso: Modificar Lugar por Administrador.

Tabla F. 1. Descripción Caso de uso Modificar Lugar por usuario Administrador.

Nombre del Caso de uso	Modificar lugar	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en modificar los atributos del lugar de manera satisfactoria.	
Precondición	Para modificar un lugar debe estar registrado en el sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Modificar”.
	2	El sistema, muestra el formulario correspondiente para indicar el lugar a Modificar.
	3	El Administrador ingresa el patrón de búsqueda del lugar a modificar.
	4	El sistema ejecuta la búsqueda del lugar, mostrando un formulario con los atributos respectivos.
	5	El Administrador modifica los atributos del lugar, selecciona aceptar.
	6	El sistema realiza las modificaciones en la BD y finaliza el caso de uso.
Postcondición	Se modifican los atributos del lugar de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no modificar el lugar y selecciona cancelar el caso de uso.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. El lugar debe estar registrado para que pueda ser modificado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

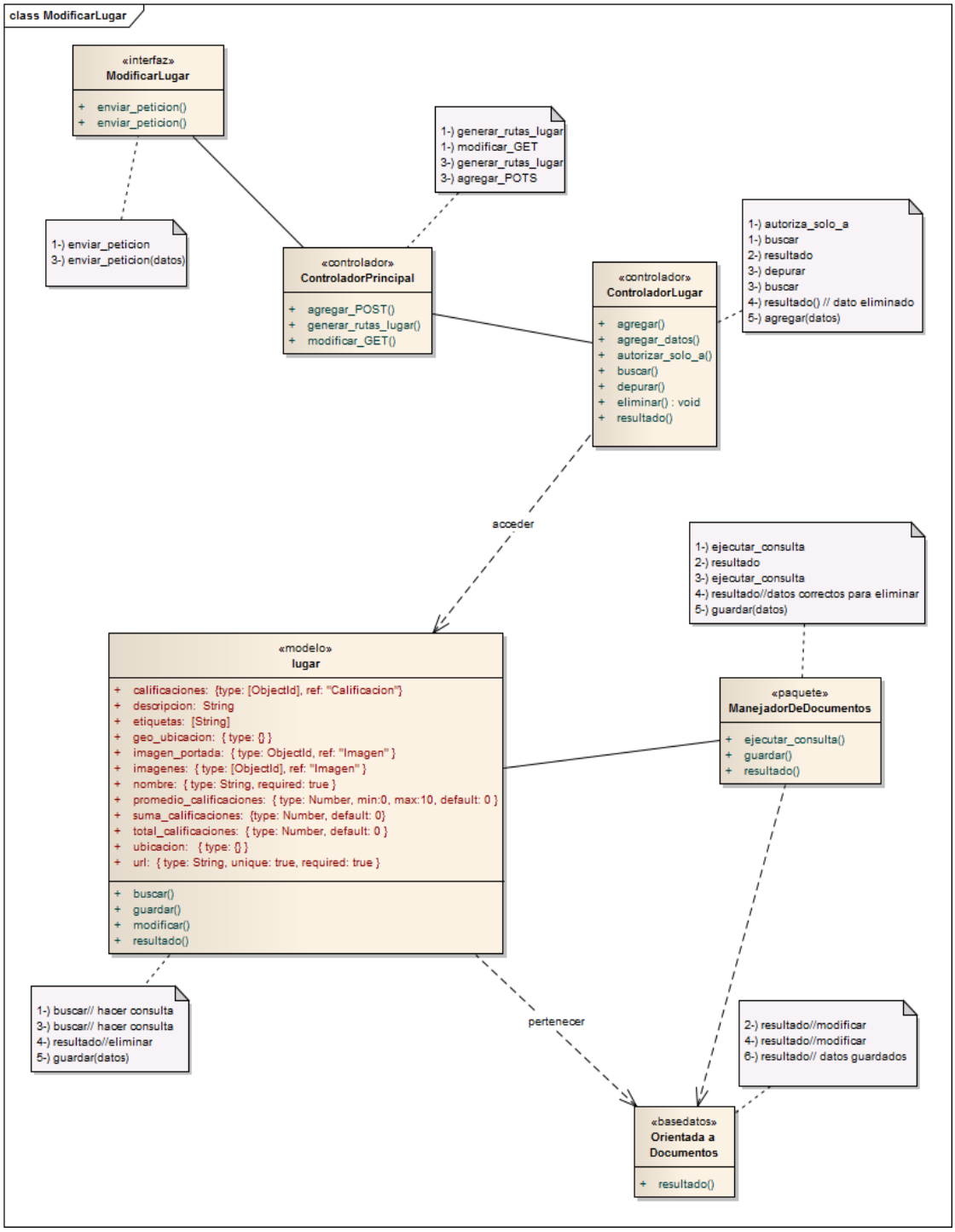


Figura F. 2. Diagrama de Clases: Modificar Lugar por usuario Administrador.

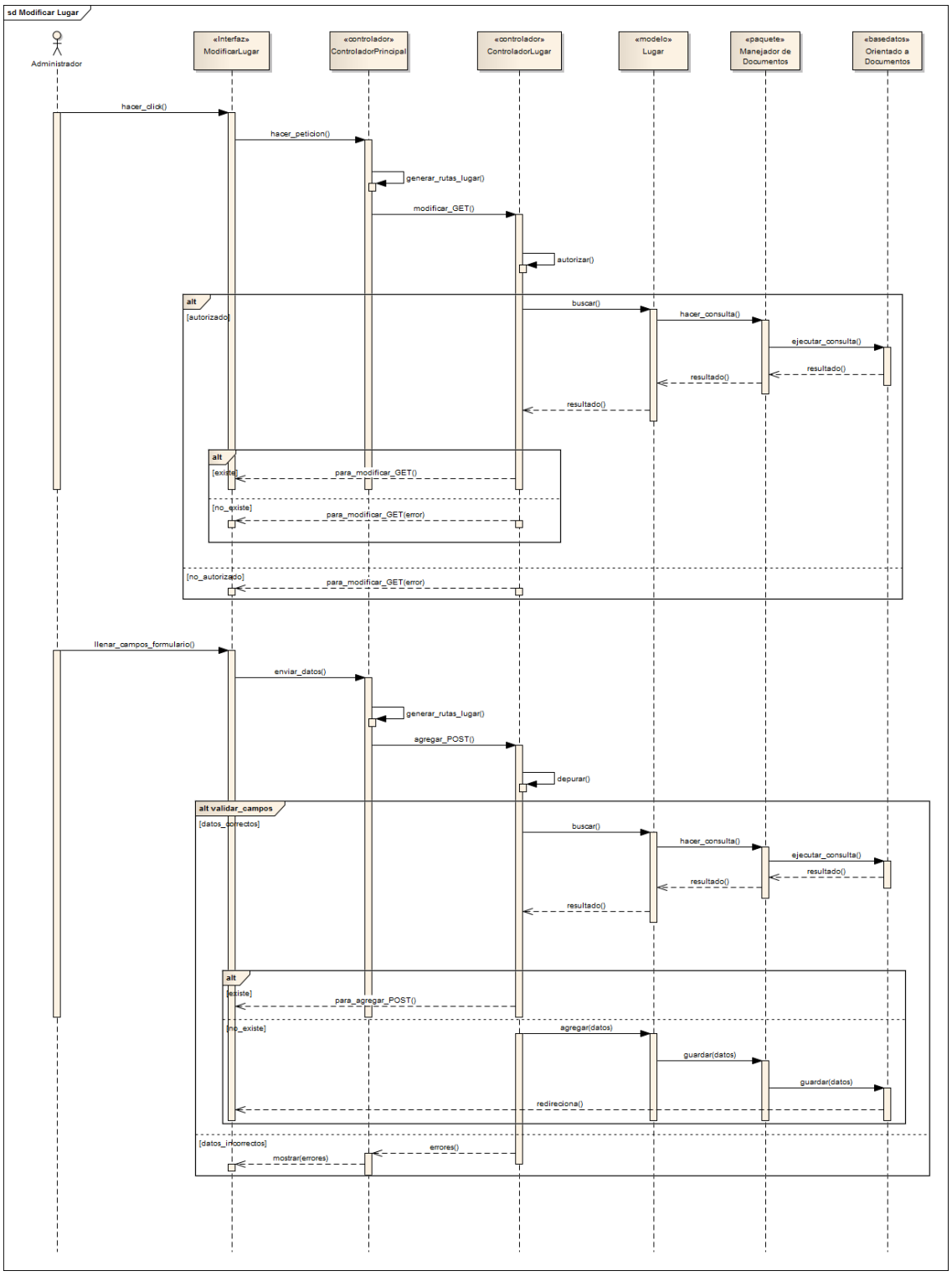


Figura F. 3.Diagrama de secuencia: Modificar Lugar por usuario Administrador.

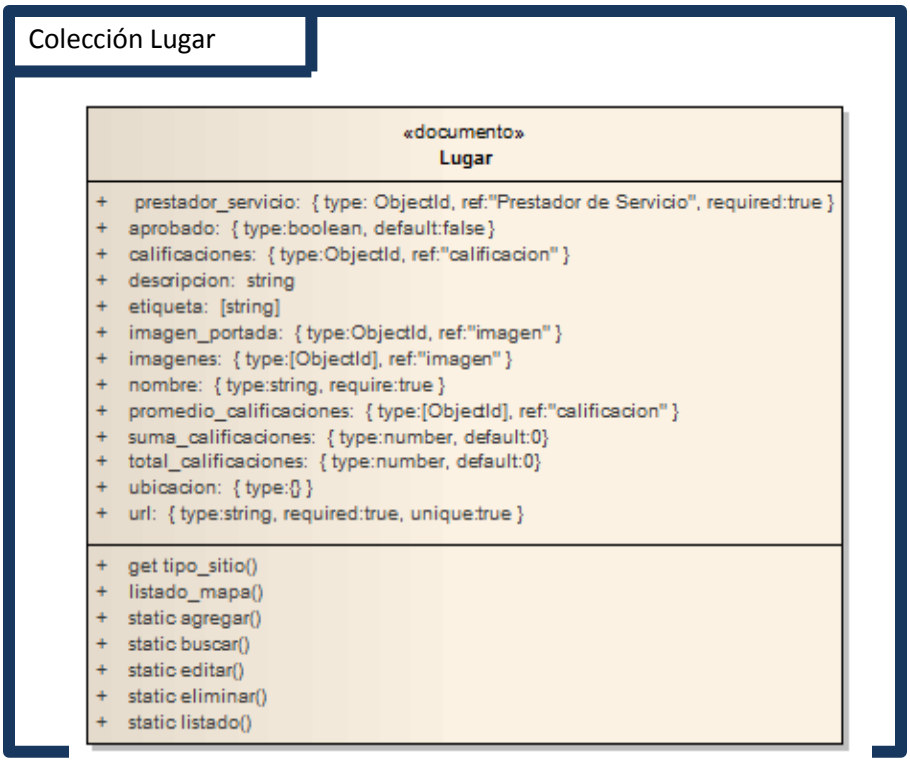


Figura F. 4. Colección de lugares: Modificar Lugar por usuario Administrador.

A continuación, se muestran los modelos de plantillas para el proceso de modificar lugar.

Figura F. 5. Plantilla de interfaz 1: Modificar Lugar por usuario Administrador.

Figura F. 6. Plantilla de interfaz 2: Modificar Lugar por usuario Administrador.

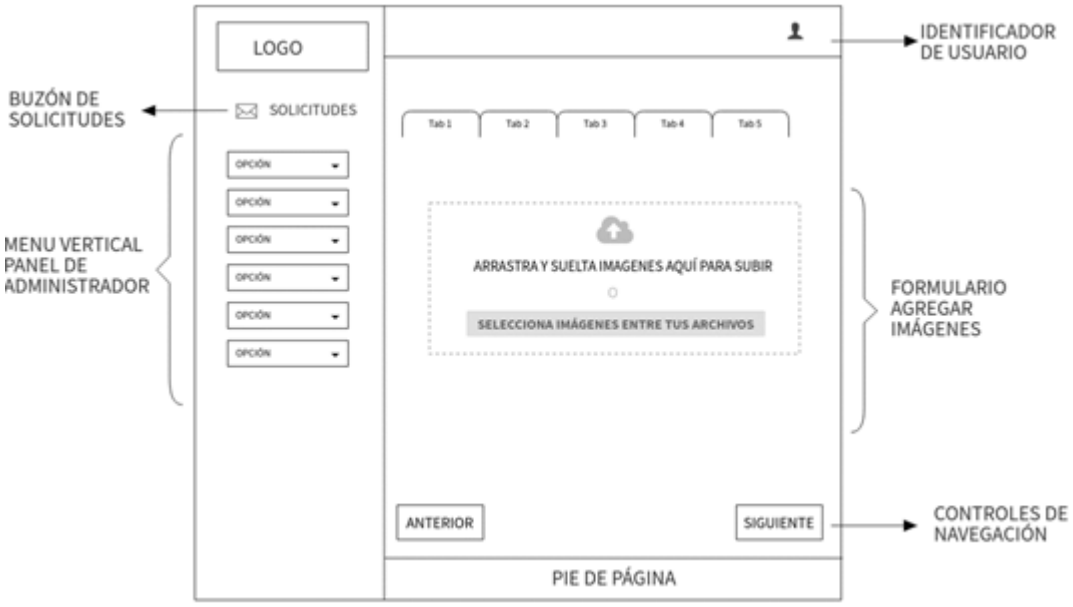


Figura F. 7. Plantilla de interfaz 3: Modificar Lugar por usuario Administrador.

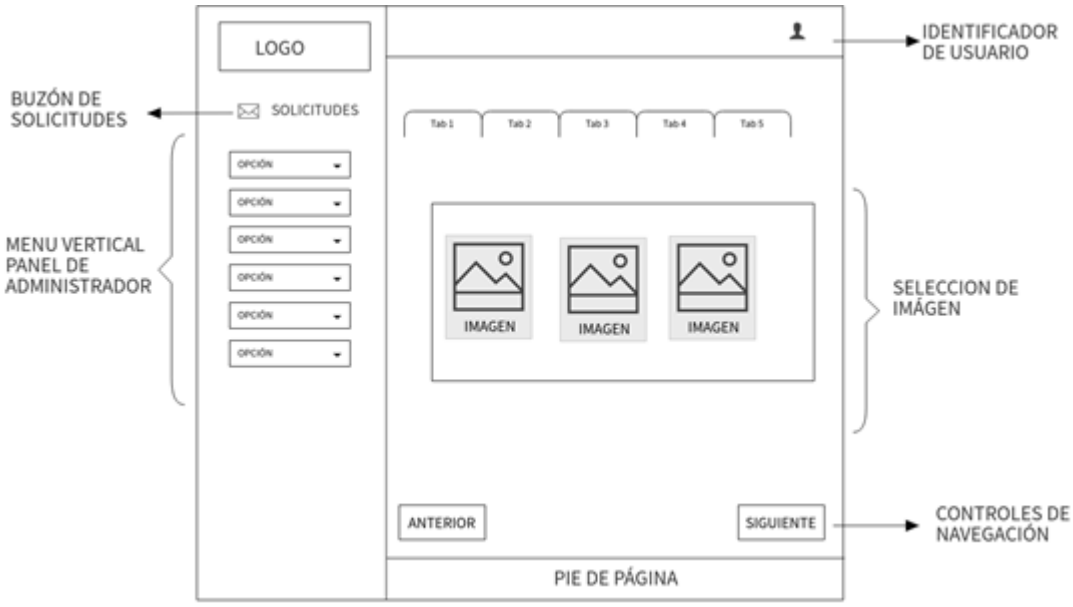


Figura F. 8. Plantilla de interfaz 4: Modificar Lugar por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Modificar Evento y el sistema responde mostrando el formulario correspondiente.

Figura F. 9. Interfaz 1 de proceso de Modificar Lugar por usuario Administrador.

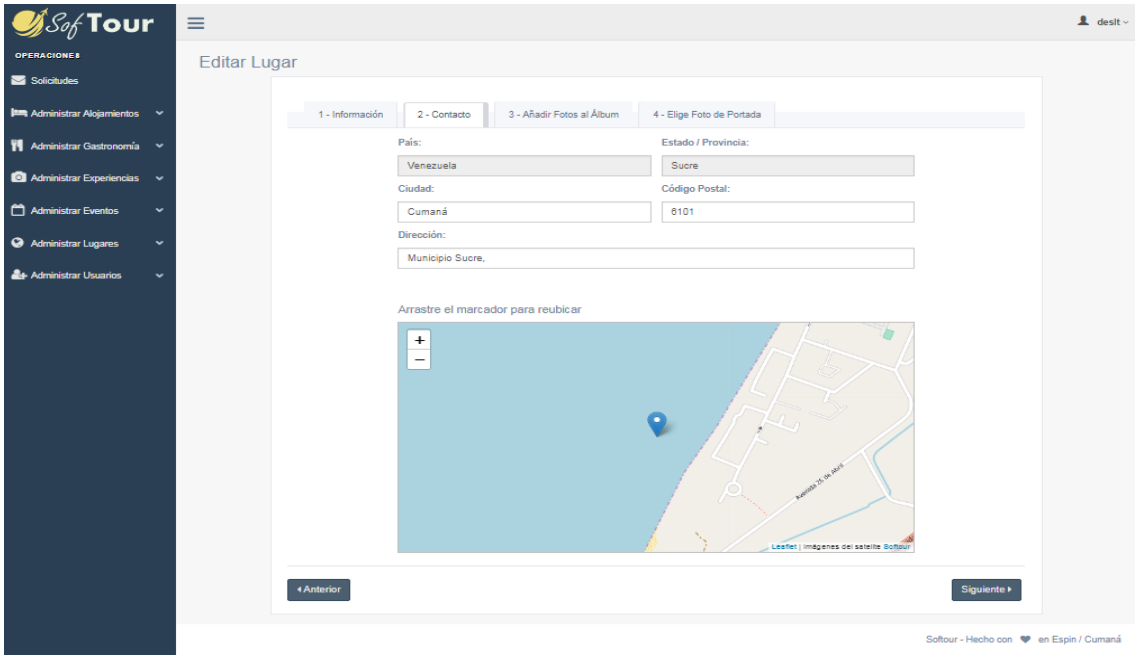


Figura F. 10. Interfaz 2 de proceso de Modificar Lugar por usuario Administrador.

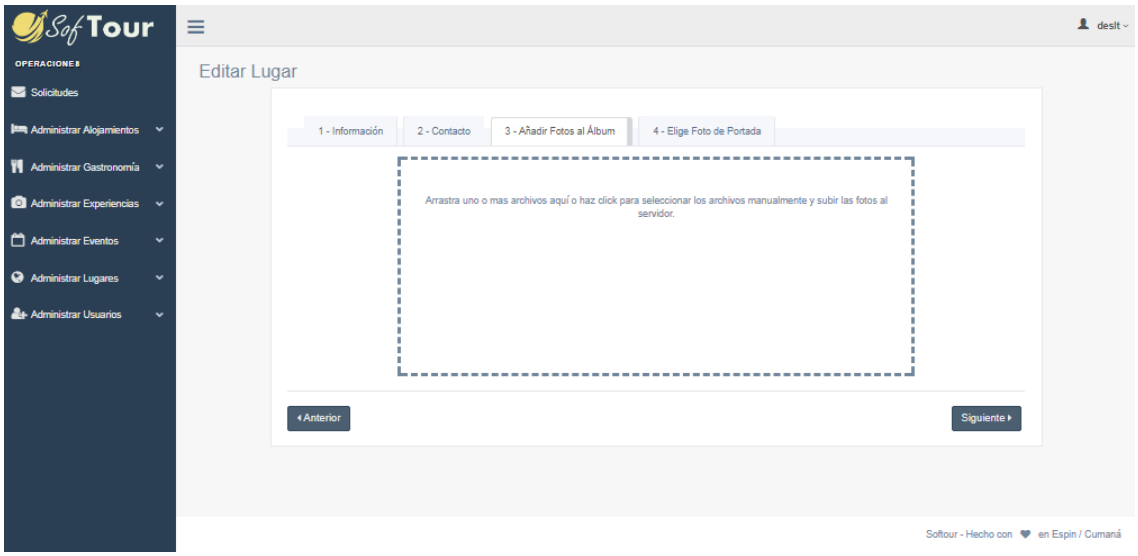


Figura F. 11. Interfaz 3 de proceso de Modificar Lugar por usuario Administrador.

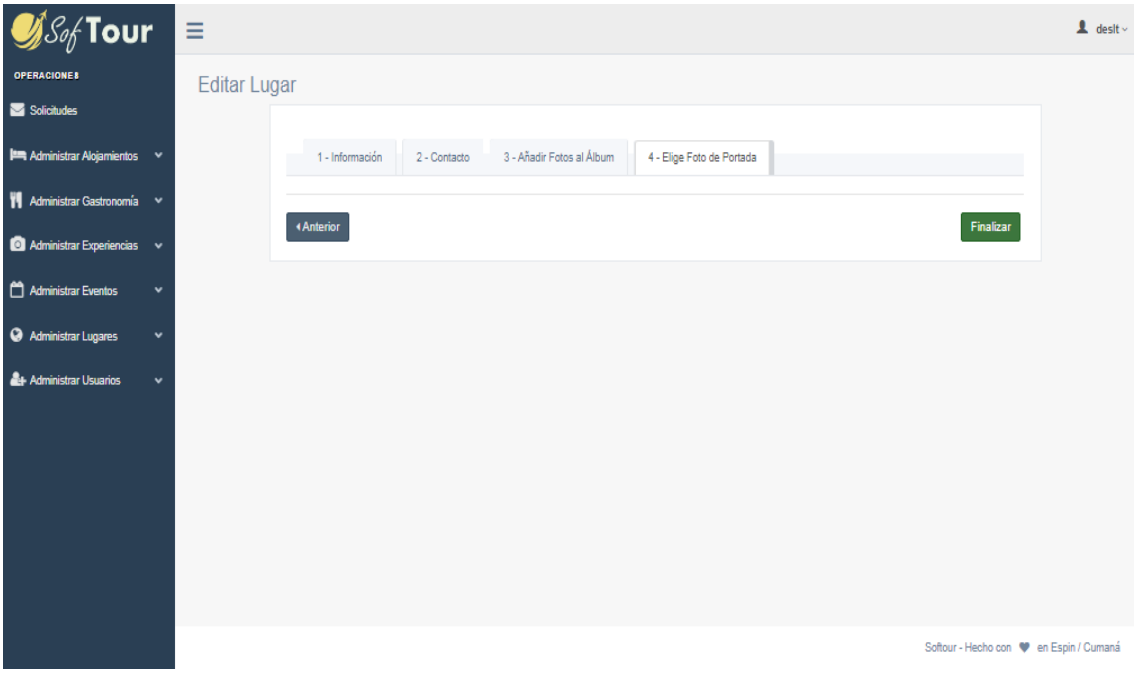


Figura F. 12. Interfaz 4 de proceso de Modificar Lugar por usuario Administrador

Código desarrollado para la funcionalidad Modificar Alojamiento por usuario Administrador

```
modificar(req, res, next) {
    this._validar.campos(req.body, true, (error, resultados) => {
        if(error) return next(error);
        res.locals.datos = resultados;

        if(res.locals.datos.error){
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_modificar(req, res);
        }

        else {
            let tipo_entidad = req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.
                tipo_entidad.substring(1);

            this._Lugar.buscar({"url": req.params.url.toLowerCase(), "__t": tipo_entidad}, (error, buscado) => {
                if(error){
                    res.locals.datos = {"error": error};
                    res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
                    this._respuesta.para_modificar(req, res);
                }

                if(!buscado) return next(new this._error_NoEncontrado());
                res.locals.datos.value.geo_ubicacion = {
                    "type": "Point",
                    "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
                };
                res.locals.datos.value.ubicacion = {
                    "direccion": res.locals.datos.value.direccion,
                    "pais": res.locals.datos.value.pais,
                    "estado": res.locals.datos.value.estado,
                    "ciudad": res.locals.datos.value.ciudad,
                    "codigo_postal": res.locals.datos.value.codigo_postal
                };

                res.locals.datos.value.url = req.params.url.toLowerCase();
                let lugar = this._fabricar.lugarTipo(res.locals.datos.value.tipo_lugar);
                lugar.modificar(res.locals.datos.value, (error, modificado) => {
                    if(error) {
                        res.locals.datos.error = error;
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_modificar(req, res);
                    }
                    if(!modificado) return next(new this._error_NoEncontrado());

                    res.locals.datos = {"value": modificado};
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_modificar(req, res);
                });
            });
        }
    });
}
```

Figura F. 13. Código desarrollado para el proceso Modificar Lugar por usuario Administrador.

Prueba Modificar Lugar

Para realizar prueba a esta función del sistema se tomó en cuenta que deben estar agregados lugares al sistema. El fin de esta función es modificar los datos de lugares ya agregados a sistema y guardar de formar satisfactoria el cambio.

La prueba se realizó de acuerdo a los siguientes parámetros:

Se procedió a modificar los datos de información y dirección, que anteriormente habían sido llenados en las pruebas realizadas a la función Agregar Lugar.

Nueva descripción a agregar: Debe su nombre a los valientes indígenas. Son trece hectáreas a las orillas del río Manzanares, desarrolladas para el disfrute y sano esparcimiento de la familia. Cuenta con una concha acústica, cabañas, parque infantil, caminerías, fuente de soda, muchos árboles, donde abunda el mango, apamates y caobos.

Nueva dirección a agregar: Municipio Sucre, Avenida Arismendi, calle Gómez Rubio y avenida Bermúdez.

A continuación, se muestra secuencia de imágenes mientras se realiza prueba de modificación.

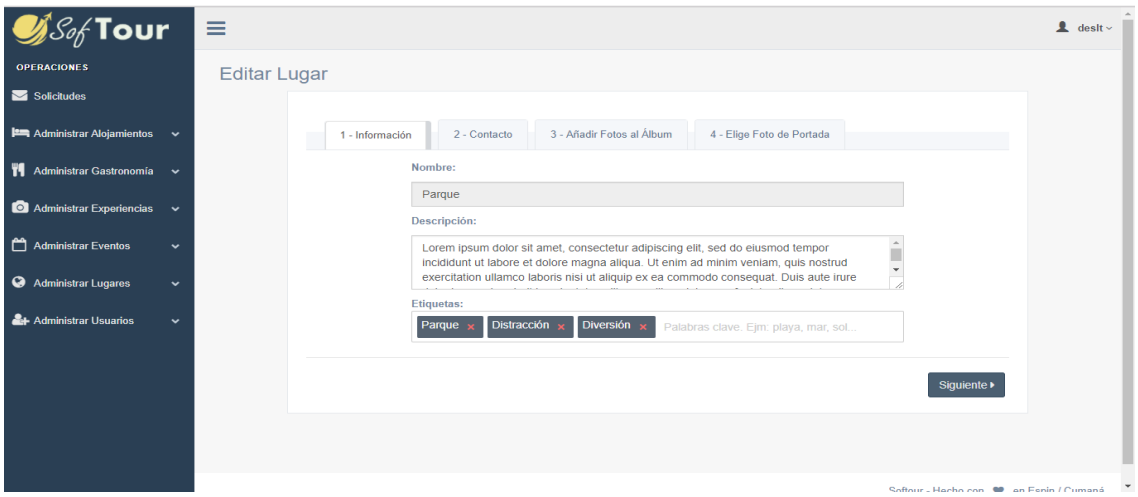


Figura F. 14.Prueba Modificar Lugar por Administrador.

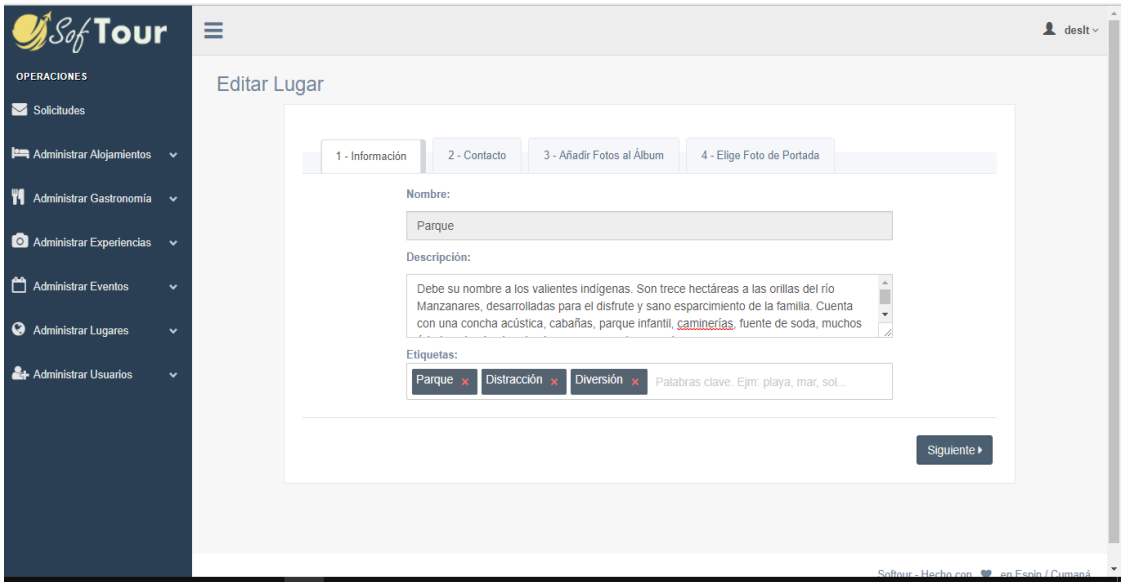


Figura F. 15.Prueba Modificar Lugar por Administrador.

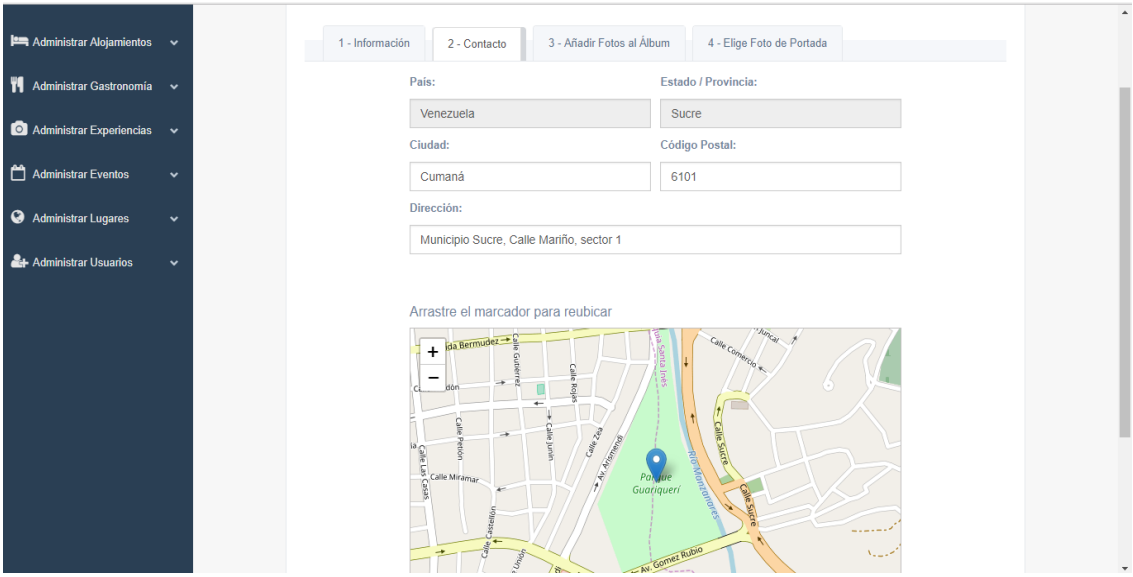


Figura F. 16.Prueba Modificar Lugar por Administrador.

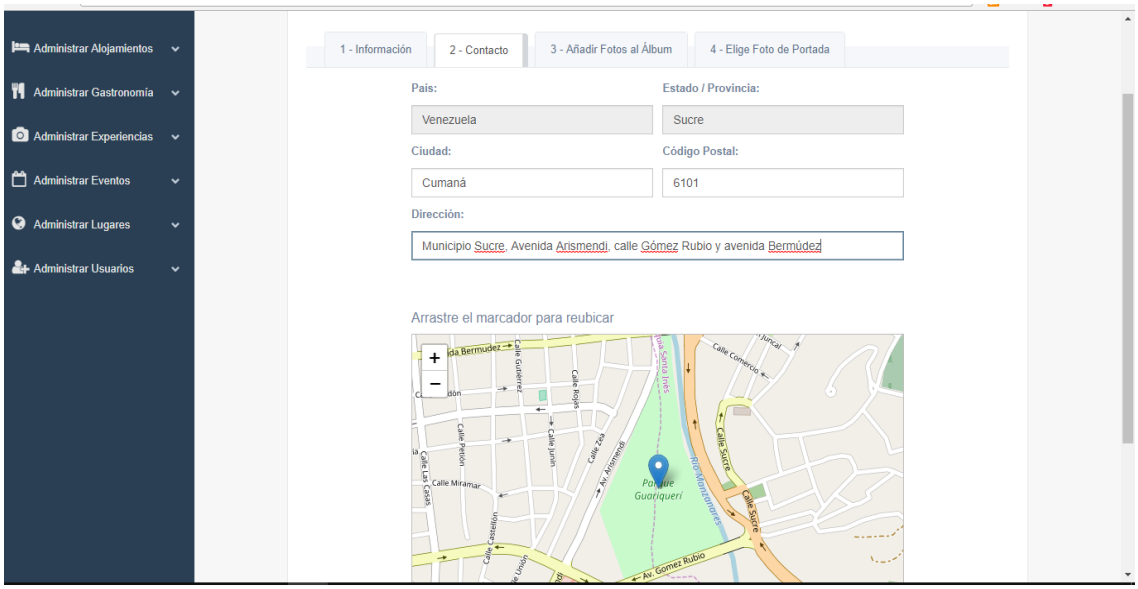


Figura F. 17. Prueba Modificar lugar por Administrador.

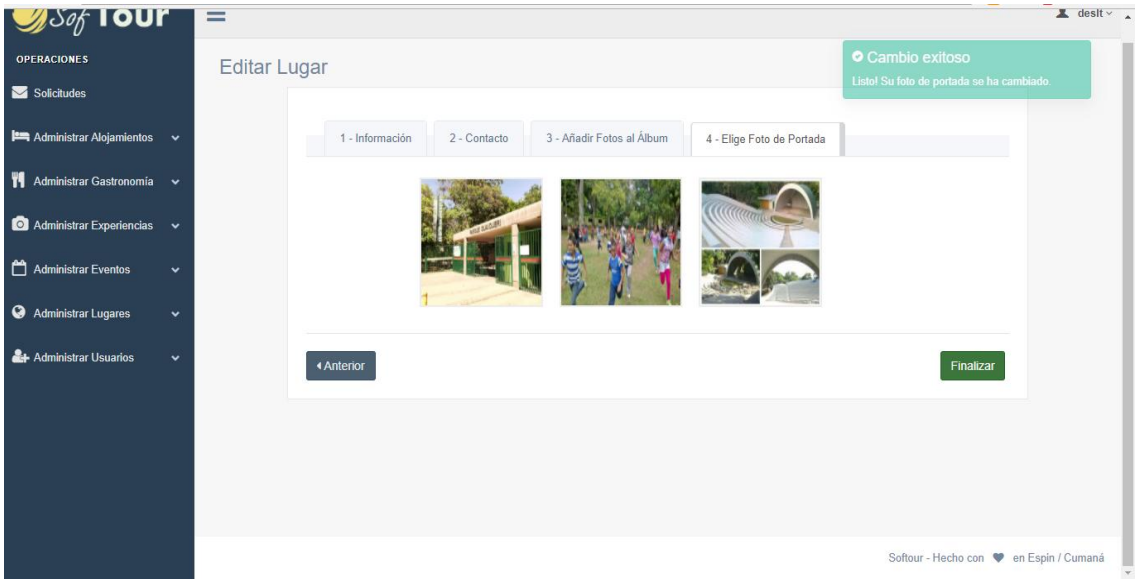


Figura F. 18.Prueba Modificar Lugar por Administrador.

Conclusión de la prueba

Se pudo validar que la función cumple con el fin para el cual fue creada.

Prueba de interfaz y usabilidad

Buscando mantener estándares de calidad tanto en funcionalidad como en interfaz el equipo de desarrollo presentó la siguiente interfaz para la función Modificar Lugar.

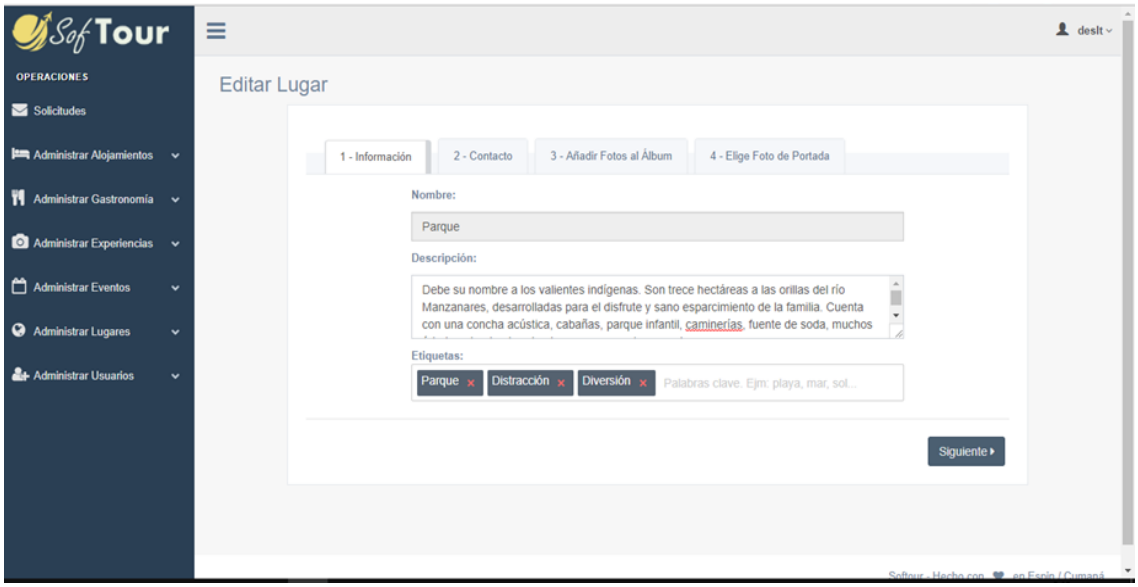


Figura F. 19.Modificar Lugar por Administrador.

Haciendo el análisis se puede denotar que cumple con la consistencia con la interfaz de Agregar Lugar, por ende, cumple con las mismas características.

Apéndice G. Ciclo de Versión Administrador: Iteración Eliminar Lugar.

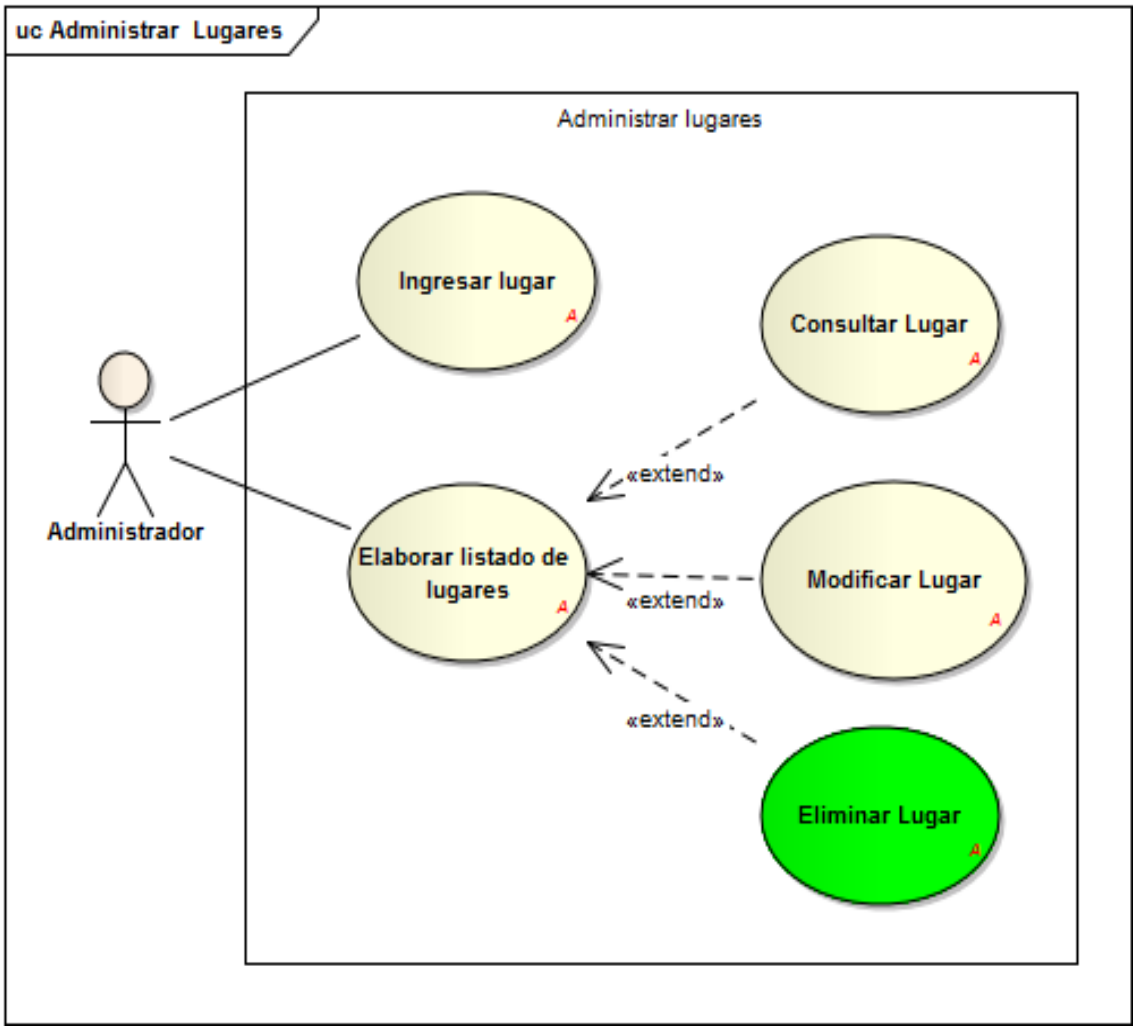


Figura G. 1. Caso de uso: Eliminar Lugar por Administrador.

Tabla G. 1. Descripción Caso de uso Eliminar Lugar por usuario Administrador.

Nombre del Caso de uso	Eliminar lugar	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en eliminar lugar de manera satisfactoria.	
Precondición	Para eliminar un lugar debe estar registrado en el sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Eliminar”.
	2	El sistema ejecuta la búsqueda del lugar.
	3	El Administrador visualiza el lugar que desea eliminar y selecciona aceptar.
	4	El sistema muestra un mensaje “¿Está seguro de eliminar la cuenta del lugar seleccionado?”
	5	El administrador selecciona la opción “Si, estoy seguro”
	6	El sistema realiza la eliminación del lugar en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se elimina el lugar de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	5.1.	El Administrador decide no continuar con el caso de uso y selecciona cancelar el caso de uso, seleccionando otra opción del sistema.
	5.2.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. El lugar debe estar registrado para que pueda ser eliminado.	
Inclusiones	Ninguna.	

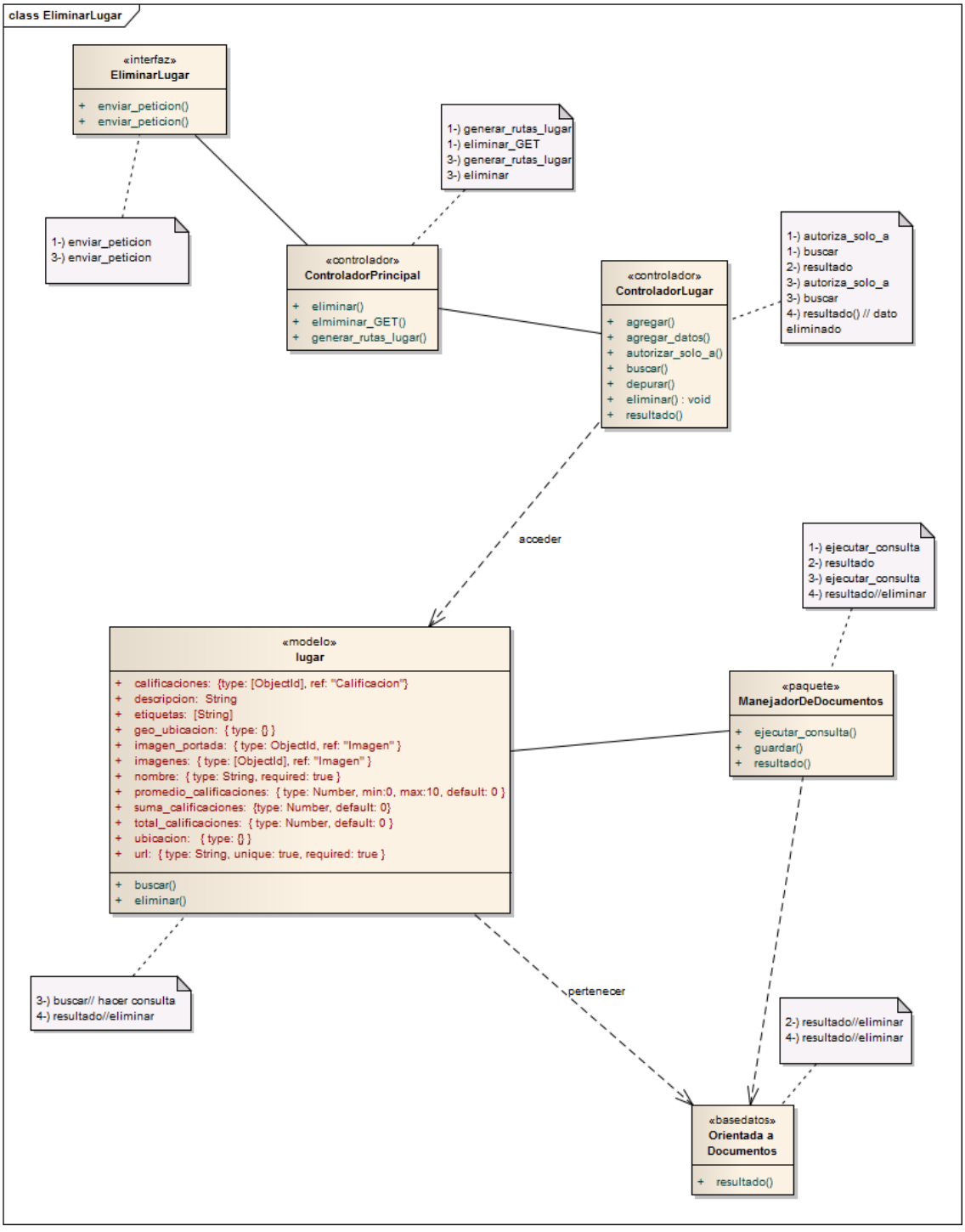


Figura G. 2. Diagrama de Clases: Eliminar Lugar por usuario Administrador.

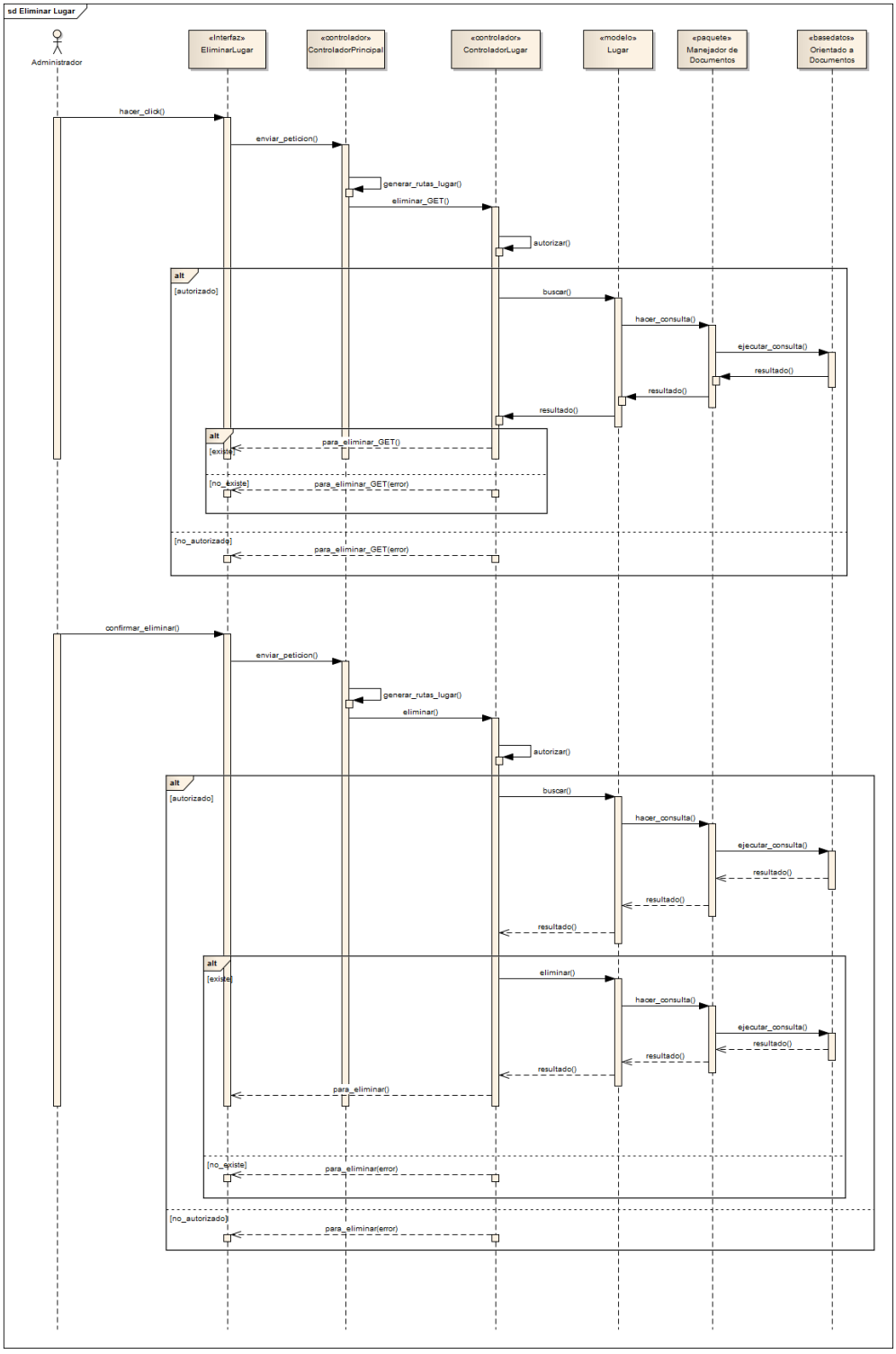


Figura G. 3. Diagrama de secuencia: Eliminar Lugar por usuario Administrador.

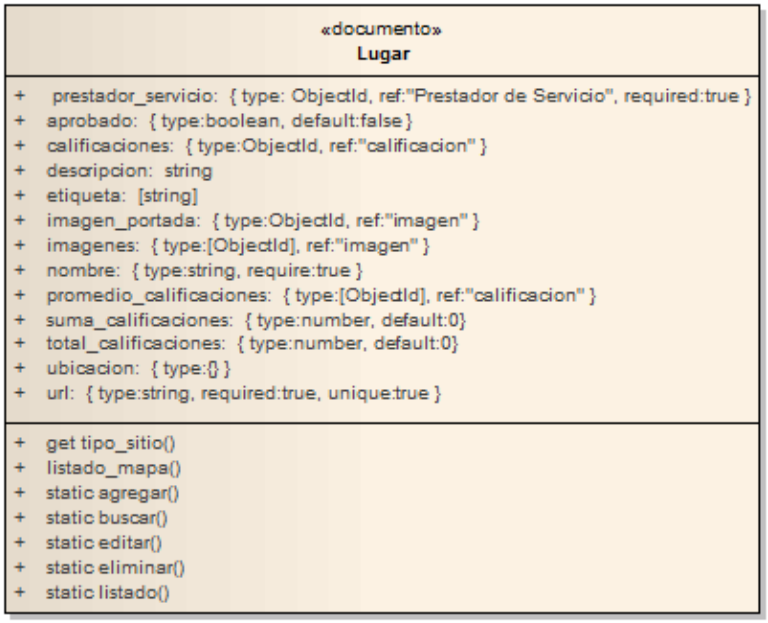


Figura G. 4. Colección de Lugares: Eliminar lugar por usuario Administrador

A continuación, se muestra el modelo de plantilla para el proceso de eliminar lugar.

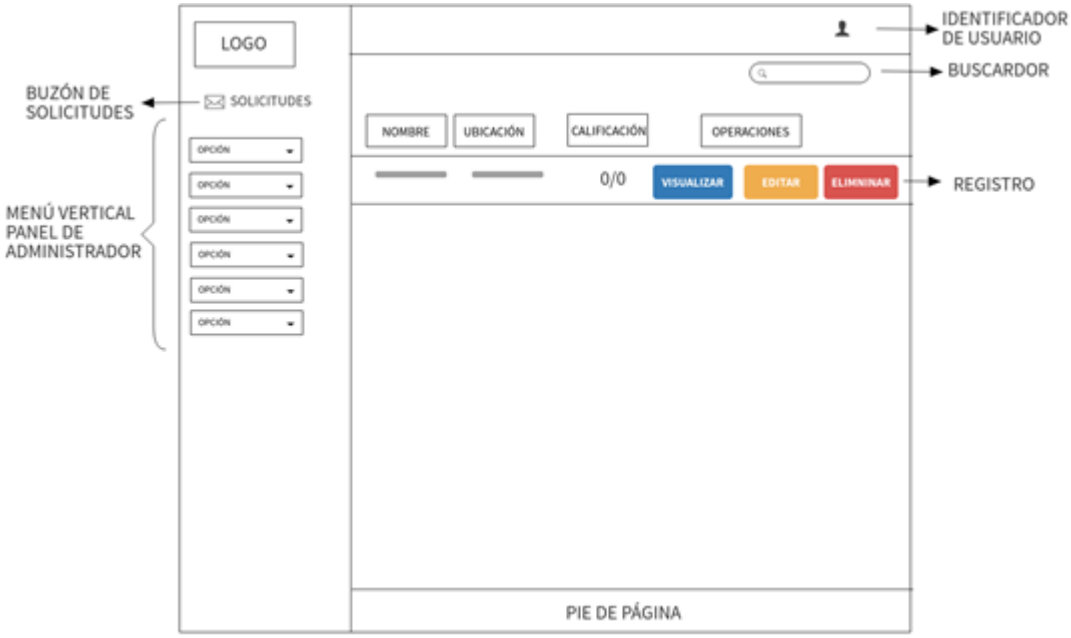


Figura G. 5. Plantilla de interfaz 1: Eliminar Lugar por usuario Administrador.

Figura G. 6. Plantilla de interfaz: Eliminar Lugar por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Eliminar Evento y el sistema responde mostrando la interfaz de confirmación de acción.

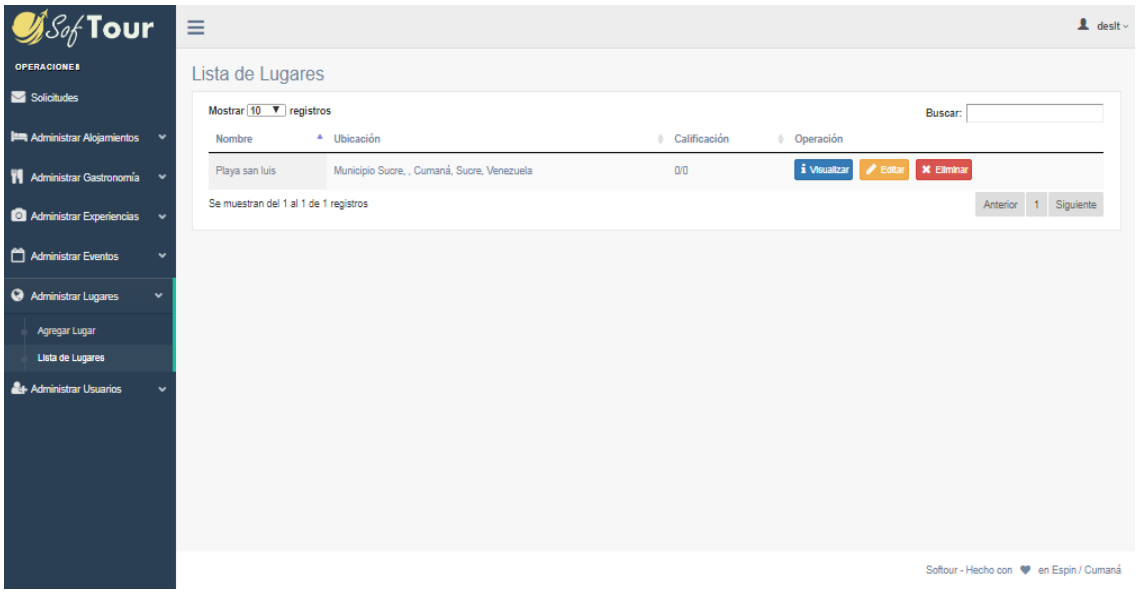


Figura G. 7. Interfaz 1 de proceso de Eliminar Lugar por usuario Administrador.

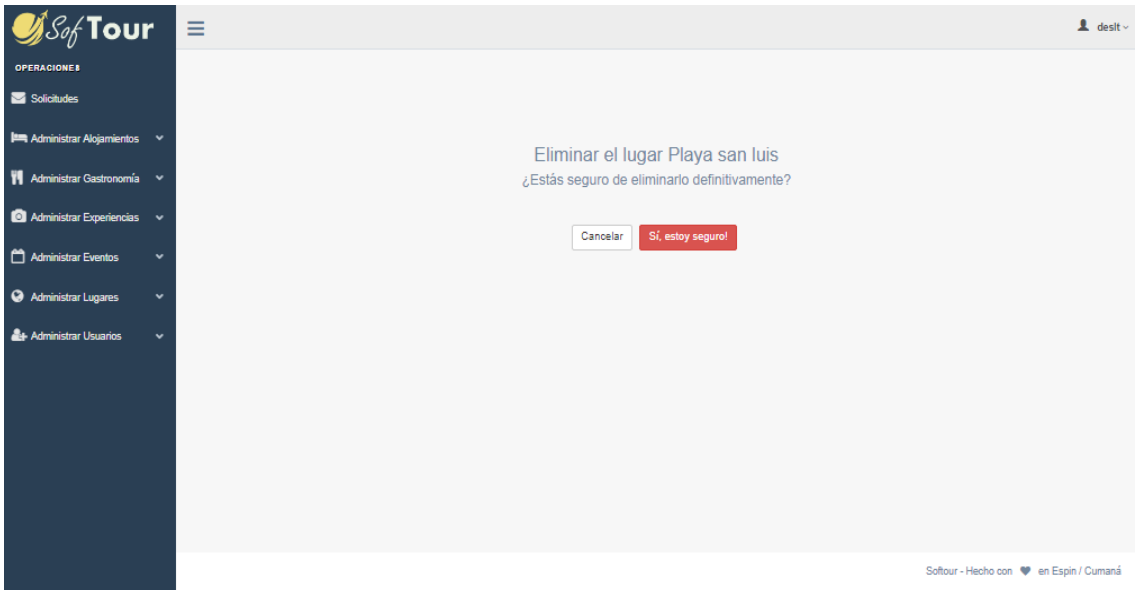


Figura G. 8. Interfaz de proceso de Eliminar Lugar por usuario Administrador.

Código desarrollado para la funcionalidad Eliminar Evento por usuario Prestador de Servicio.

```
eliminar(req, res, next) {
    let tipo_entidad = req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.
        substring(1);

    this._Lugar.eliminar({"url": req.params.url.toLowerCase(), "__t": tipo_entidad}, (error, eliminado) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_eliminar(req, res);
        }
        if(!eliminado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": {"url": "listado"}};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_eliminar(req, res);
    });
}
```

Figura G. 9. Código desarrollado para el proceso Eliminar Lugar por usuario Administrador.

Prueba Eliminar Lugar

La funcionalidad de este módulo de acuerdo a lo descrito en el caso de uso busca borrar de la base de datos del sistema un lugar agregado.

Para llevar a cabo esta prueba se toman los valores agregados con anticipación en las pruebas realizadas a la función Agregar lugar. A continuación, se muestran secuencias de pantallas al momento de realizar la prueba:

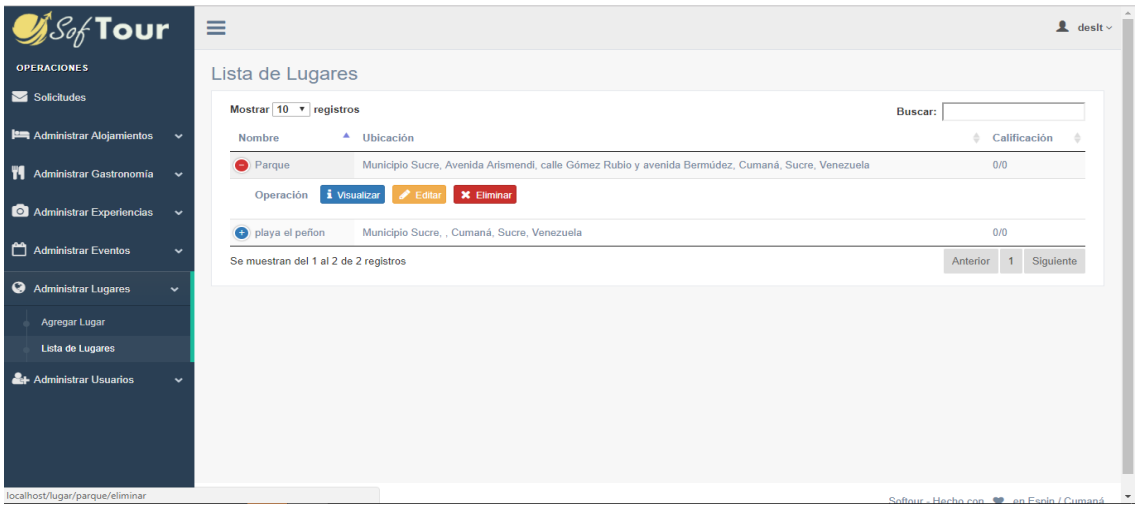


Figura G. 10. Prueba Eliminar Lugar desde el perfil Administrador.

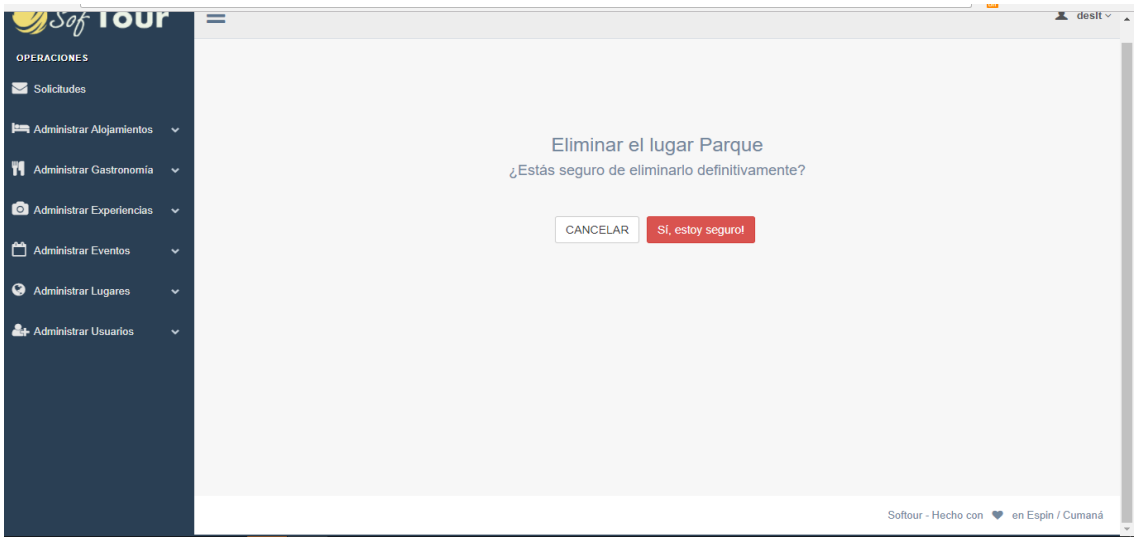


Figura G. 11.Prueba Eliminar Lugar desde el perfil Administrador.

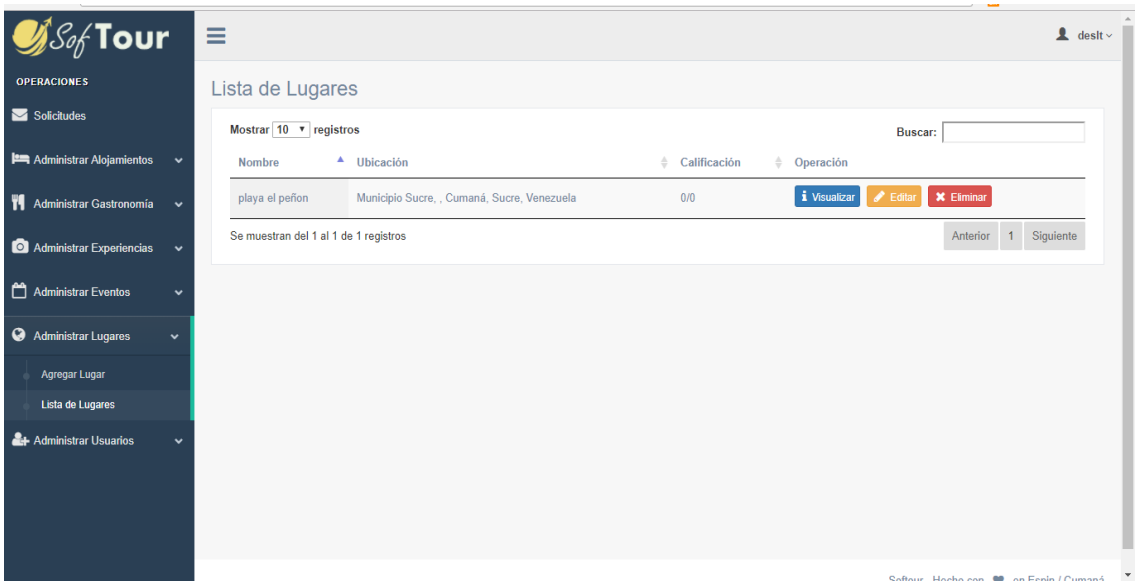


Figura G. 12.Prueba Eliminar Lugar desde el perfil Administrador.

Conclusión de la prueba

La función fue ejecuta con éxito.

Prueba de interfaz y usabilidad

Como se puede notar en la secuencia de pantallas de la prueba previa la interfaz continúa siendo constante, utilizando las bases heurísticas de Nielsen, como lo es el diseño minimalista, flexibilidad y uso de palabras que se ajustan al ambiente o realidad de cualquier usuario.

Apéndice H. Ciclo de Versión Administrador: Iteración Agregar Evento.

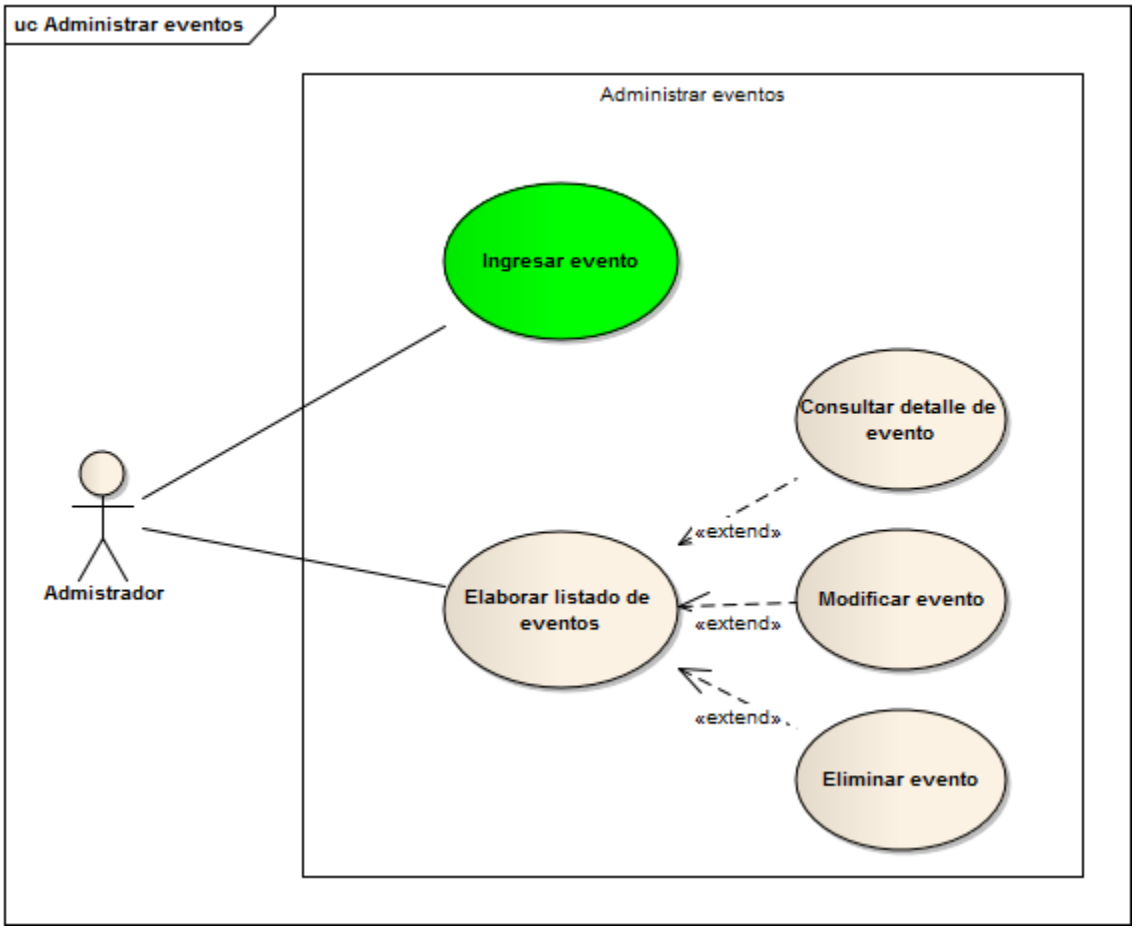


Figura H. 1. Caso de uso: Ingresar Evento por Administrador

Tabla H. 1. Descripción Caso de uso Ingresar Evento por usuario Administrador.

Nombre del Caso de uso	Ingresar evento.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en ingresar los atributos del evento de manera satisfactoria.	
Precondición	Para ingresar un evento, no debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Agregar evento”.
	2	El sistema muestra el formulario correspondiente.
	3	El Administrador ingresar los atributos del evento.
	4	El sistema almacena los atributos del evento y finaliza el caso de uso.
Postcondición	Se ingresa los atributos del evento en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no ingresar el evento y selecciona cancelar el caso de uso, seleccionando otra opción en el sistema.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. Para que un evento pueda ser ingresado no debe estar registrado en la base de datos del sistema.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

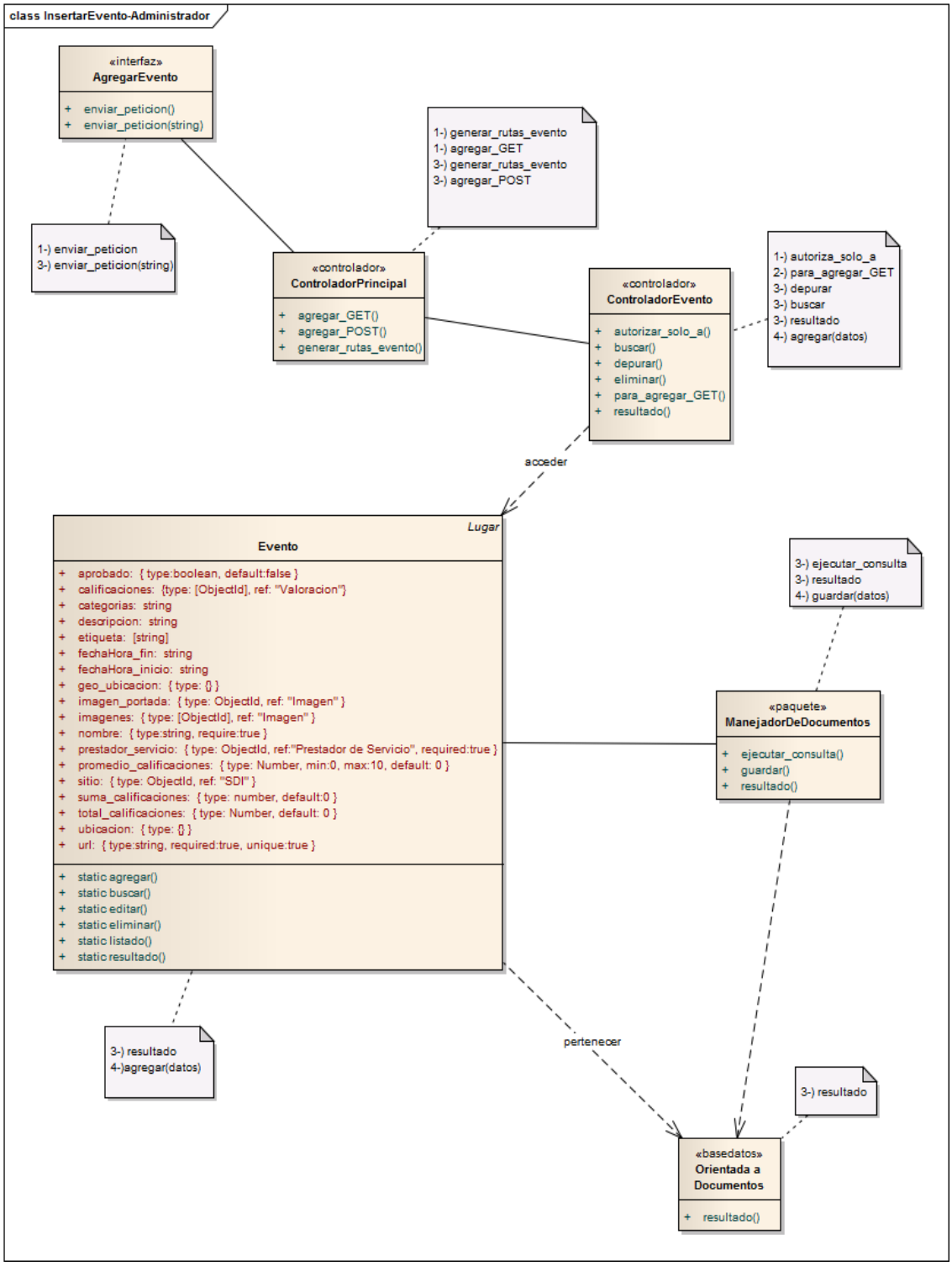


Figura H. 2. Diagrama de Clases: Ingresar Evento por usuario Administrador.

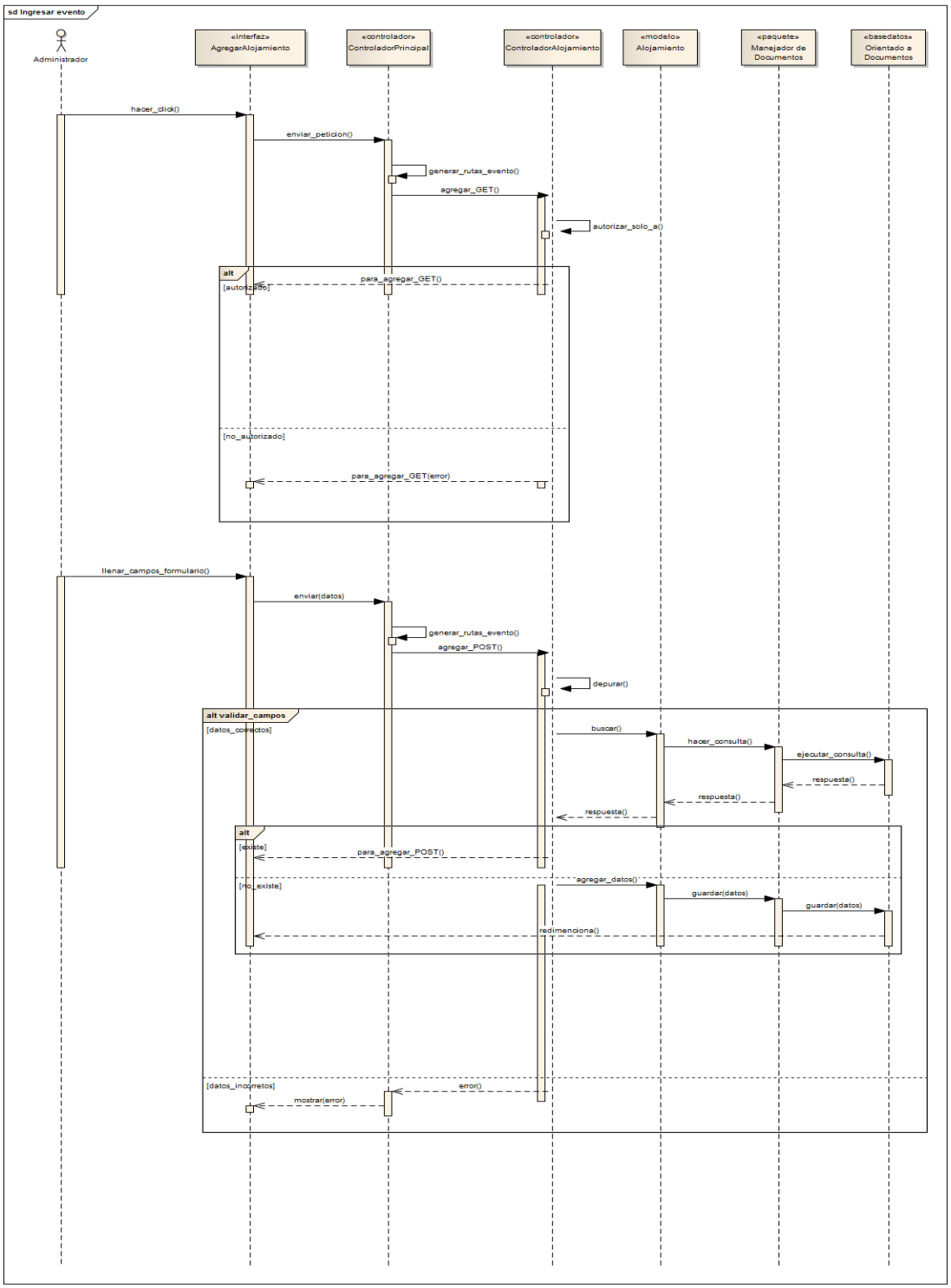


Figura H. 3. Diagrama de secuencia: Ingresar Evento por usuario Administrador

Figura H. 4. Colección Evento: ingresar evento por usuario Administrador

A continuación, se muestran los modelos de plantillas para el proceso de ingresar evento.

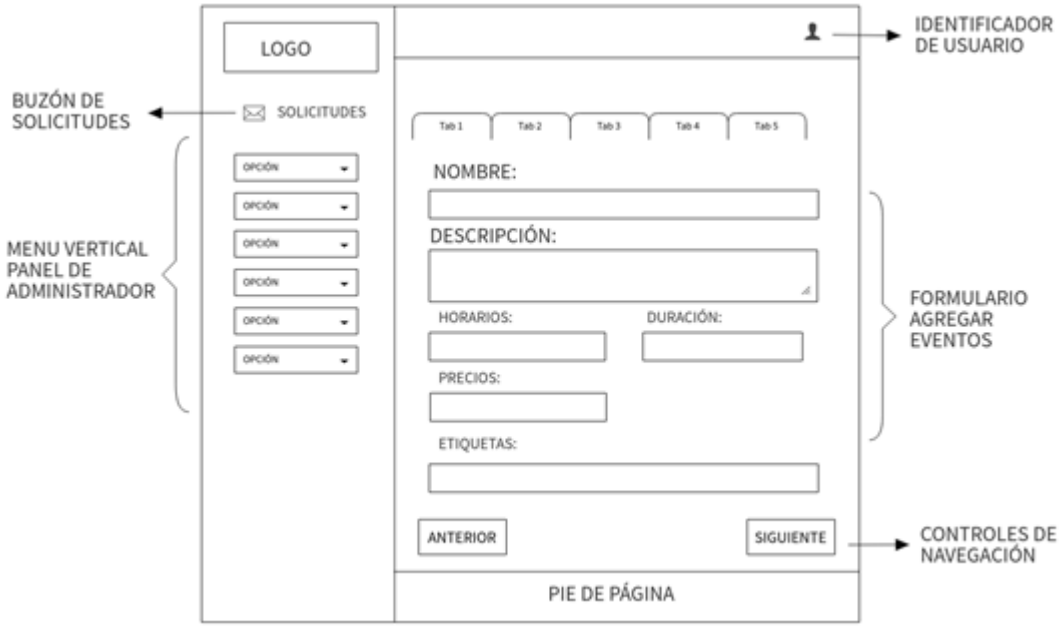


Figura H. 5. Plantilla de interfaz 1: Ingresar Evento por usuario Administrador.

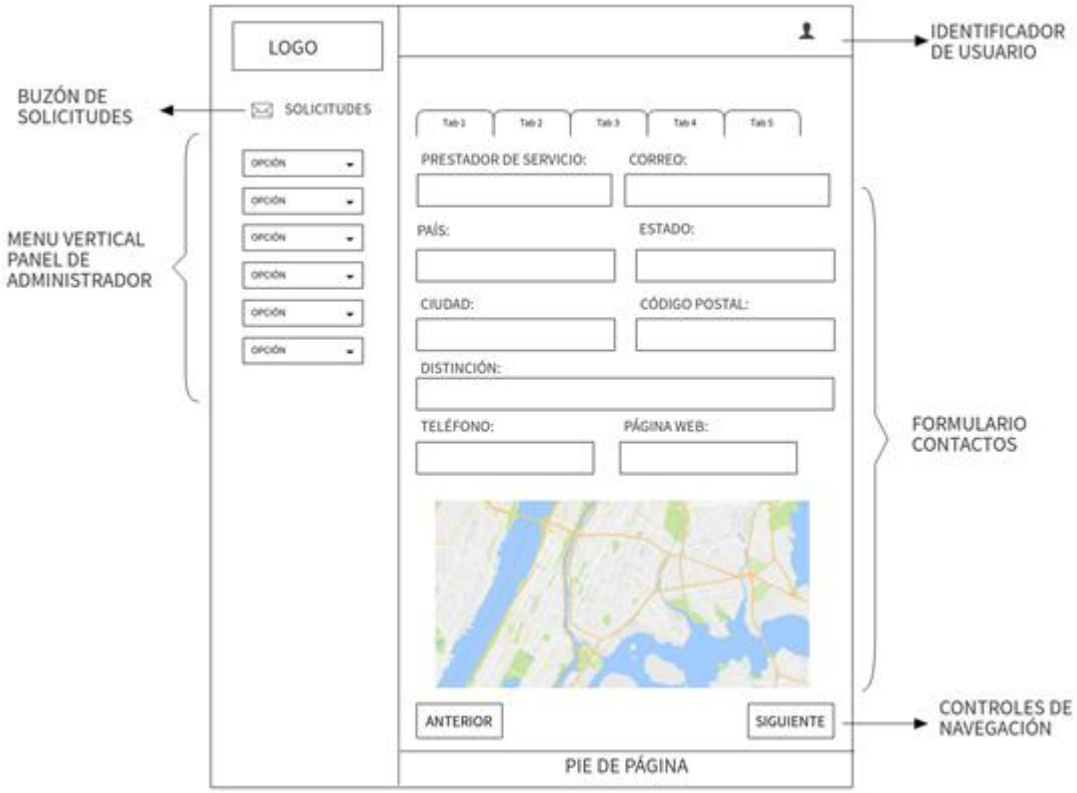


Figura H. 6. Plantilla de interfaz 2: Ingresar Evento por usuario Administrador.

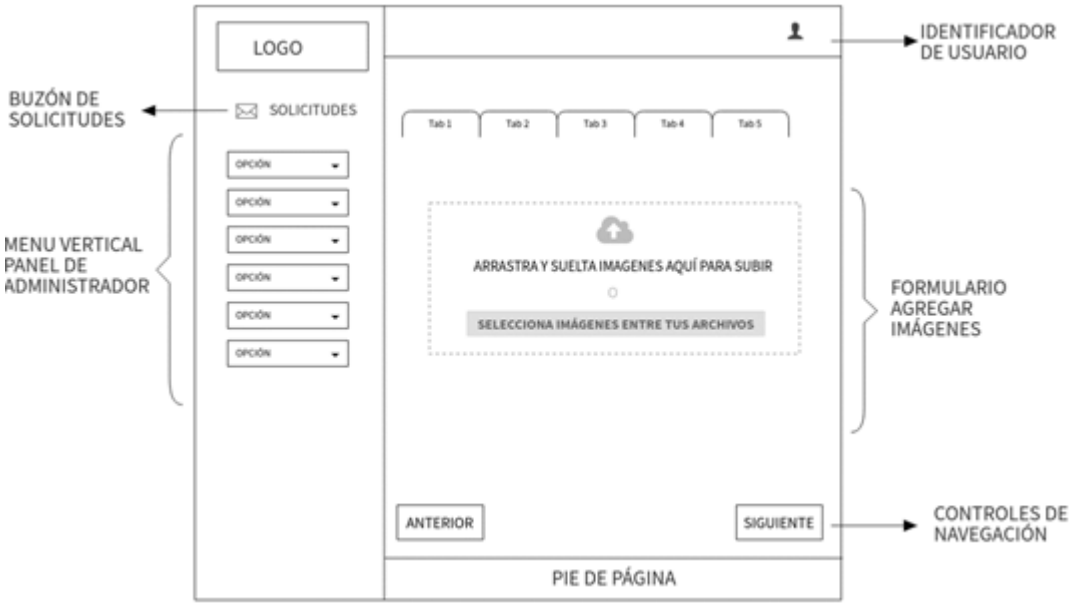


Figura H. 7. Plantilla de Interfaz 3: Ingresar Evento por usuario Administrador.

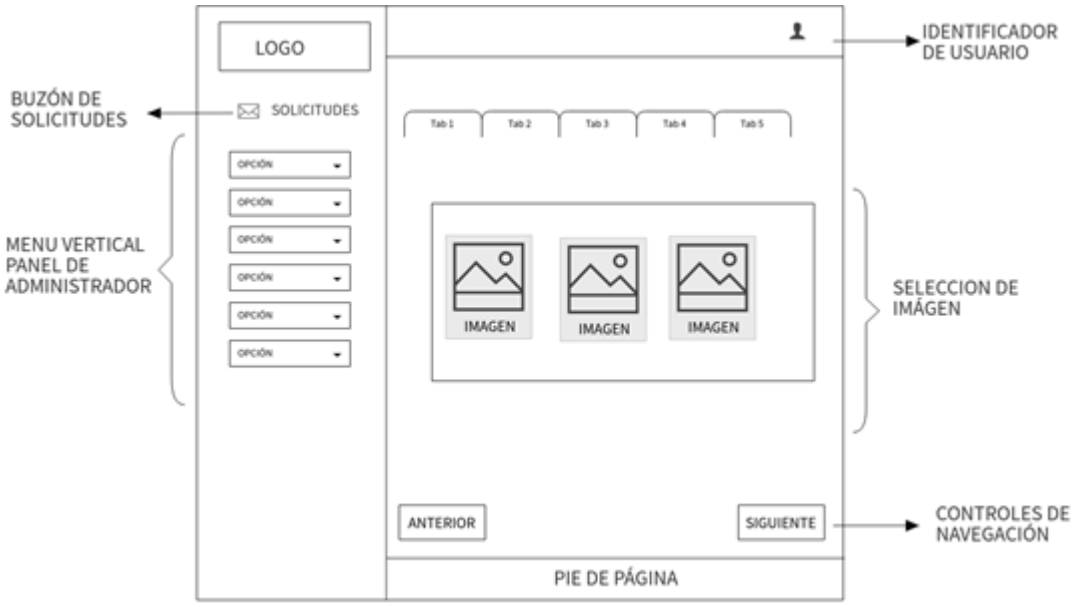


Figura H. 8. Plantilla de interfaz 4: Ingresar Evento por usuario Administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Ingresar Evento y el sistema responde mostrando el formulario correspondiente.

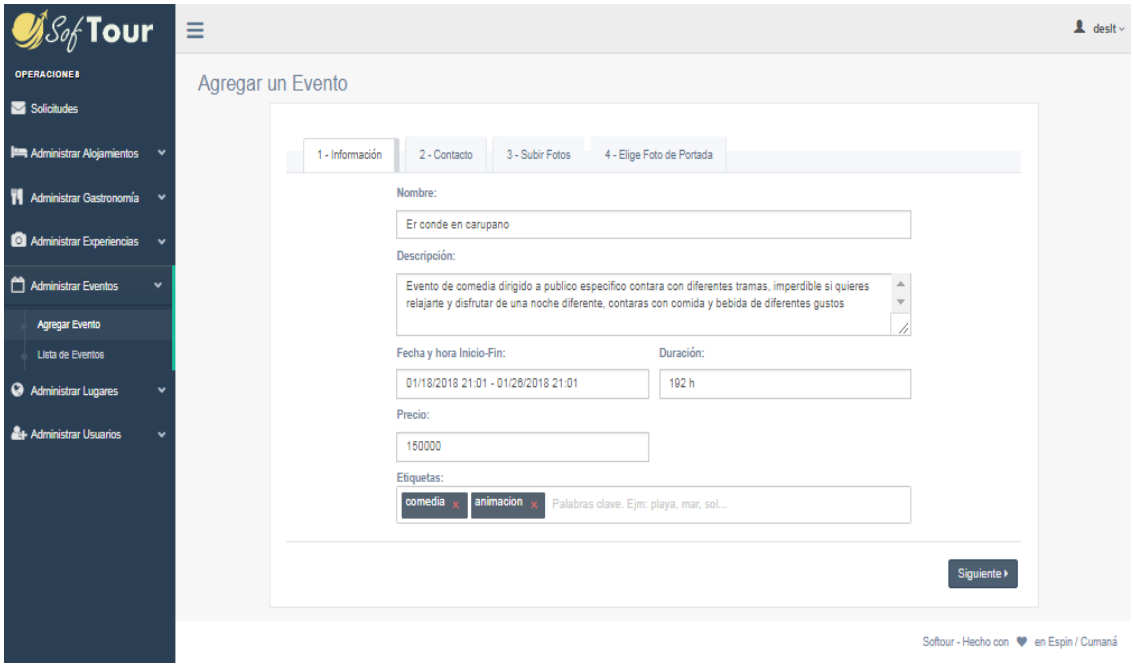


Figura H. 9. Interfaz 1 de proceso de Ingresar Evento por usuario Administrador

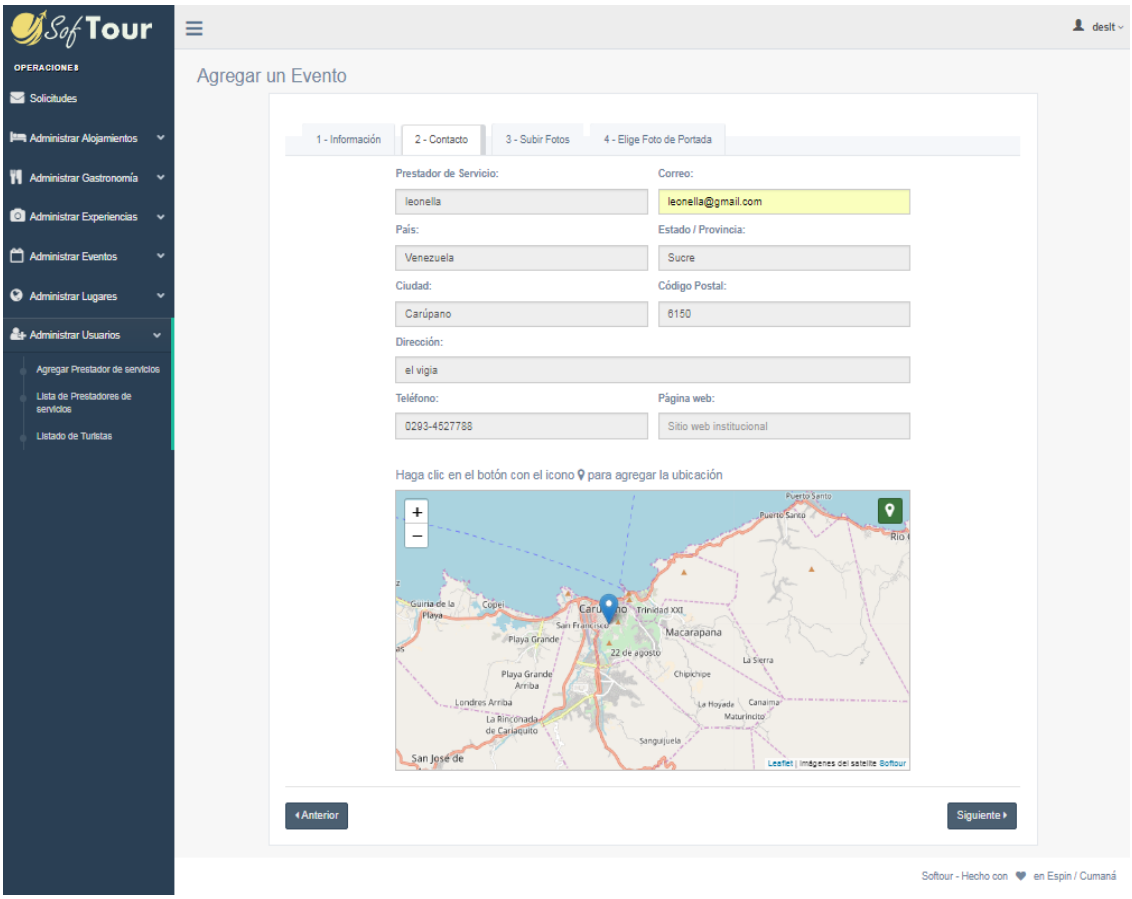


Figura H. 10. Interfaz 2 de proceso de Ingresar Evento por usuario Administrador.

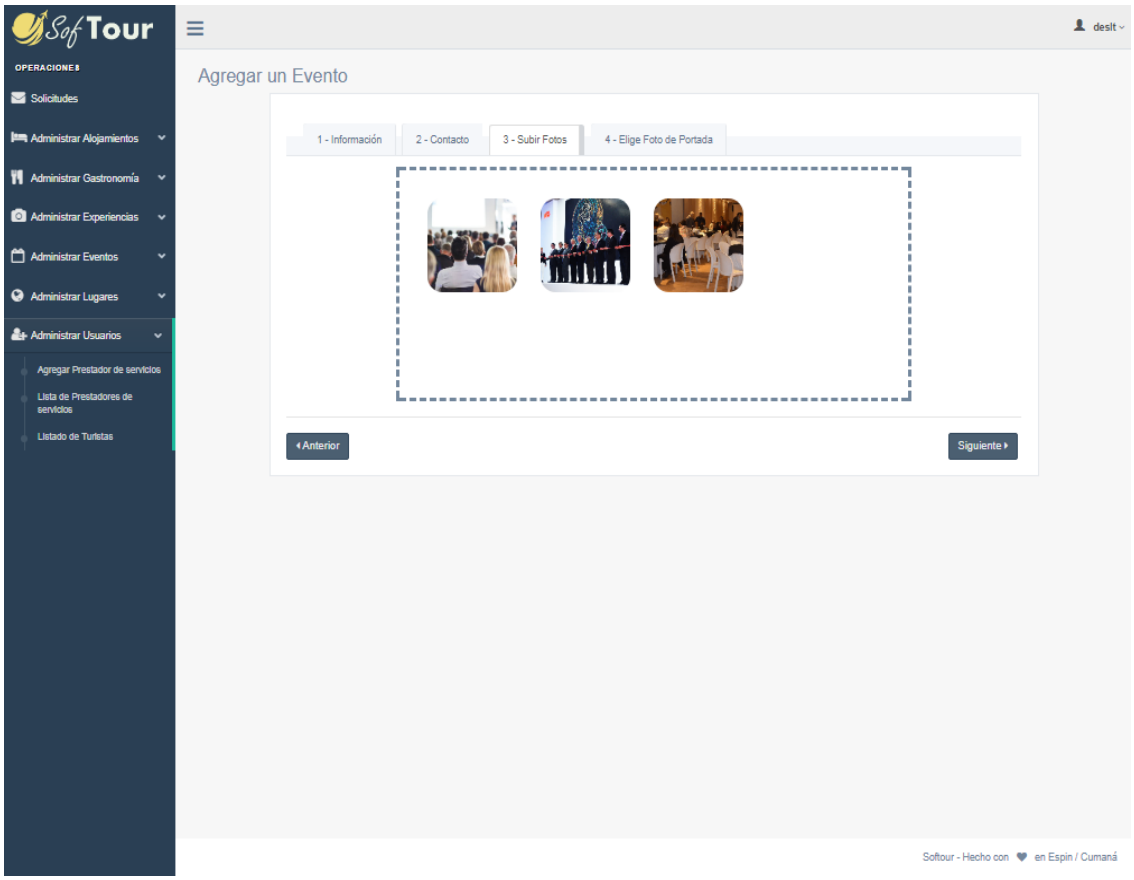


Figura H. 11.Interfaz 3 de proceso de Ingresar Evento por usuario Administrador

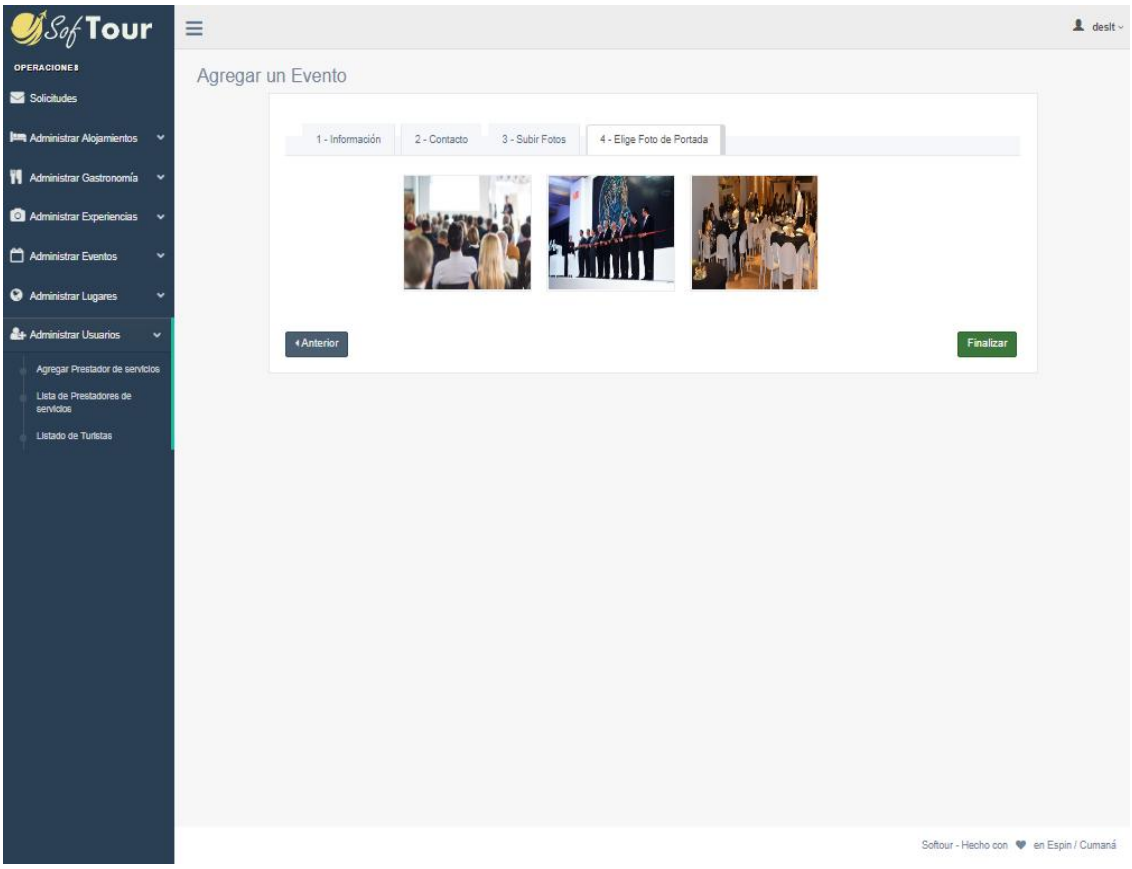


Figura H. 12. Interfaz 4 de proceso de Ingresar Evento por usuario Administrador

Código desarrollado para la funcionalidad Ingresar Evento por usuario Administrador

```
ingresar_POST(req, res, next) {
    this._validar.campos(req.body, false, (error, resultados) => {
        if(error) next(error);
        res.locals.datos = resultados;

        if (res.locals.datos.error) {
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;

            this._respuesta.para_ingresar_POST(req, res);
        }

        else {
            res.locals.datos.value.geo_ubicacion = {
                "type": "Point",
                "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
            };
            res.locals.datos.value.ubicacion = {
                "direccion": res.locals.datos.value.direccion,
                "pais": res.locals.datos.value.pais,
                "estado": res.locals.datos.value.estado,
                "ciudad": res.locals.datos.value.ciudad,
                "codigo_postal": res.locals.datos.value.codigo_postal
            };

            this._Evento.buscar({"url": res.locals.datos.value.url}, (error, evento) => {
                if (evento) {
                    res.locals.datos.error = { "details": [ {
                        "path": "nombre",
                        "message": 'Ya existe una evento con este nombre.'
                    } ] };
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_ingresar_POST(req, res);
                }

                else{
                    if(req.session.passport.user.tipo_usr == "Administrador")
                        res.locals.datos.value.aprobado = true;

                    this._Evento.ingresar(res.locals.datos.value, (error, creado) => {
                        if(error) {
                            res.locals.datos.error = error;

                            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                            this._respuesta.para_ingresar_POST(req, res);
                        }

                        res.locals.datos = {"value": creado};
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_ingresar_POST(req, res);
                    });
                }
            });
        }
    });
}

visualizar(req, res, next) {
    this._Evento.buscar_visualizar({"url": req.params.url.toLowerCase()}, (error, buscado) => {
        if(error) {
```

Figura H. 13. Código desarrollado para el proceso Ingresar Evento por usuario Administrador.

Prueba agregar evento

Para realizar la prueba de funcionalidad de este caso de uso desarrollado se continuó utilizando la técnica de caja negra, en donde se ingresan datos planteados en un caso de prueba y luego se procede a validar que cumpla con la finalidad para la cual fue planteada.

Ingresar evento tiene como finalidad capturar los datos de los eventos suministrados por el usuario Administrador y proceder a guardarlos en la base de datos, los datos que se deben ingresar en los campos deben cumplir con ciertos parámetros como se presentan a continuación:

Tabla H. 2. Clases de equivalencia Agregar Evento por Administrador.

Condición de entrada	Tipo	Clases de equivalencias válidas	Clases de equivalencias no válidas
Nombre	Alfanumérico	1:Cadena de caracteres, números.	2: Campo en blanco
Descripción	Alfanumérico	1:Cadena de caracteres, números.	2: Campo en blanco
Fecha inicio	Selección		2: Campo en blanco
Duración	Alfanumérico	1:Cadena de caracteres, números.	2: Campo en blanco
Precio	Alfanumérico	1:Cadena de caracteres, números.	2: Campo en blanco
Etiqetas	Alfanumérico	1: Cadena de caracteres, números.	
Prestador de servicio	Selección		2:Campo en blanco
Correo	Correo	3: símbolos, caracteres, números	2: Campo en blanco 4: formato invalido

País	Alfabético	1: Cadena de caracteres	5: Símbolos 6:números
Estado/provincia	Alfabético	1: Cadena de caracteres	5: Símbolos 6:números
Ciudad	Alfabético	1: Cadena de caracteres	5: Símbolos 6:números
Código postal	Numérico	6: números	1: Cadena de caracteres
Dirección	Alfanumérico	1:Cadena de caracteres, números.	2: Campo en blanco
Teléfono	Numérico	6: números	1: Cadena de caracteres
Pagina web	Alfanumérico	1:Cadena de caracteres, números.	
Selección mapa	selección		
Subir foto	Imagen	7: JPG, GIF, PNG	2: Campo en blanco
Seleccionar foto	selección		

De acuerdo a lo expresado en la tabla anterior se presentan los siguientes casos de prueba para la evaluación de la funcionalidad:

Tabla H. 3. Casos de prueba Agregar Evento por Administrador.

Condición de entrada	A1	A2	A3
Nombre	Amanecer Llanero	Futbol sala Cesarger	1…2…3… Play
Descripción	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,

	ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.	ullamco laboris nisi ut aliquip ex ea commodo consequat.	
Fecha inicio	12-10	10-12	01-01
Duración	6 horas	3 horas	24 horas
Precio	200 $	400 Bs	50 Bs
Etiquetas	Música; amanecer; arpa	Futbol; deporte;	
Prestador de servicio	Selección	Selección	Selección
Correo	Llaneros@vene.com	futbol@depor.com	play@rumbo.com
País	Venezuela	Venezuela	Venezuela
Estado/provincia	Sucre	Sucre	Sucre
Ciudad	Cumaná	Cumanacoa	Carúpano
Código postal	6101	mjsh	1441
Dirección	Avenida perimetral, concha custica	Gimnasio vertical	
Teléfono	0293123456	029454845kl	02934518578
Pagina web	Llaneras.com	Deporte.ve	Play.es
Selección mapa	Selección	Selección	Selección
Subir foto	JPG, PNG	PDF	
Seleccionar foto	Selección	Selección	Selección
Clases Equivalencia	1,3,4,5,6,7	1,2,3,4,5	1,2,3,4
Resultado	Registro satisfactorio	Registro no procede por datos no válidos.	Registro no procede por datos no válidos.

Se inicia la prueba seleccionando agregar un nuevo evento desde el panel de Administrador

Como se muestra en la secuencia de imágenes, en el caso de prueba A1 se consiguió un registro satisfactorio:

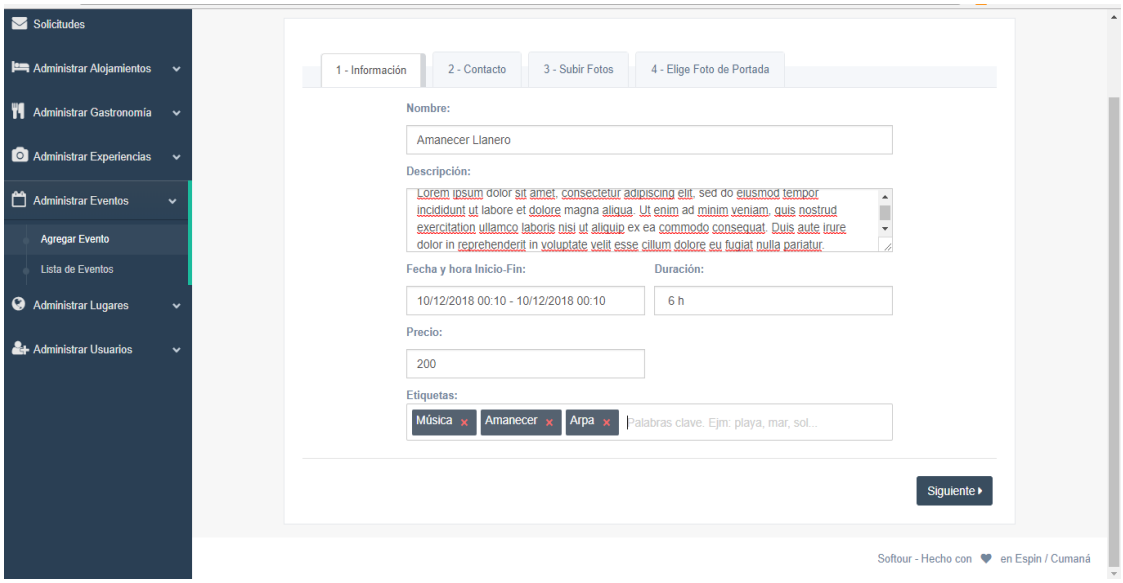


Figura H. 14.Caso de pruebas A1 Ingresar evento por Administrador. Paso 1.

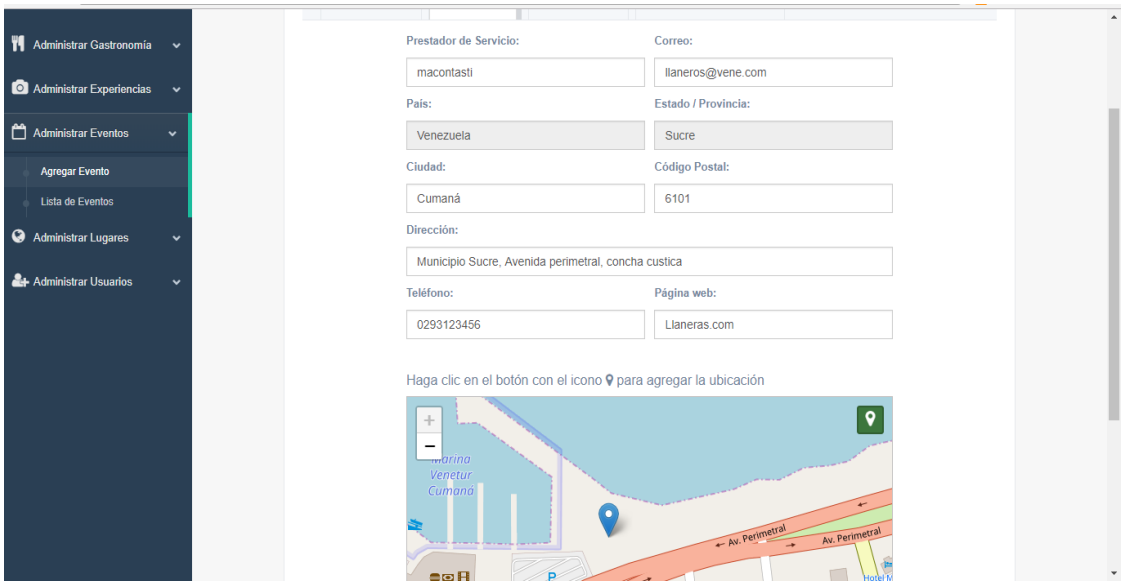


Figura H. 15. Caso de pruebas A1 Ingresar Evento por Administrador. Paso 2

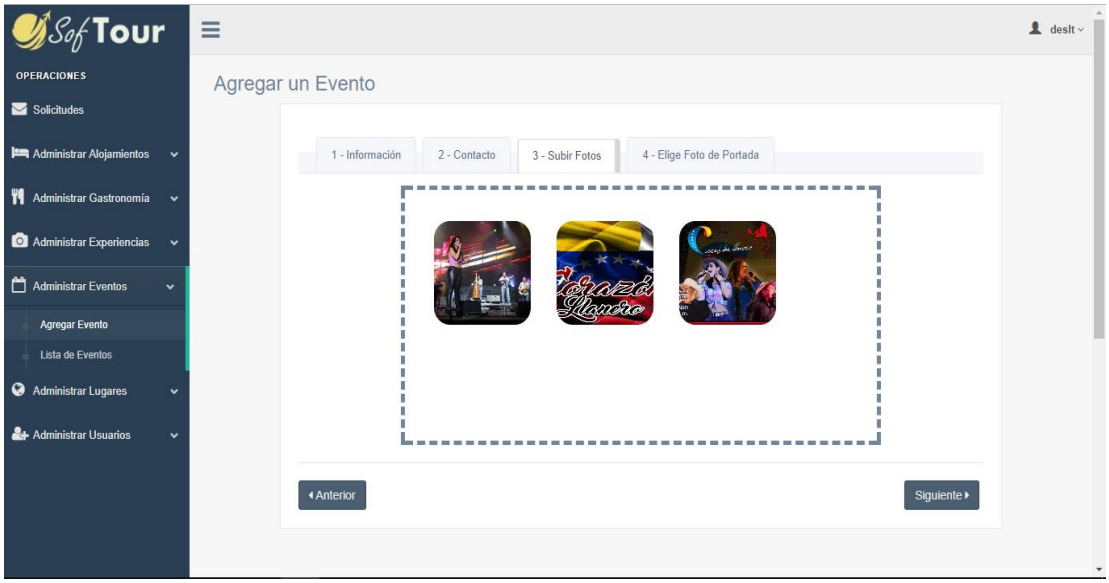


Figura H. 16.Caso de pruebas A1 Ingresar Evento por Administrador. Paso 3.

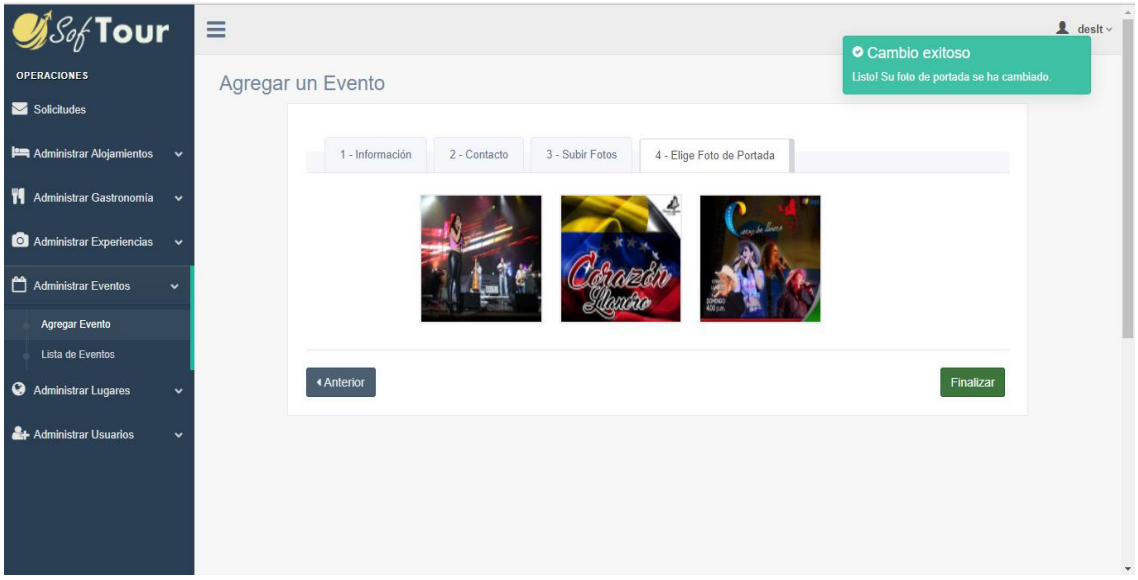


Figura H. 17.Caso de pruebas A1 Ingresar evento por Administrador. Paso 4.

Para el caso de pruebas A2 el registro no fue satisfactorio ya que el sistema está configurado para no permitir entrada de caracteres alfabéticos en información que requiere caracteres numéricos, es decir, el sistema cumplió con su validación.

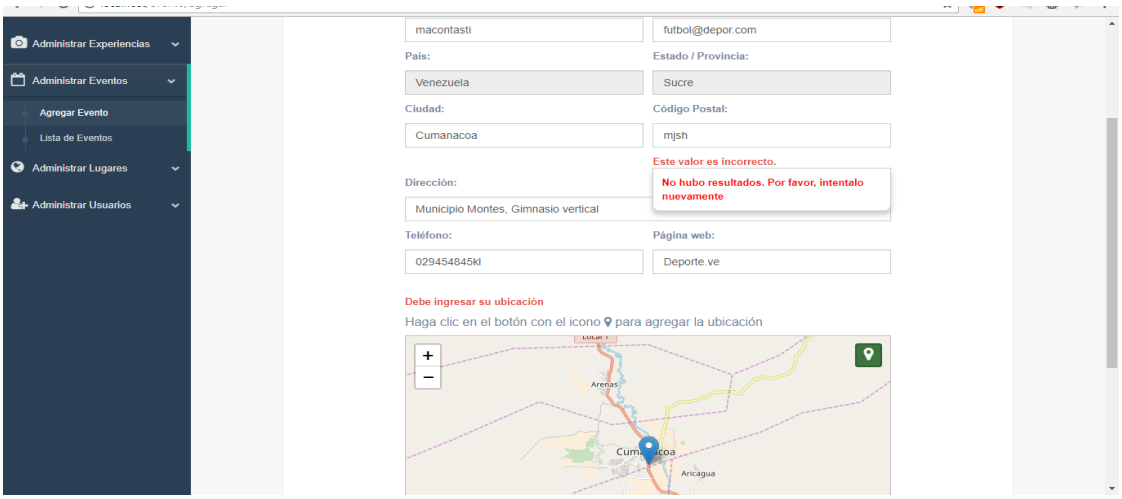


Figura H. 18.Caso de pruebas A2 Ingresar evento por Administrador.

Para el caso de prueba A3 de Agregar Evento, se planteó ingresar un evento con caracteres alfanuméricos y el sistema no permitió el ingreso, además de que valido el campo descripción, ya que este establece un mínimo de 150 caracteres.

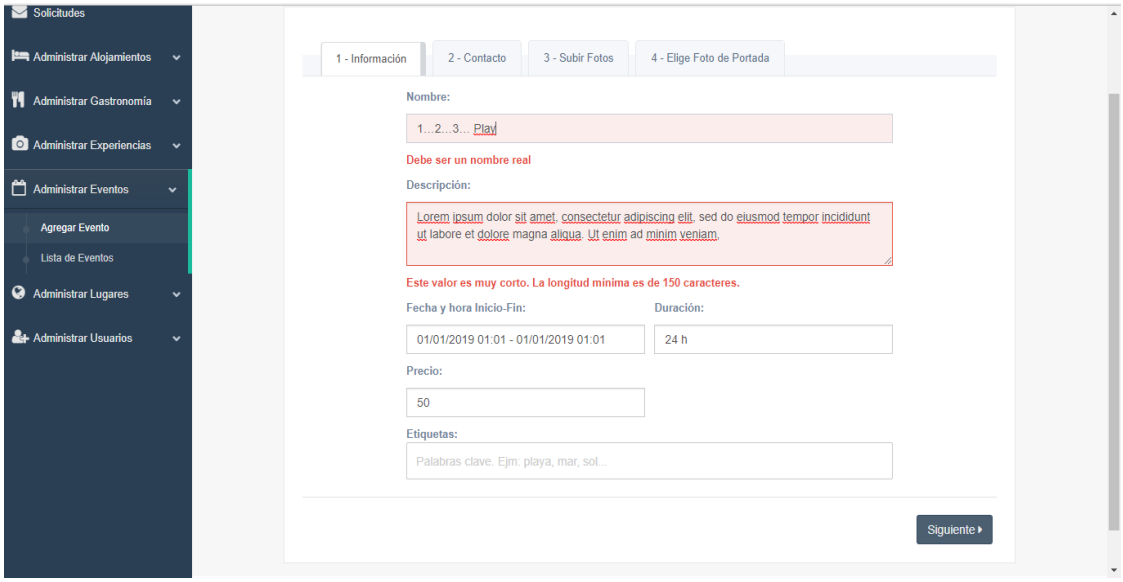


Figura H. 19.Caso de pruebas A3 Ingresar Evento por Administrador.

Conclusión de la prueba

La función cumple el objetivo para la cual fue implementada, pero se tomó en consideración que el nombre del evento debe ser corregido para permitir caracteres alfanuméricos ya que puede darse el caso de que un evento lleve números en su nombre. Esta última observación se le hizo llegar al equipo de desarrollo.

Prueba de interfaz y usabilidad

La interfaz de Agregar Evento mantiene los mismos estándares aplicados en las interfaces que ya han sido evaluadas, es decir, son consistentes en color, visibilidad, recorrido, diseño y uso de metáforas, lo que quiere decir que se adapta a los principios heurísticos de usabilidad.

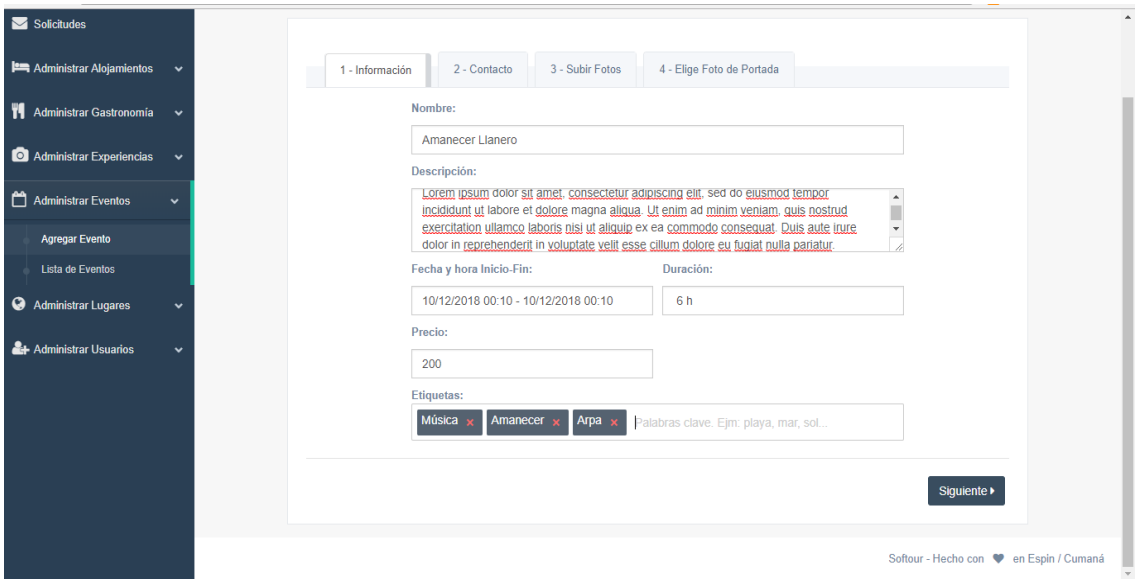


Figura H. 20.Imagen interfaz principal Ingresar Evento

Apéndice I. Ciclo de Versión Administrador: Iteración Listar Eventos.

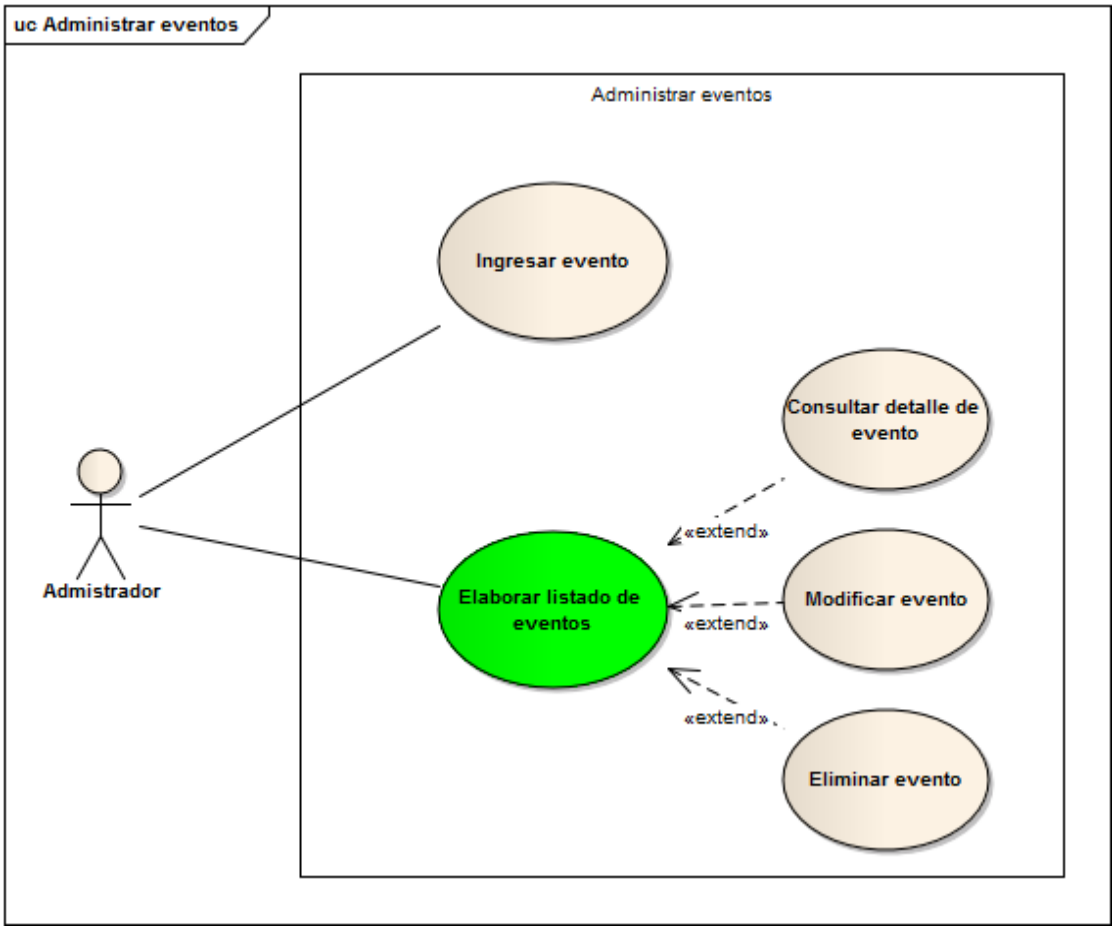


Figura I. 1. Caso de uso: Elaborar Listado de Eventos por Administrador.

Tabla I. 1. Descripción Caso de uso Elaborar Listado de Eventos por usuario Administrador.

Nombre del Caso de uso	Elaborar listado de eventos.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en elaborar un listado de los eventos registrados en la de base de datos del sistema, de manera satisfactoria.	
Precondición	Para elaborar el listado de los eventos, deben estar registrados en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Lista de eventos”.
	2	El sistema muestra el listado de los eventos almacenados en la base de datos del sistema, de acuerdo a un orden predeterminado, brindando las siguientes opciones de extensión: “Consultar evento”. “Modificar evento” “Eliminar evento”
	3	El Administrador visualiza el listado de los eventos y selecciona alguna de las opciones de extensión propuestas.
	4	El sistema extiende <<extend>> hacia el caso de uso que representa la opción seleccionada.
	5	El sistema finaliza el caso de uso.
Postcondición	Se elabora el listado de eventos existentes en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no seleccionar ninguna opción de extensión, se cancela el caso de uso, seleccionando otra opción en el sistema
	3.2.	El sistema finaliza el caso de uso.
Reglas del negocio	El listado de eventos debe ser elaborado de acuerdo a un orden preestablecido. Se podrá seleccionar, desde el listado general de eventos, un evento en específico para realizar las operaciones de consultar, modificar o eliminar.	
Inclusiones	Ninguna.	
Extensiones	“Consultar evento”. “Modificar evento”. “Eliminar evento”.	

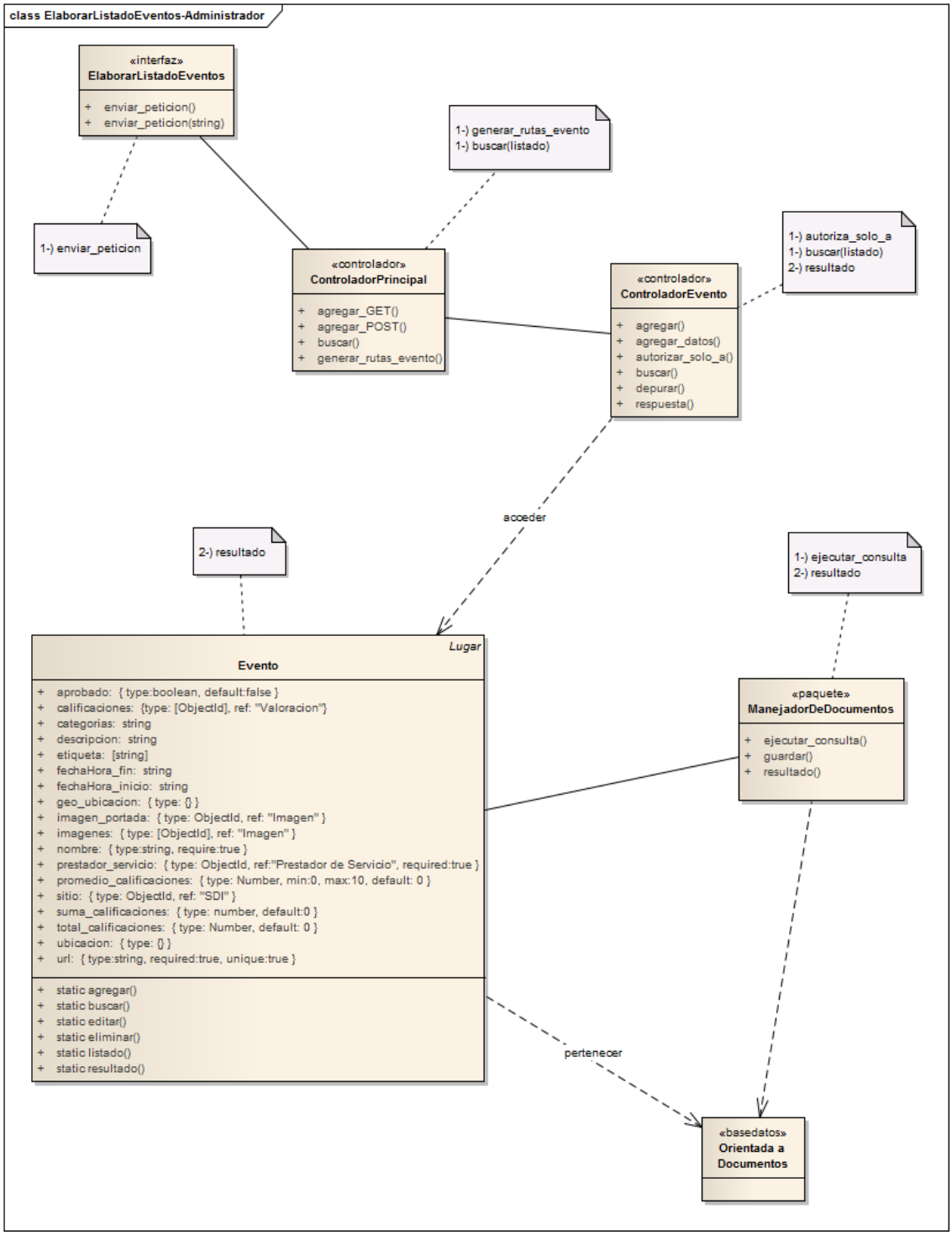


Figura I. 2.Diagrama de Clases: Elaborar Listado de Eventos por usuario Administrador.

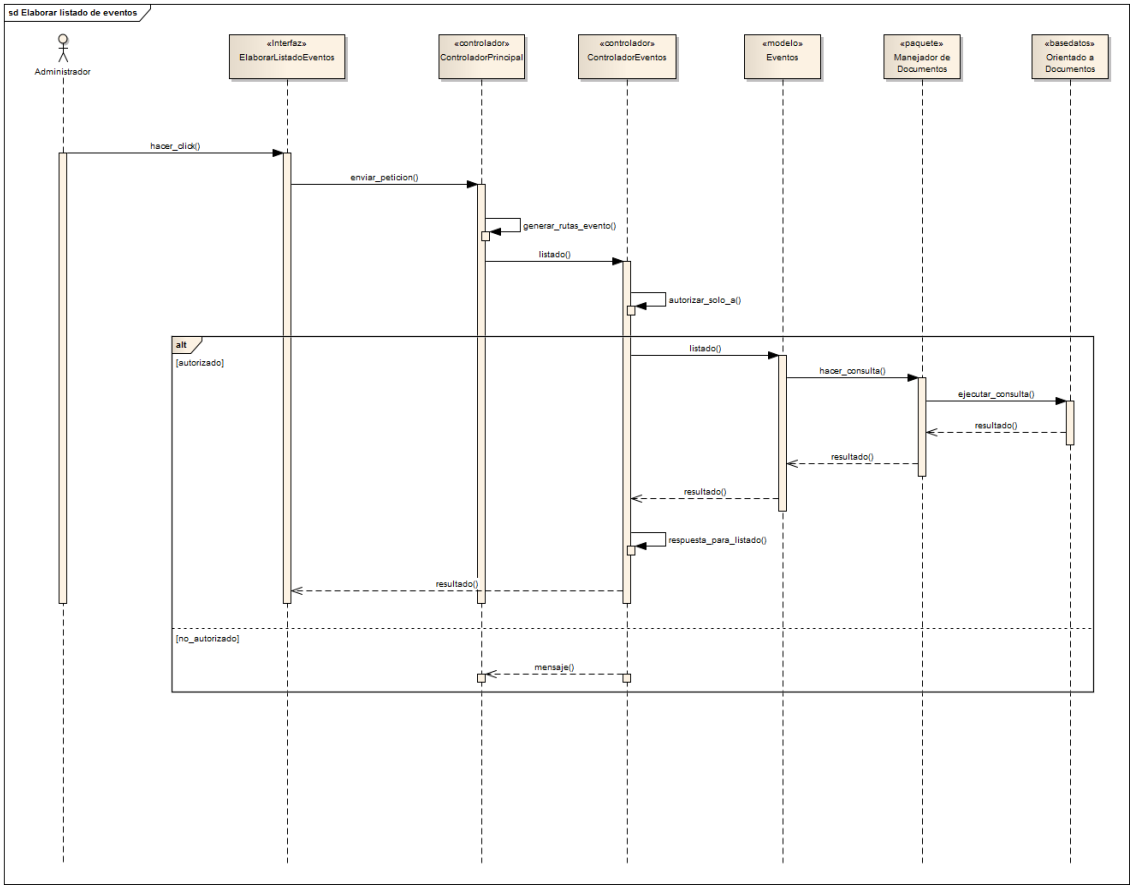


Figura I. 3.Diagrama de secuencia: Elaborar Listado de Eventos por usuario Administrador.

Figura I. 4. Colección Evento: Elaborar Listado Evento por usuario Administrador.

A continuación, se muestra el modelo de plantilla para el proceso de Listar Evento.

Figura I. 5. Plantilla de interfaz: Listado de Eventos por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Lista de Eventos y el sistema responde mostrando la interfaz correspondiente.

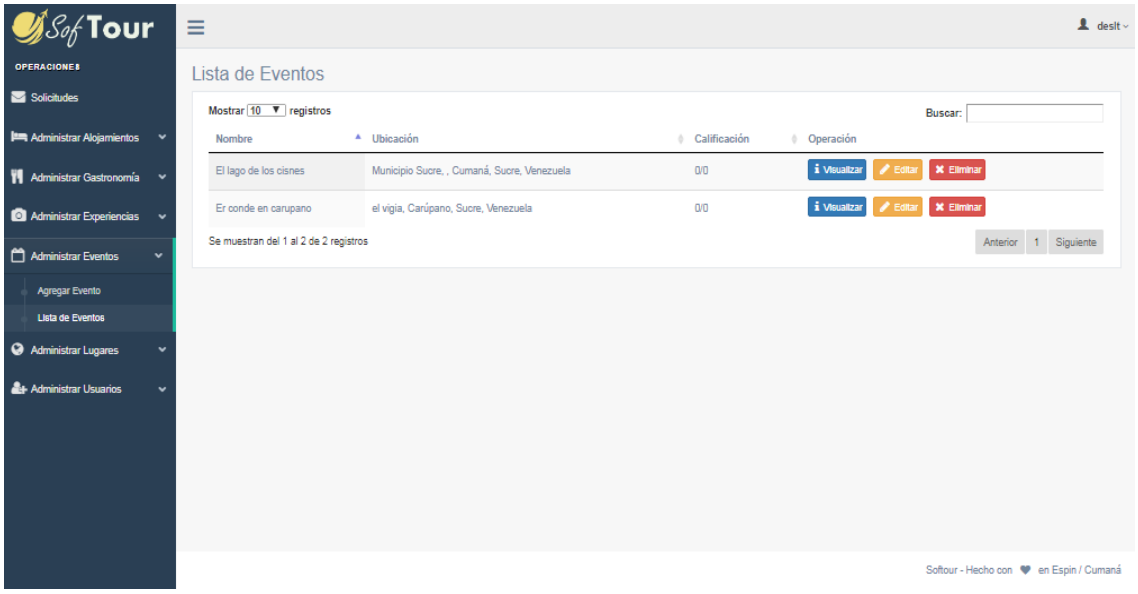


Figura I. 6. Interfaz Listado de Eventos por usuario Administrador.

Código desarrollado para la funcionalidad Listado de Eventos por usuario Administrador

```
listado(req, res, next) {
    let query = { "aprobado": true };

    if(req.session.passport.user.tipo_usr == "PrestadorServicio")
        query.prestador_servicio = req.session.passport.user.id;

    this._Evento.listado({"consulta": query}, (error, lista) => {

        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_listado(req, res);
        }

        res.locals.datos = {"value": lista};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_listado(req, res);
    })
}
```

Figura I. 7. Código desarrollado para el proceso Listar Evento por usuario Administrador.

Apéndice J. Ciclo de Versión Administrador: Iteración Consultar Evento.

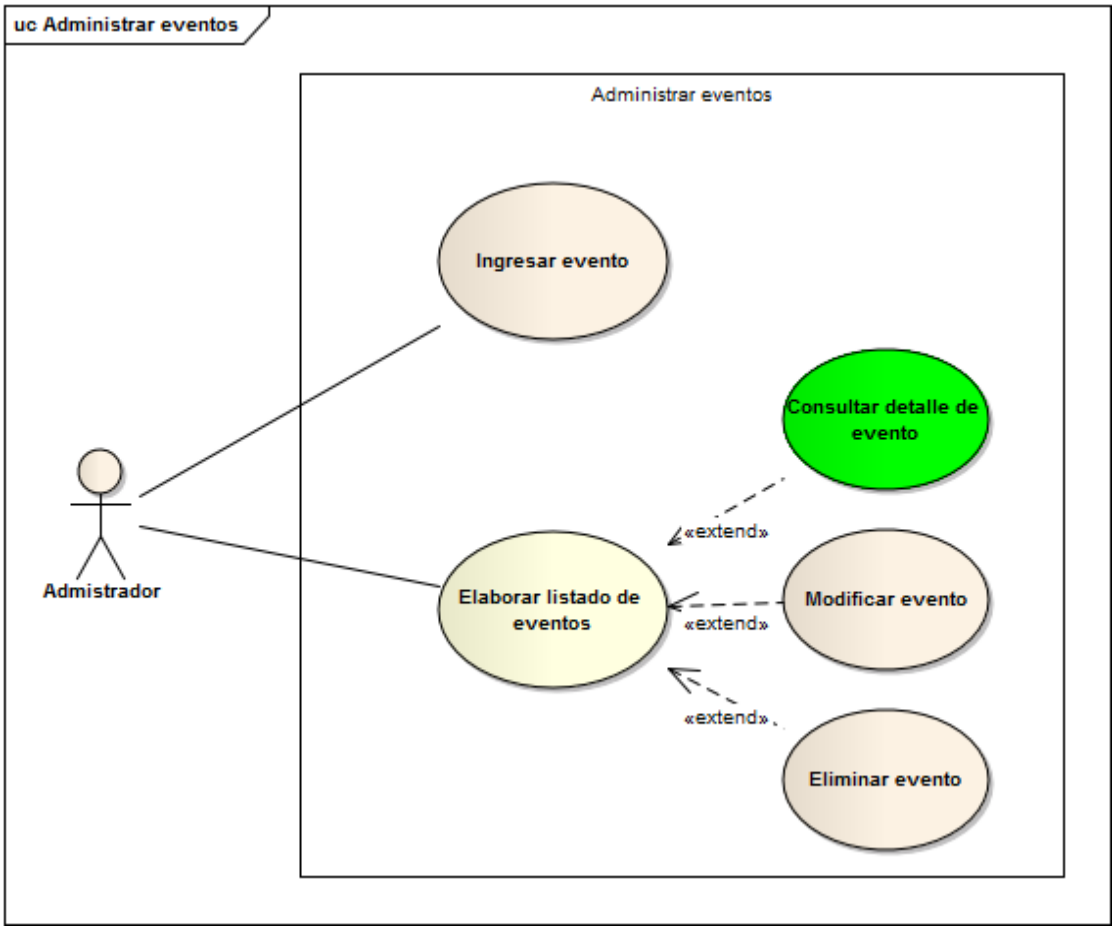


Figura J. 1.Caso de uso: Consultar Evento por Administrador.

Tabla J. 1. Descripción Caso de uso Consultar Evento por usuario Administrador.

Nombre del Caso de uso	Consultar evento.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en consultar los atributos del evento de manera satisfactoria.	
Precondición	Para consultar un evento, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Consultar”.
	2	El sistema ejecuta la búsqueda del evento.
	3	El Administrador visualiza la consulta de los atributos del evento.
	4	El sistema finaliza el caso de uso.
Postcondición	Se consultan los atributos del evento de la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
Reglas del negocio	1. El evento debe estar registrado para que pueda ser consultado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

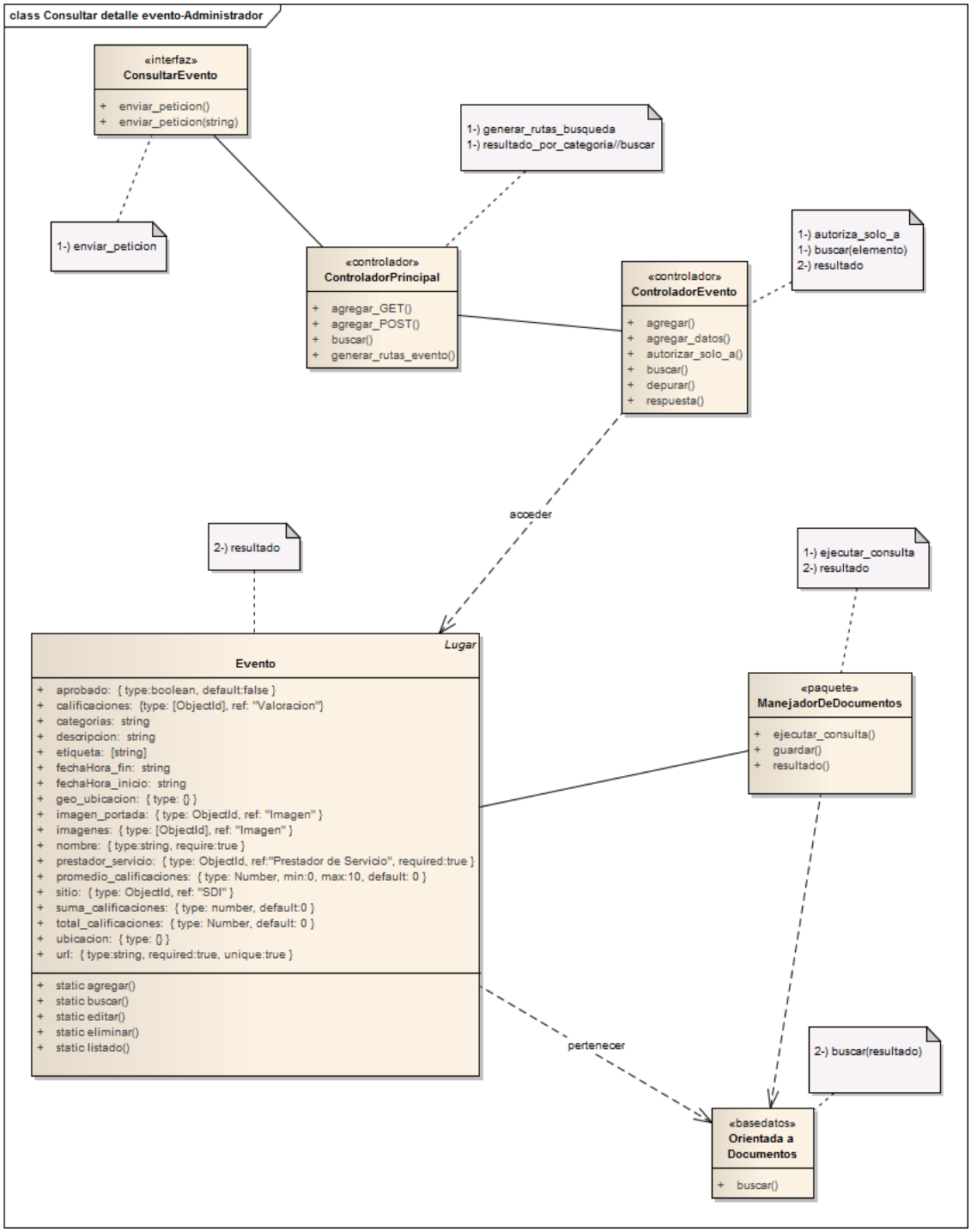


Figura J. 2.Diagrama de Clases: Consultar Evento por usuario Administrador

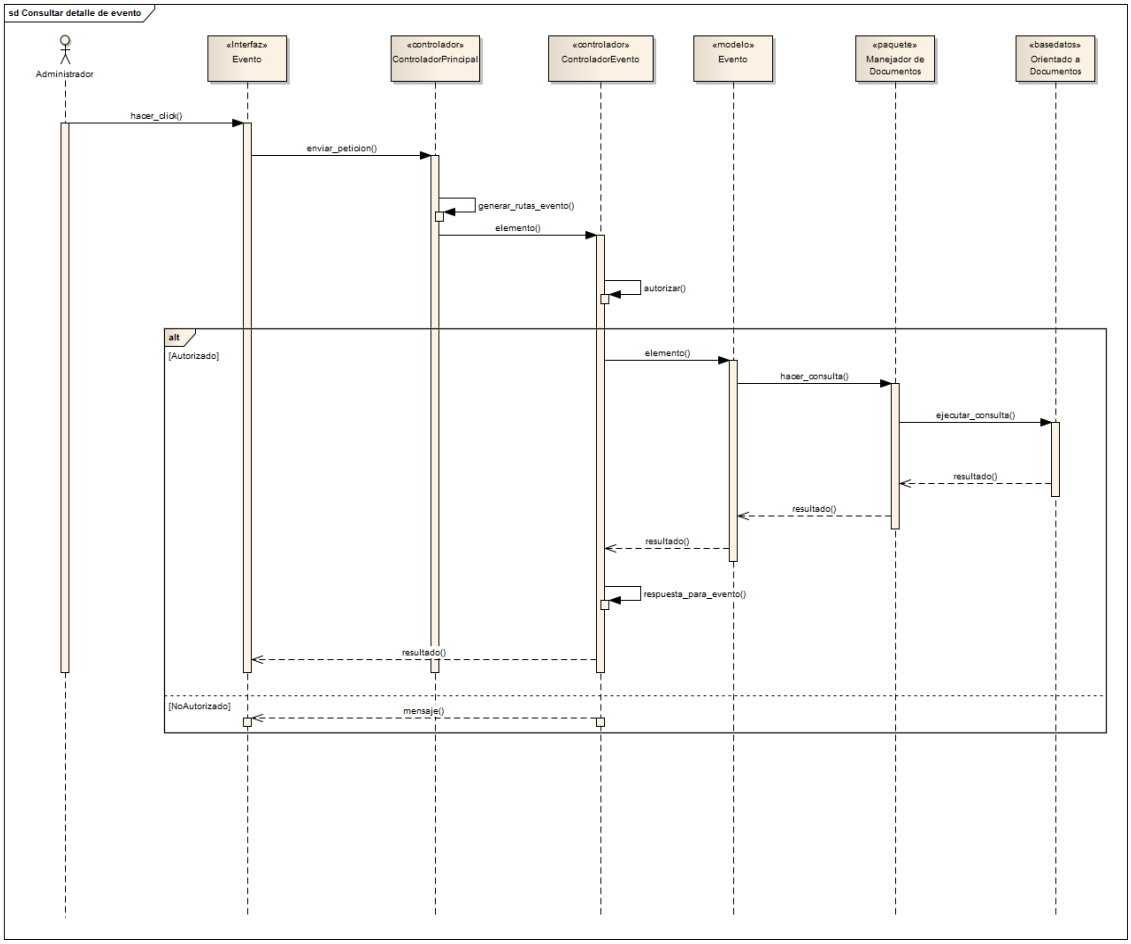


Figura J. 3. Diagrama de secuencia: Consultar Evento por usuario Administrador.

Figura J. 4. Colección Evento: consultar evento por usuario Administrador

A continuación, se muestra el modelo de plantilla para el proceso de consultar evento.

Figura J. 5. Plantilla de interfaz: Consultar Evento por usuario Administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Consultar Evento y el sistema responde mostrando la interfaz con las especificaciones del Evento.

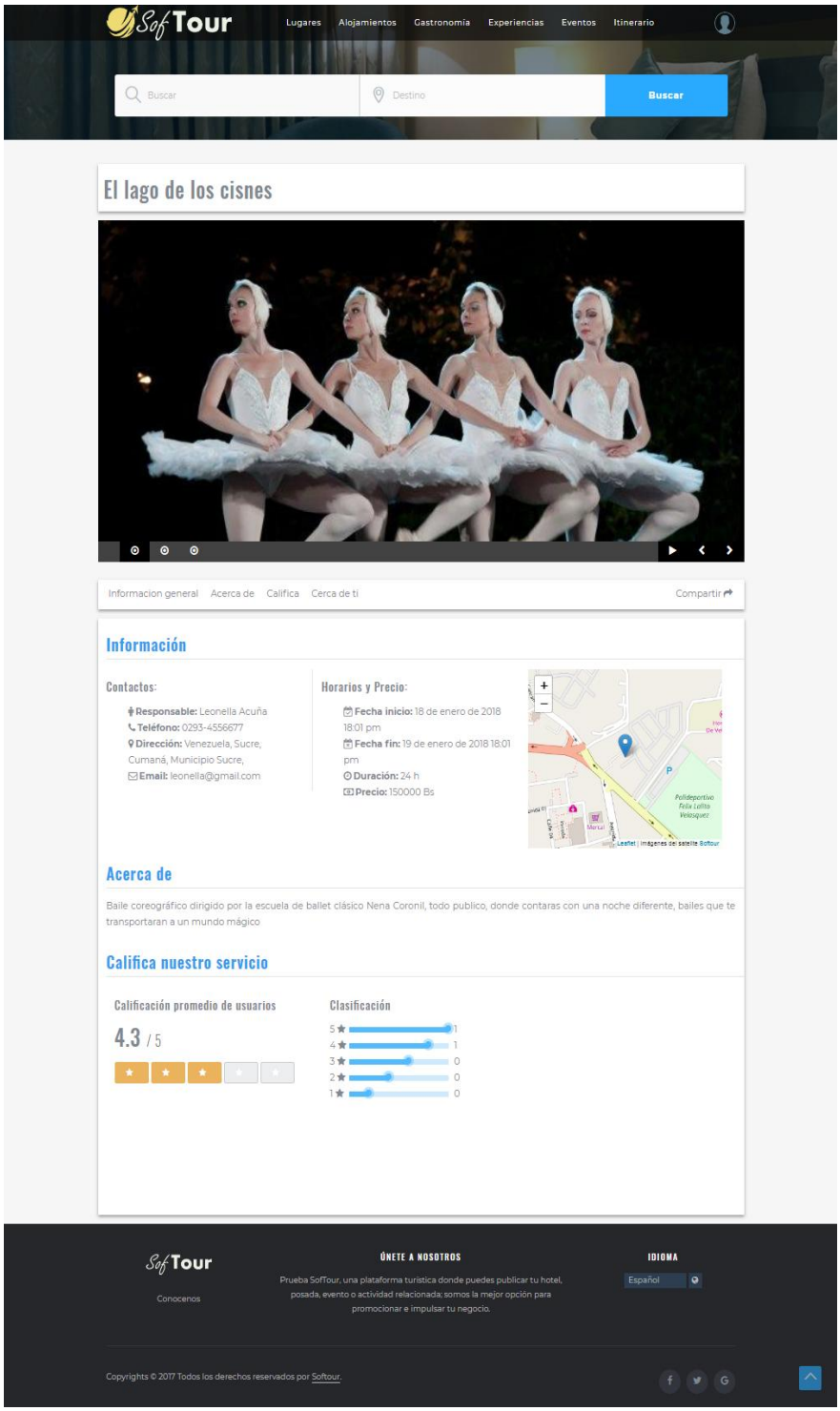


Figura J. 6. Interfaz de proceso de Consultar Evento por usuario Administrador.

Código desarrollado para la funcionalidad Consultar Evento por usuario Administrador

```
consultar(req, res, next) {
    this._Evento.buscar_consulta({"url": req.params.url.toLowerCase()}, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": buscado};
        if (req.isAuthenticated())
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_consultar(req, res);
    })
}
```

Figura J. 7. Código desarrollado para el proceso Consultar Evento por usuario Administrador.

Prueba Consultar Evento

Para llevar a cabo la prueba de esta función se tomaron en cuenta los datos ingresados de la prueba realizada a la función ingresar evento.

La forma de realizar esta prueba fue manual y consistió en ingresar al listado de eventos, seleccionar un evento cualquiera y certificar que la información encontrada en el coincide con la que se registró en pruebas anteriores.

En la siguiente secuencia de imágenes se muestra cómo se llevó a cabo una de las pruebas para validar la función.

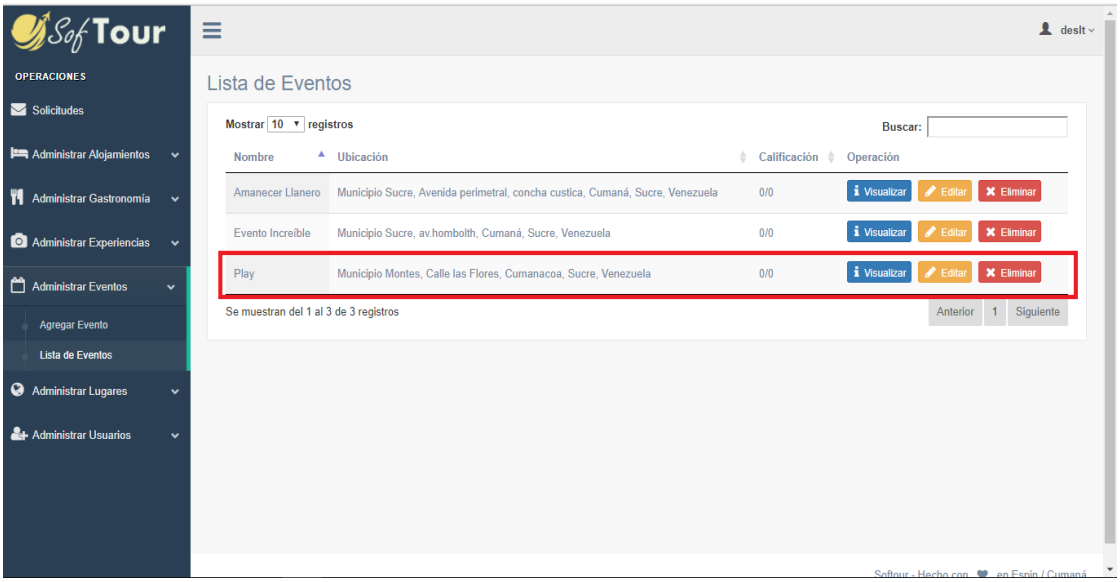


Figura J. 8.Prueba Consultar Evento por administrador. Listado de eventos registrados en sistema.

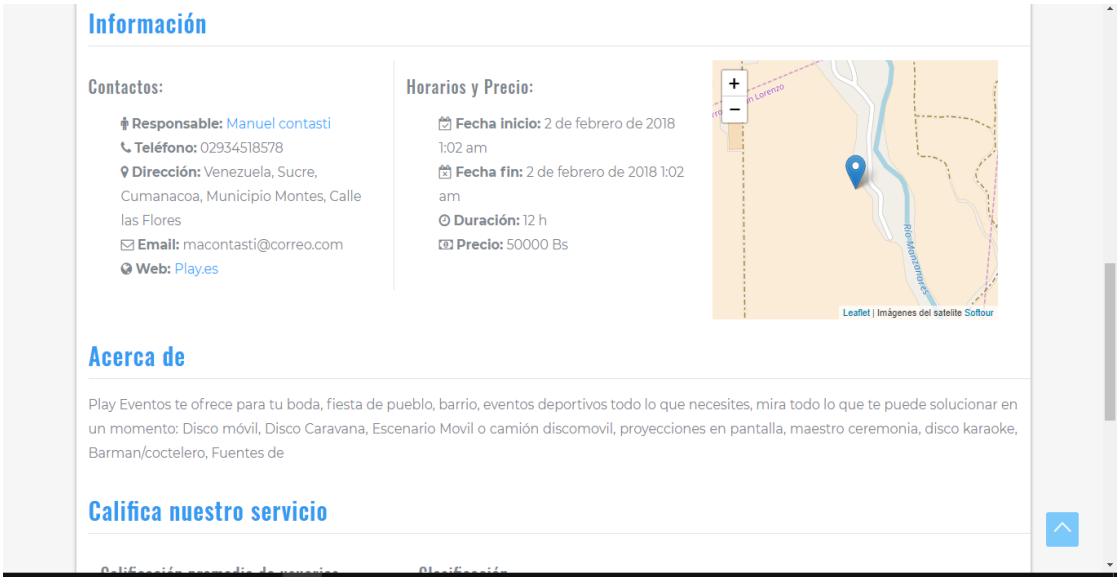


Figura J. 9.Prueba Consultar Evento por Administrador. Información de evento seleccionado.

Conclusión de la prueba

La prueba cumple con el objetivo para la cual fue desarrollada.

Prueba de interfaz y usabilidad

Esta interfaz se encuentra incluida entre el grupo de interfaz de usuario, por ende, al igual que pruebas anteriores cuenta con principios de heurística de usabilidad, está organizada de acuerdo a lo que se encuentra en la palestra del diseño web y cuenta con colores que diferencie las funciones.

Apéndice K. Ciclo de Versión Administrador: Iteración Modificar Evento.

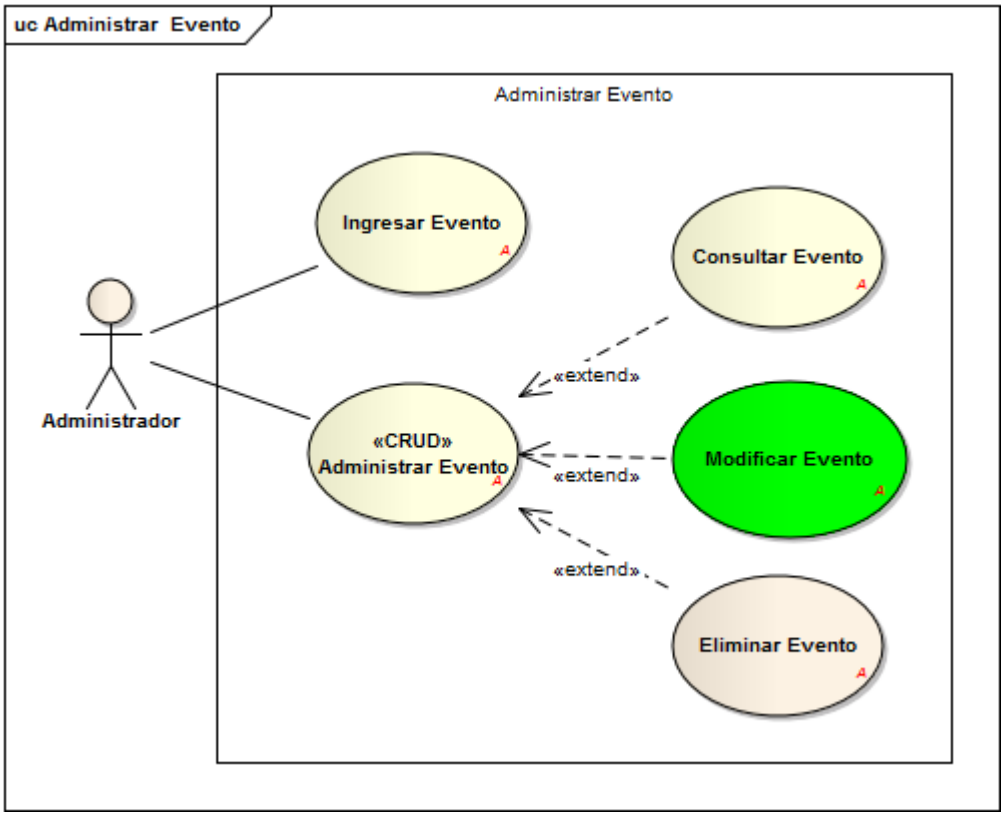


Figura K. 1.Caso de uso: Modificar Evento por Administrador

Tabla K. 1. Descripción Caso de uso Modificar Evento por usuario Administrador.

Nombre del Caso de uso	Modificar evento.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en modificar los atributos del evento de manera satisfactoria.	
Precondición	Para modificar un evento, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Modificar evento".
	2	El sistema ejecuta la búsqueda del evento, mostrando un formulario con la consulta en modo edición.
	3	El Administrador visualiza la consulta de los atributos del evento que desea modificar, los modifica y selecciona aceptar.
	4	El sistema realiza las modificaciones en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se modifican los atributos del evento en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no modificar el evento y selecciona cancelar el caso de uso, seleccionando otra opción en el sistema.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. El evento debe estar registrado para que pueda ser modificado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

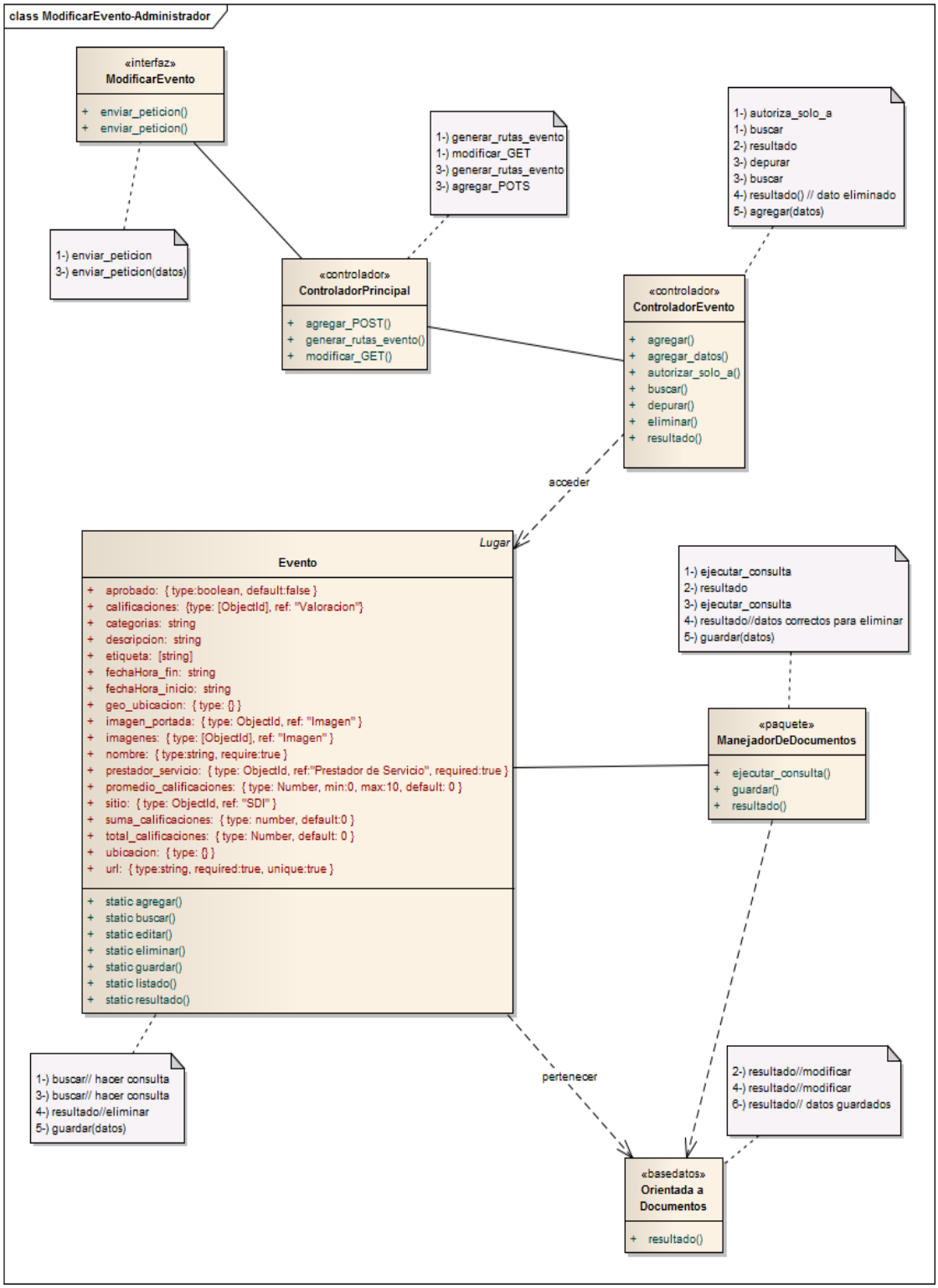


Figura K. 2. Diagrama de Clases: Modificar Evento por usuario Administrador.

Figura K. 3.Colección de Evento: Modificar Evento por usuario Administrador.

A continuación, se muestran los modelos de plantillas para el proceso de modificar evento.

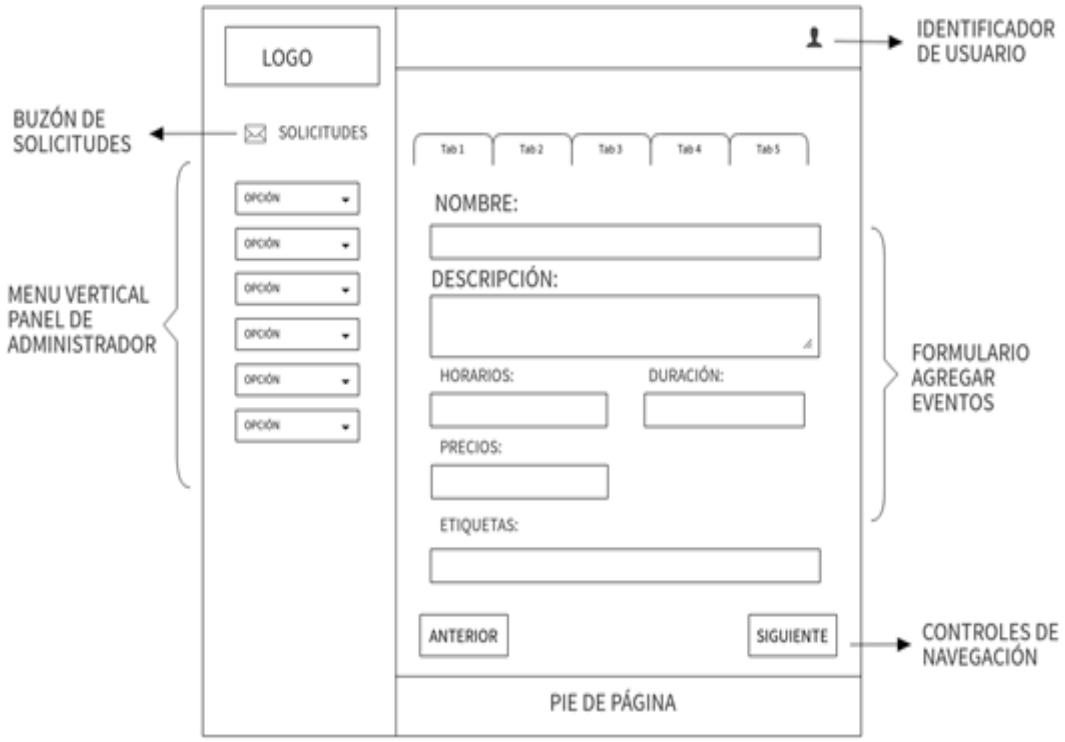


Figura K. 4. Plantilla de interfaz 1: Modificar Evento por usuario Administrador.

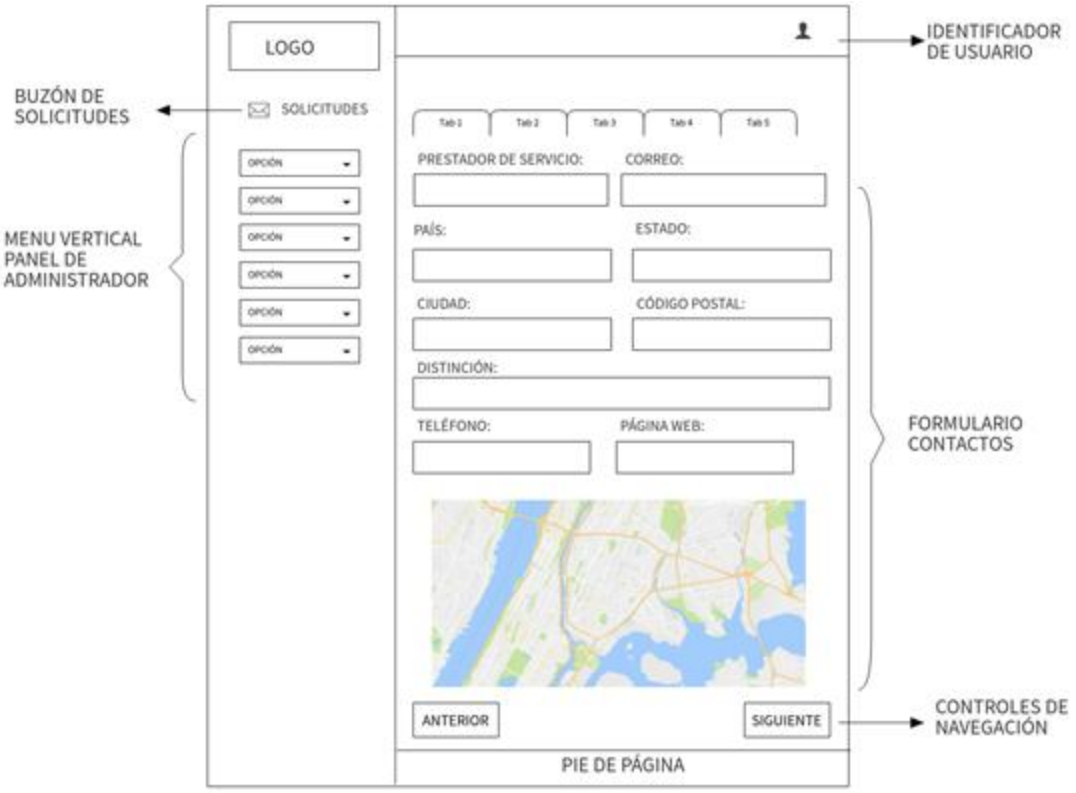


Figura K. 5. Plantilla de interfaz 2: Modificar Evento por usuario Administrador.

Figura K. 6. Plantilla de interfaz 3: Modificar Evento por usuario Administrador.

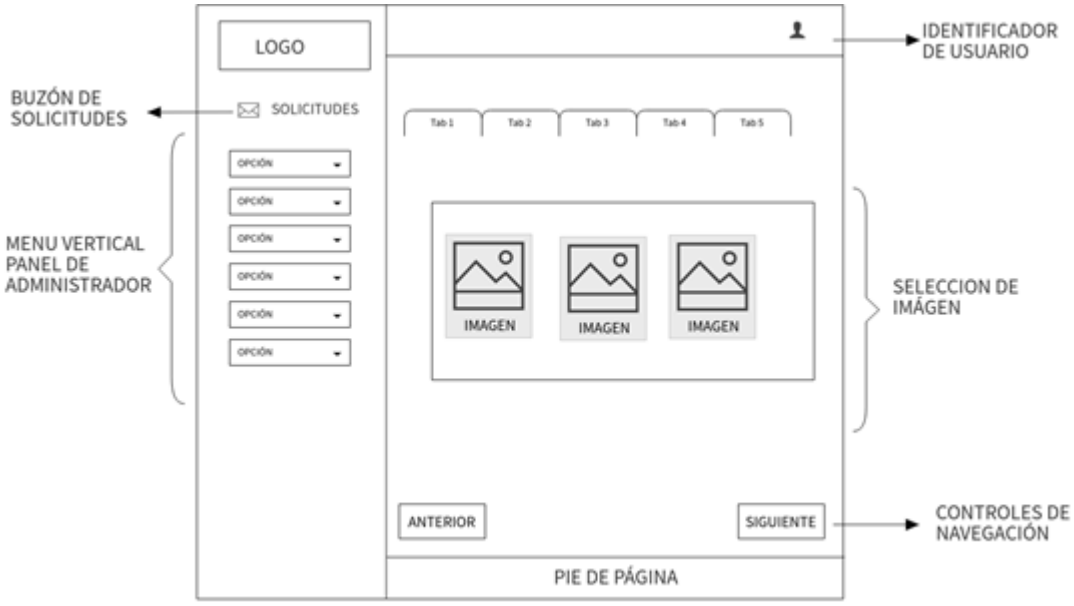


Figura K. 7.Plantilla de interfaz 4: Modificar Evento por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador de servicio selecciona la opción Modificar Evento y el sistema responde mostrando el formulario correspondiente.

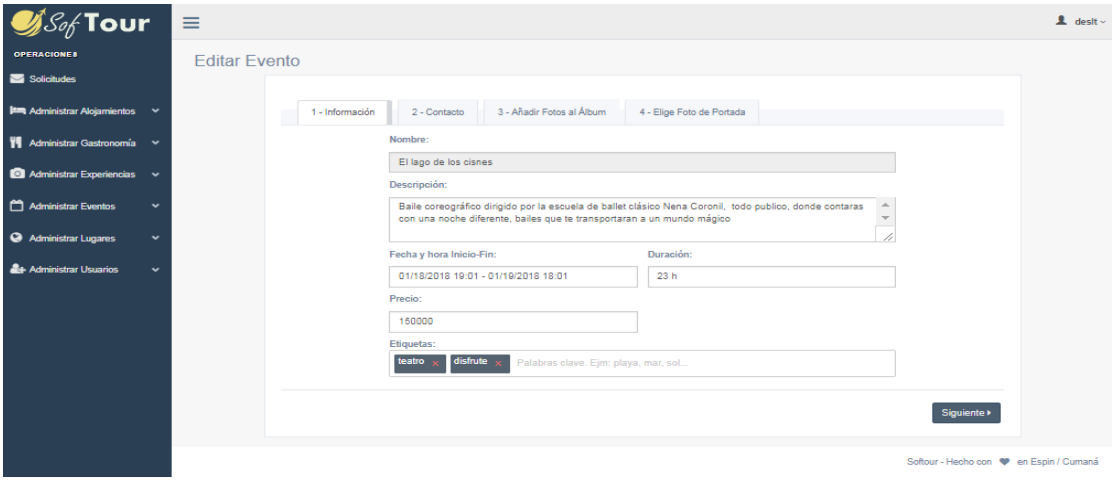


Figura K. 8.Interfaz 1 de proceso de Modificar Evento por usuario Administrador

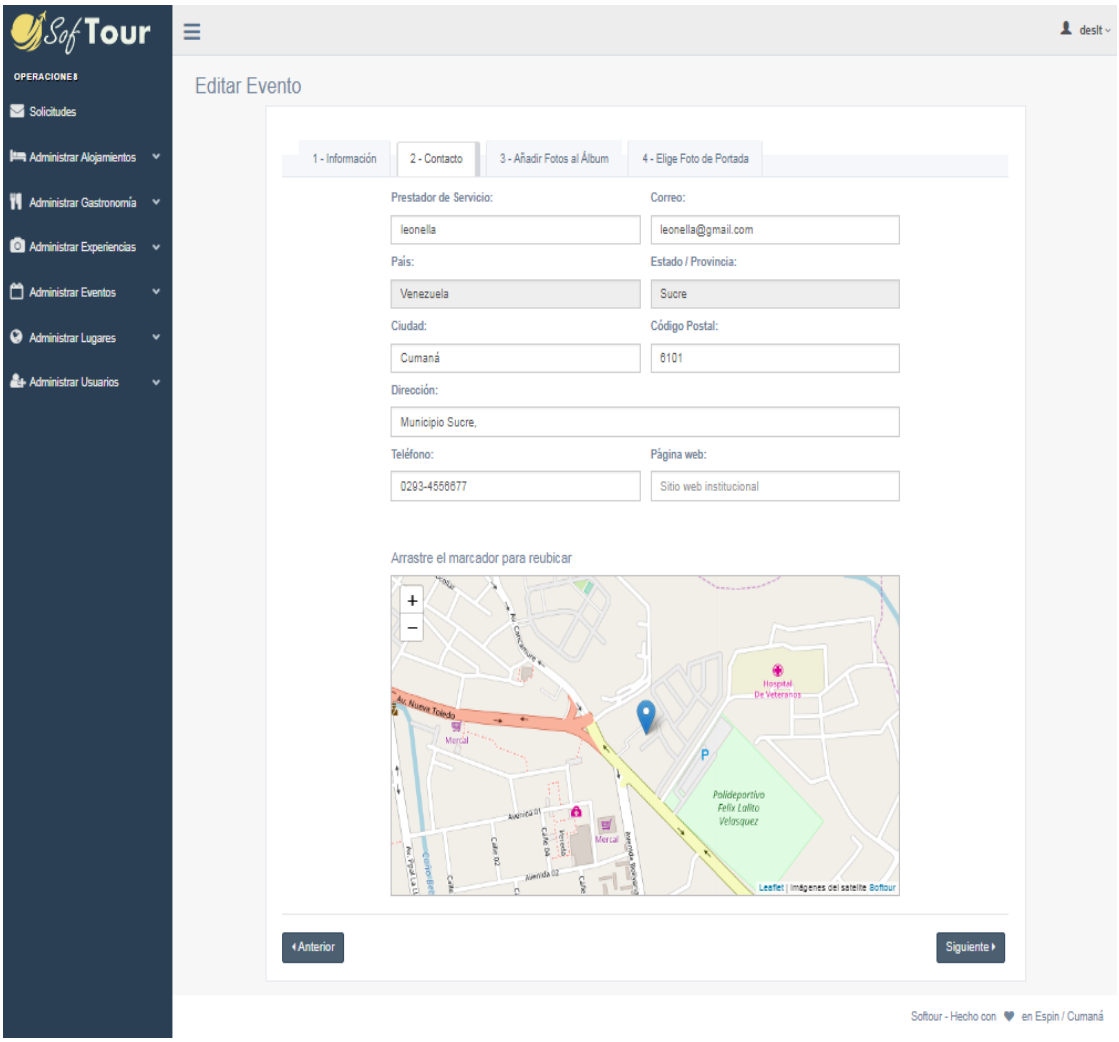


Figura K. 9.Interfaz 2 de proceso de Modificar Evento por usuario Administrador

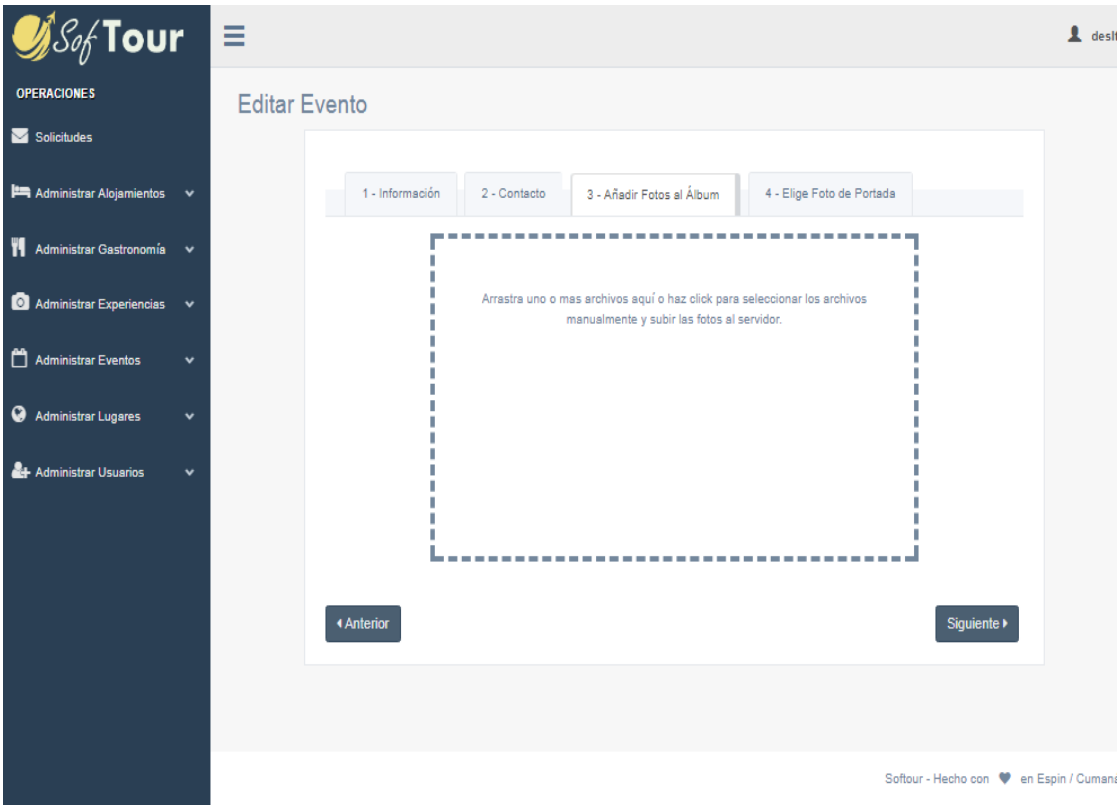


Figura K. 10.Interfaz 3 de proceso de Modificar Evento por usuario Administrador.

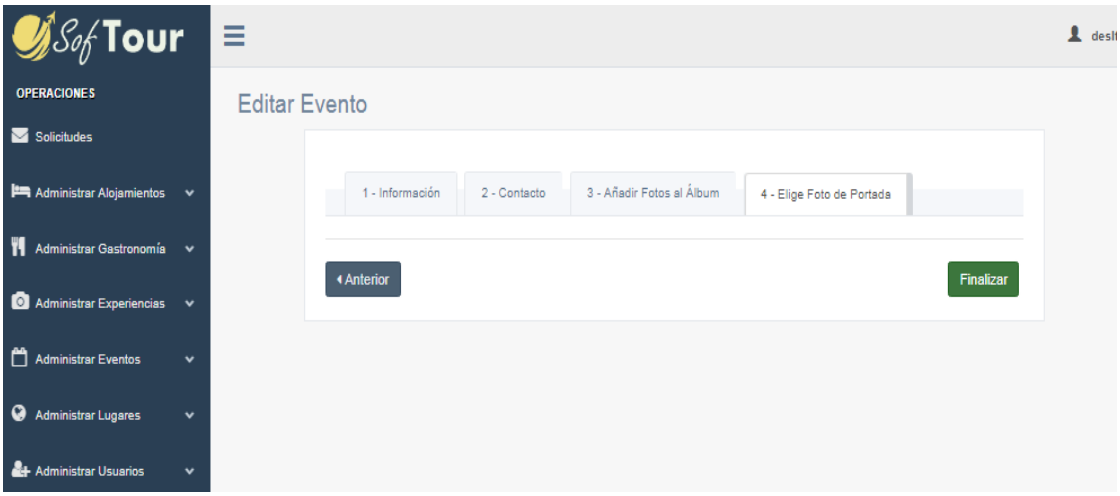


Figura K. 11. Interfaz 4 de proceso de Modificar Evento por usuario Administrador.

Código desarrollado para la funcionalidad Modificar Alojamiento por usuario administrador

```
modificar(req, res, next) {
    this._validar.campos(req.body, true, (error, resultados) => {
        if(error) return next(error);
        res.locals.datos = resultados;

        if(res.locals.datos.error){
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_modificar(req, res);
        }

        else {
            this._Evento.buscar({"url": req.params.url.toLowerCase()}, (error, buscado) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
                    this._respuesta.para_modificar(req, res);
                }

                if(!buscado) return next(new this._error_NoEncontrado());
                res.locals.datos.value.geo_ubicacion = {
                    "type": "Point",
                    "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
                };
                res.locals.datos.value.ubicacion = {
                    "direccion": res.locals.datos.value.direccion,
                    "pais": res.locals.datos.value.pais,
                    "estado": res.locals.datos.value.estado,
                    "ciudad": res.locals.datos.value.ciudad,
                    "codigo_postal": res.locals.datos.value.codigo_postal
                };

                res.locals.datos.value.url = req.params.url.toLowerCase();
                this._Evento.modificar(res.locals.datos.value, (error, modificado) => {
                    if(error) {
                        res.locals.datos.error = error;
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_modificar(req, res);
                    }
                    if(!modificado) return next(new this._error_NoEncontrado());

                    res.locals.datos = {"value": modificado};
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_modificar(req, res);
                });
            });
        }
    });
}
```

Figura K. 12.Código desarrollado para el proceso Modificar Evento por usuario Administrador.

Prueba editar evento

Tomando en cuenta los valores agregados en la prueba realizada a la función Agregar Evento se procede a realizar la técnica de caja negra, en donde se les cambiaran valores a los eventos ingresados y se procederá a documentar los resultados obtenidos.

Se cambiaron los siguientes valores durante la prueba para validar la funcionalidad del módulo:

Fecha: 02/02/18.

Descripción: Play Eventos te ofrece para tu boda, fiesta de pueblo, barrio, eventos deportivos todo lo que necesites, mira todo lo que te puede solucionar en un momento: Disco móvil, Disco Caravana, Escenario Movil o camión discomovil, proyecciones en pantalla, maestro ceremonia, disco karaoke, Barman/coctelero, Fuentes de ,

Duración: 12 horas.

Ciudad: Cumanacoa.

Direccion: Calle las Flores

Para llevar a cabo esta prueba se seleccionó un evento previamente cargado en sistema y se procedió a modificar cierta información, como se muestra en la siguiente secuencia de imágenes:

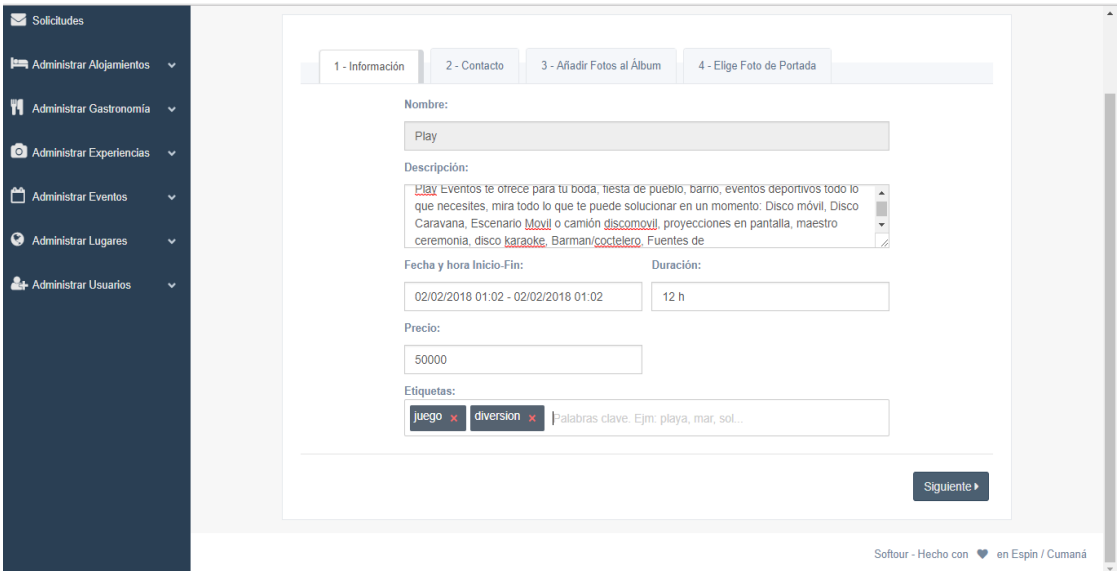


Figura K. 13.Modificación de información de un evento por Administrador. Paso 1

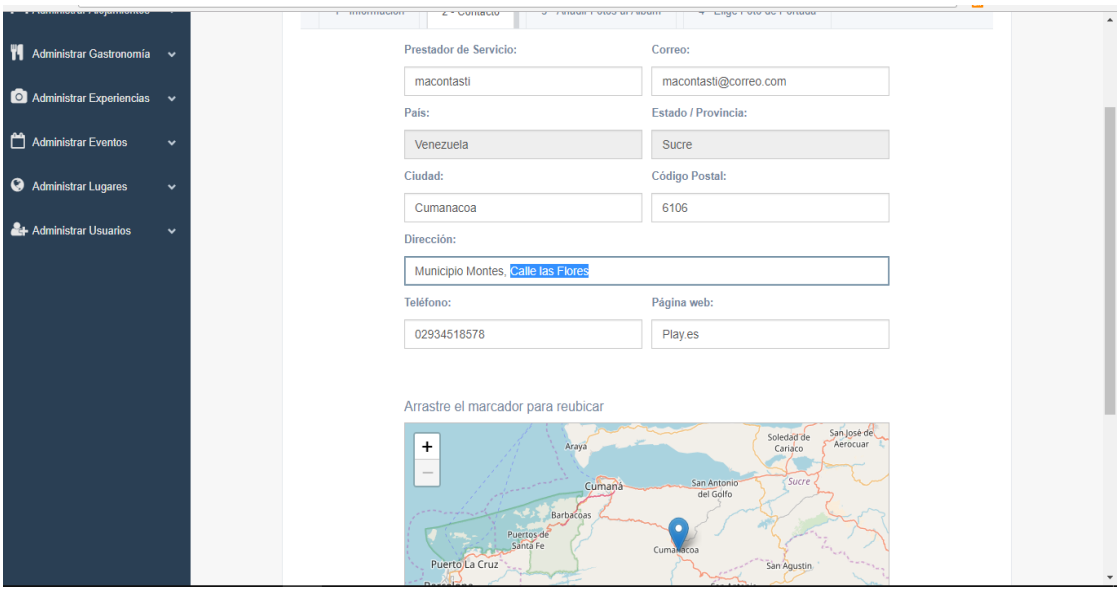


Figura K. 14.Modificación de información de un evento por administrador. Paso 2

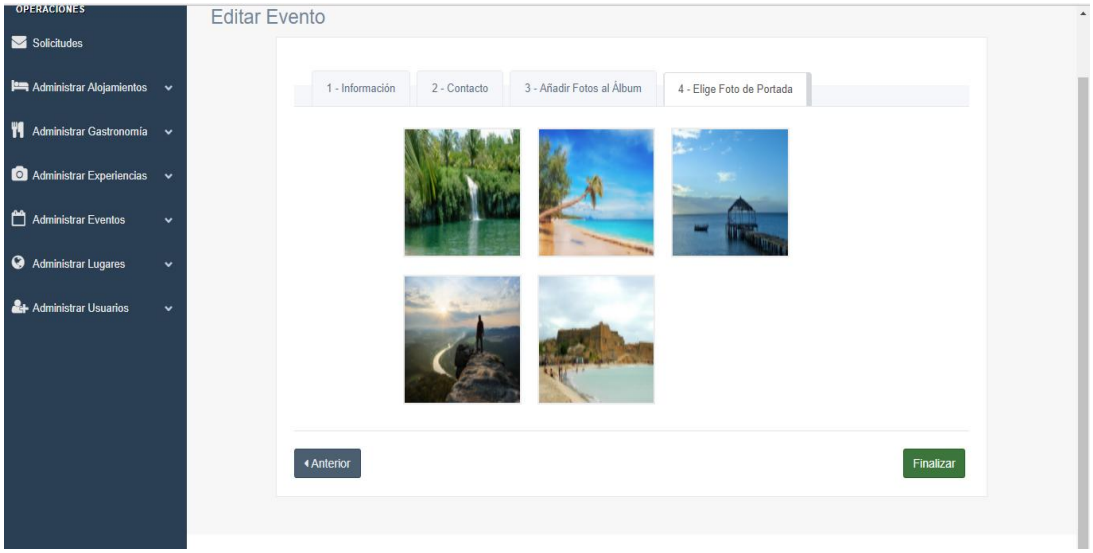


Figura K. 15.Modificación de información de un evento por Administrador. Paso 3

Conclusión de la prueba

La función cumple con el objetivo para la que fue desarrollada.

Prueba de interfaz y usabilidad

Esta interfaz se encuentra con los mismos estándares aplicados en interfaces evaluadas anteriormente por el método de análisis de heurísticas de usabilidad, en los cuales se validan la existencia de la mayoría de los principios de usabilidad.

Apéndice L. Ciclo de Versión Administrador: Iteración Eliminar Evento.

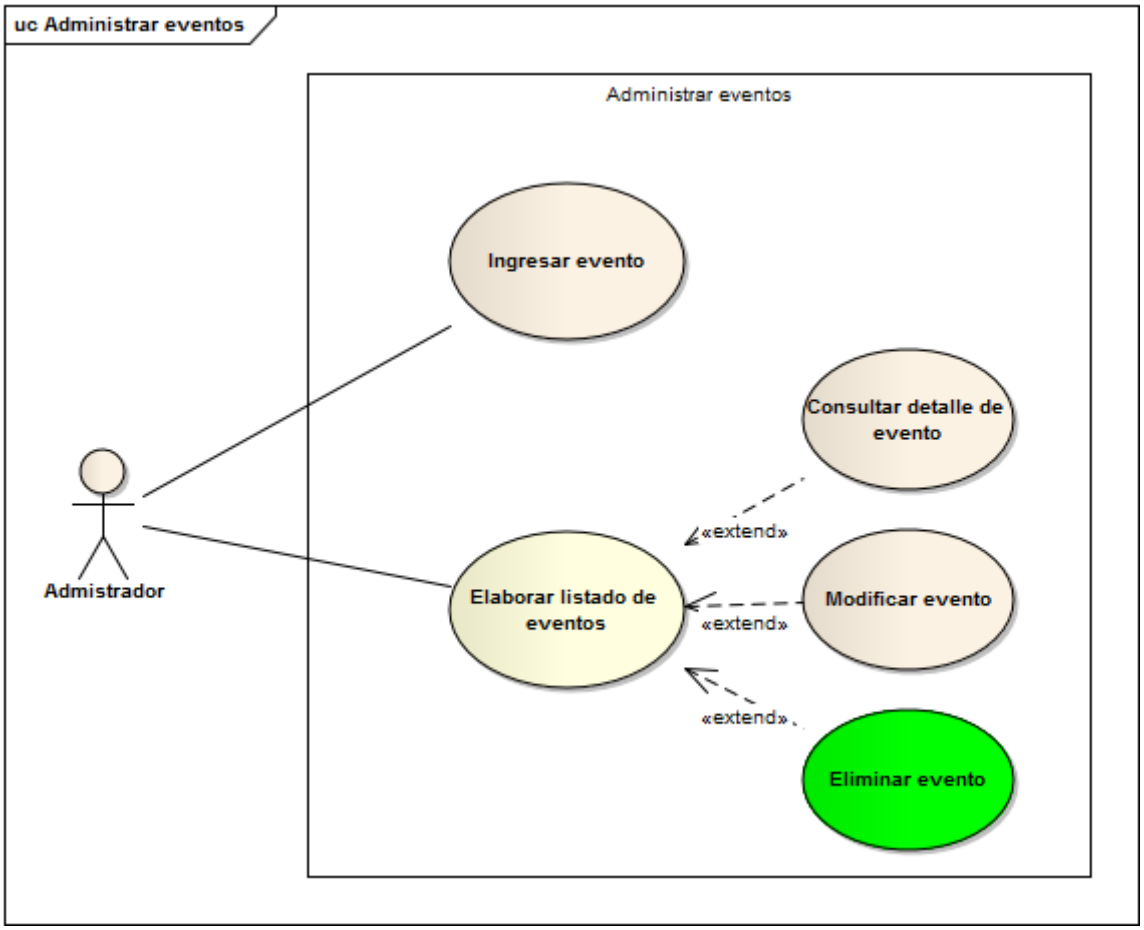


Figura L. 1. Caso de uso: Eliminar Evento por Administrador.

Tabla L. 1. Descripción Caso de uso Eliminar Evento por usuario Administrador.

Nombre del Caso de uso	Eliminar evento.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en eliminar un evento de manera satisfactoria.	
Precondición	Para eliminar un evento, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Eliminar”.
	2	El sistema ejecuta la búsqueda del evento.
	3	El Administrador visualiza el evento que desea eliminar y selecciona aceptar.
	4	El sistema muestra un mensaje “¿Está seguro de eliminar la cuenta del evento seleccionado?”
	5	El administrador selecciona la opción “Si, estoy seguro”
	6	El sistema realiza la eliminación del evento en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se elimina el evento de la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	5.1.	El Administrador decide no continuar con el caso de uso y selecciona cancelar el caso de uso, seleccionando otra opción del sistema.
	5.2.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. El evento debe estar registrado para que pueda ser eliminado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

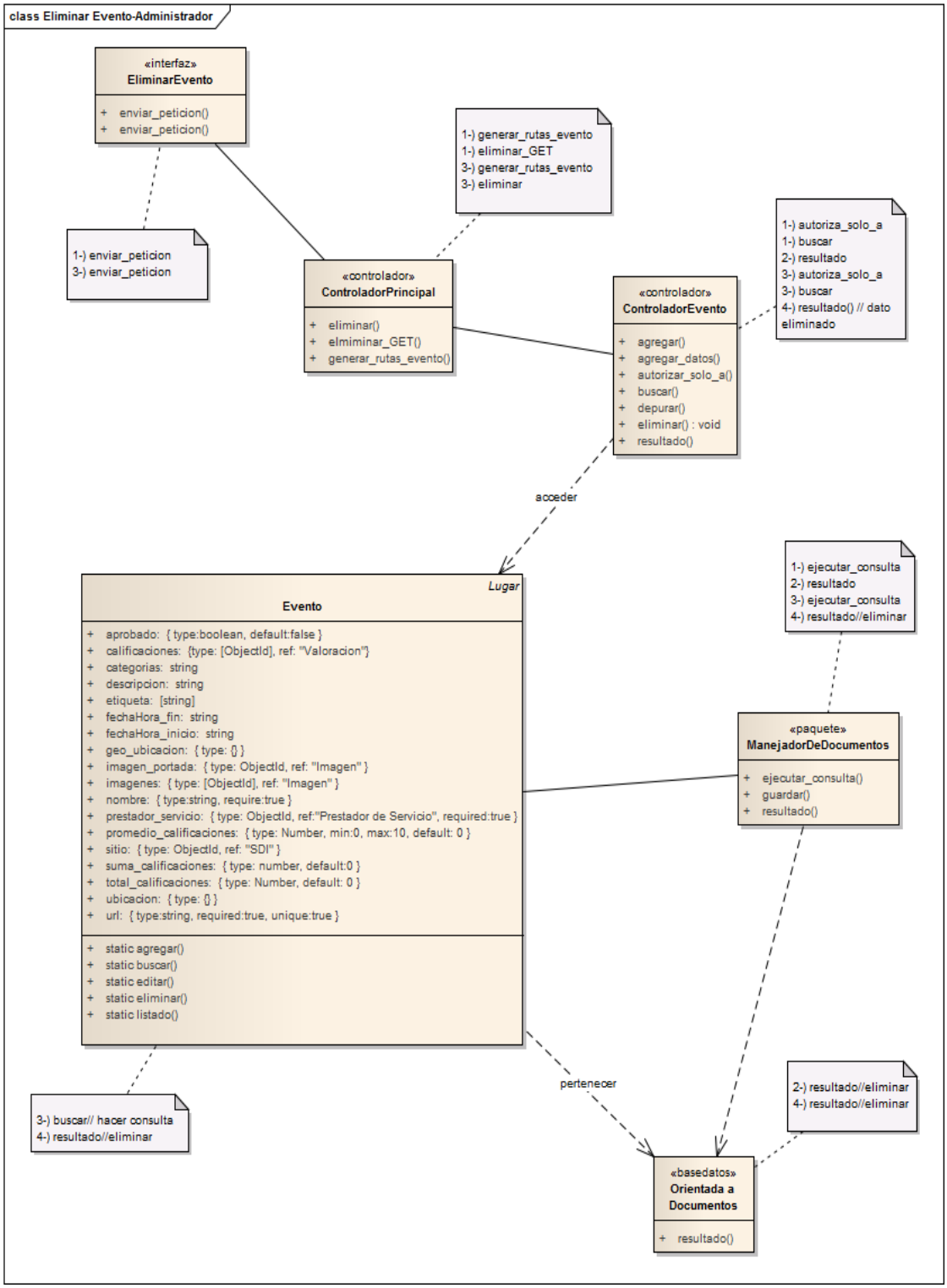


Figura L. 2. Diagrama de Clases: Eliminar Evento por usuario Administrador.

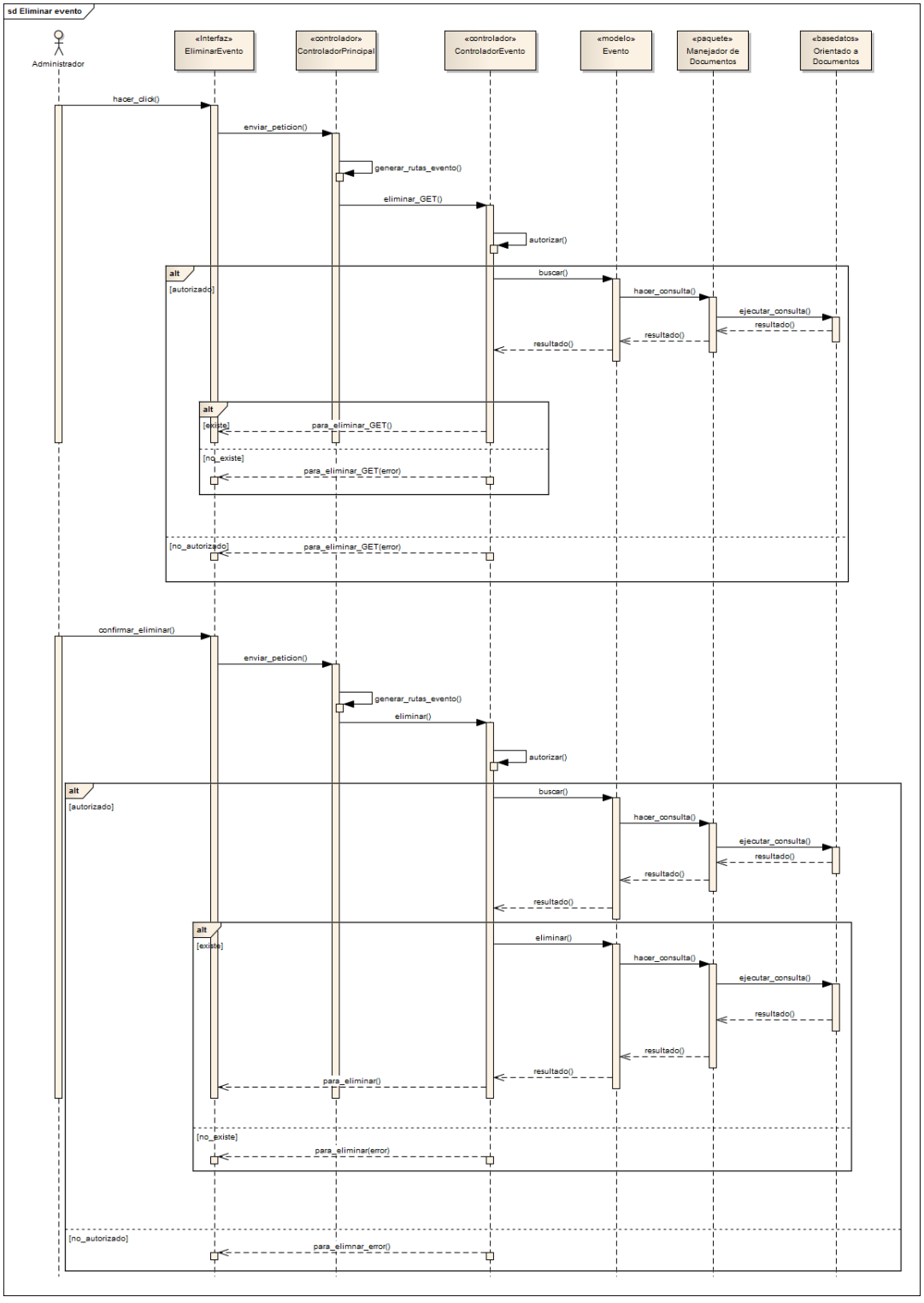


Figura L. 3. Diagrama de secuencia: Eliminar Evento por usuario Administrador.

A continuación, se muestra el modelo de plantilla para el proceso de eliminar evento.

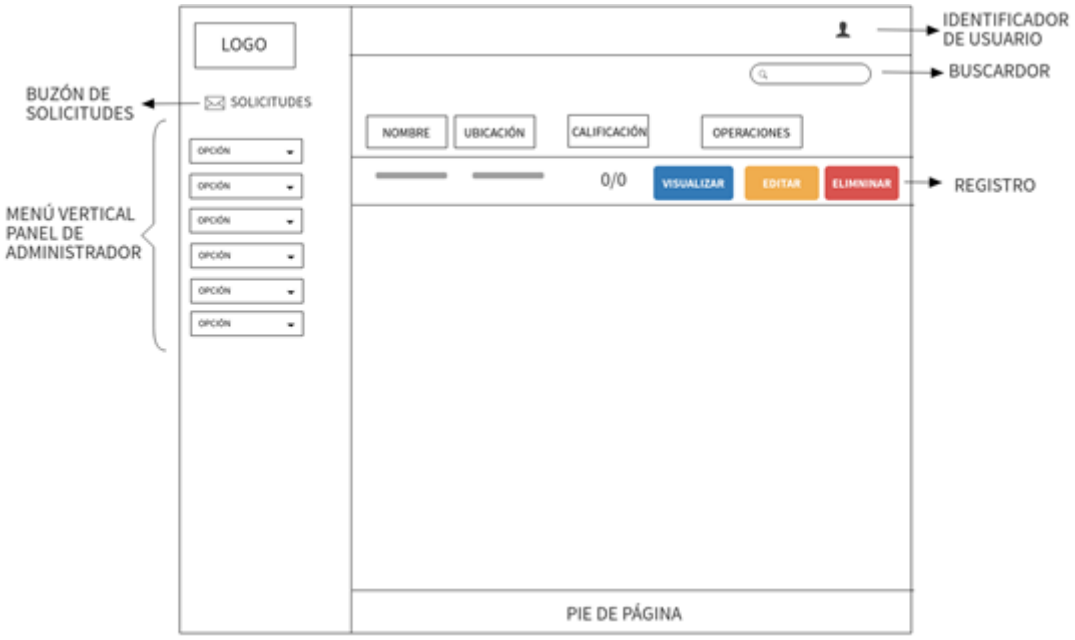


Figura L. 4. Plantilla de interfaz 1: Eliminar Evento por usuario Administrador.

Figura L. 5. Plantilla de interfaz 2: Eliminar Evento por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Eliminar Evento y el sistema responde

mostrando la interfaz de confirmación de acción

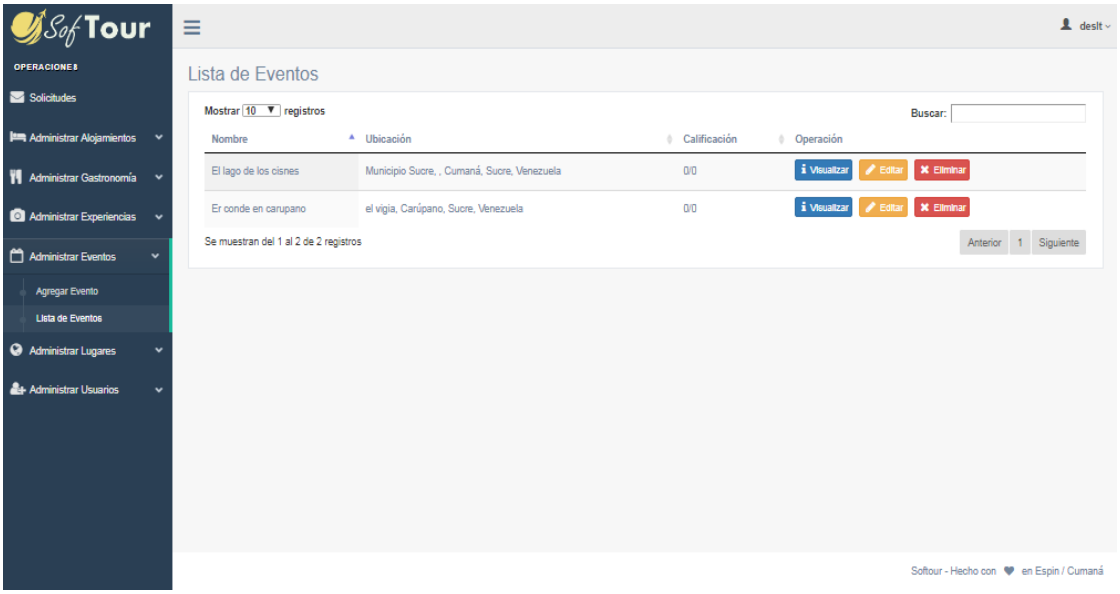


Figura L. 6.Interfaz 1 de proceso de Eliminar Evento por usuario Administrador.

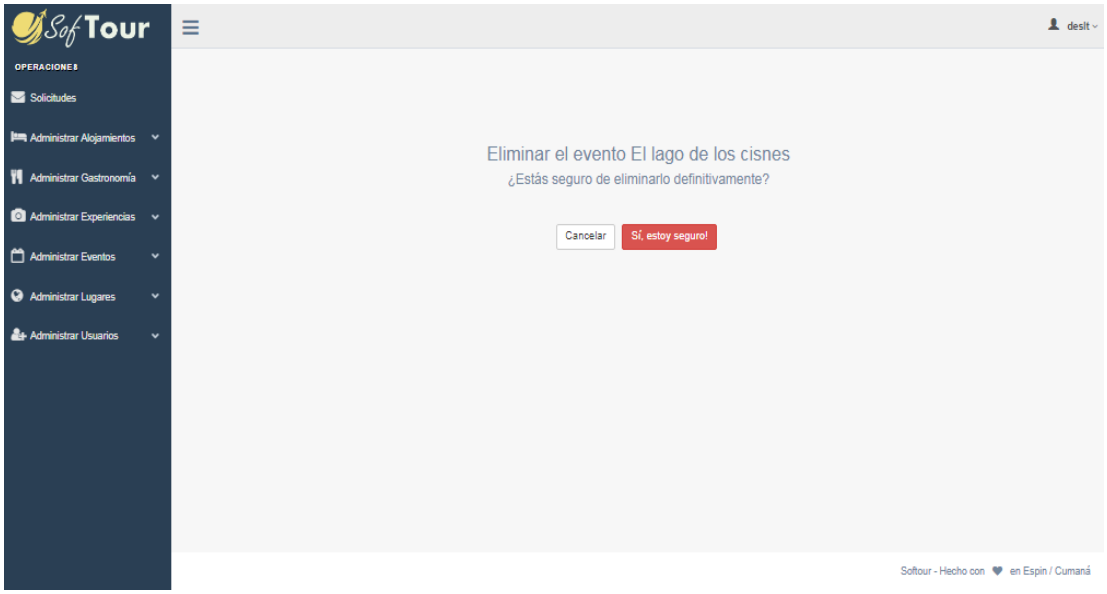


Figura L. 7. Interfaz 2 de proceso de Eliminar Evento por usuario Administrador

Parte del código desarrollado para la funcionalidad Eliminar Evento por usuario Administrador

```
eliminar(req, res, next) {
    this._Evento.eliminar({"url": req.params.url.toLowerCase()}, (error, eliminado) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_eliminar(req, res);
        }
        if(!eliminado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": {"url": "listado"}};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_eliminar(req, res);
    });
}
```

Figura L. 8. Código desarrollado para el proceso Eliminar Evento por usuario Administrador.

Prueba Eliminar Evento

Para llevar a cabo esta prueba se necesitan tener en la base de datos eventos registrados, para lo cual se utilizaron los eventos ingresados en la prueba realizada a la función Ingresar evento.

La forma de realizar la prueba consiste en ubicarse en el listado de eventos en sistema y proceder a utilizar el botón de eliminar, sin el evento se elimina satisfactoriamente la función cumple su finalidad.

En la siguiente secuencia de imágenes se muestra una de las pruebas realizadas a esta función:

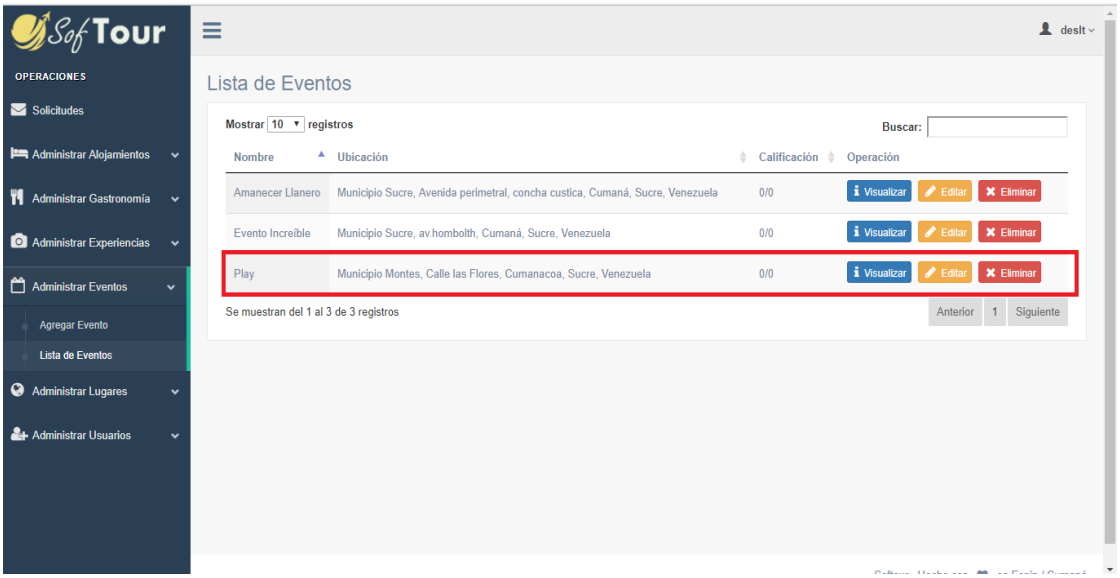


Figura L. 9.Imagen Litado de Eventos.

Se procede a eliminar el evento denominado Play:

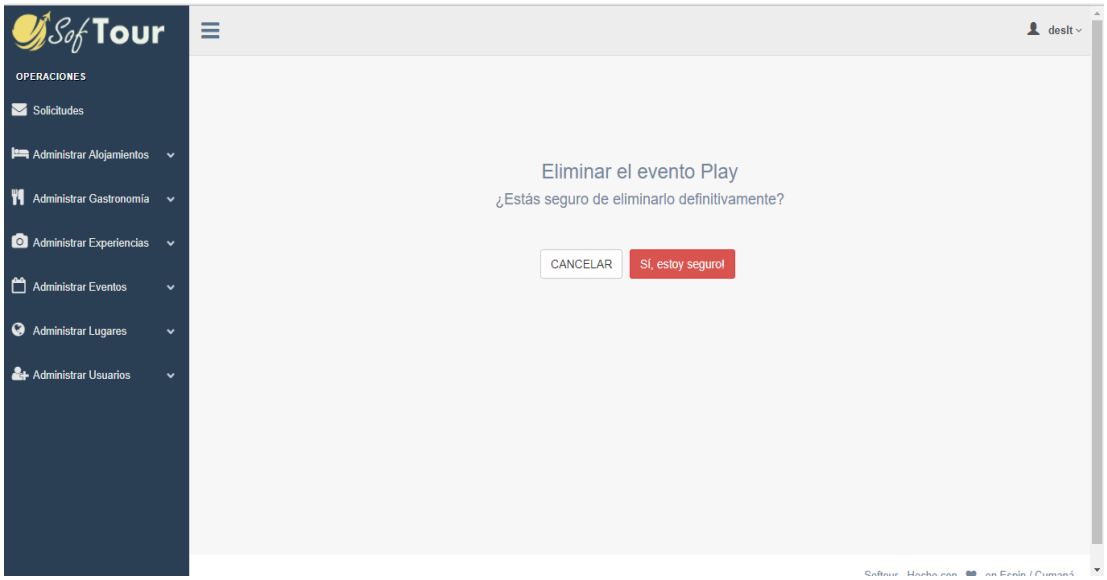


Figura L. 10. Imagen el sistema confirma que desea realizar la función.

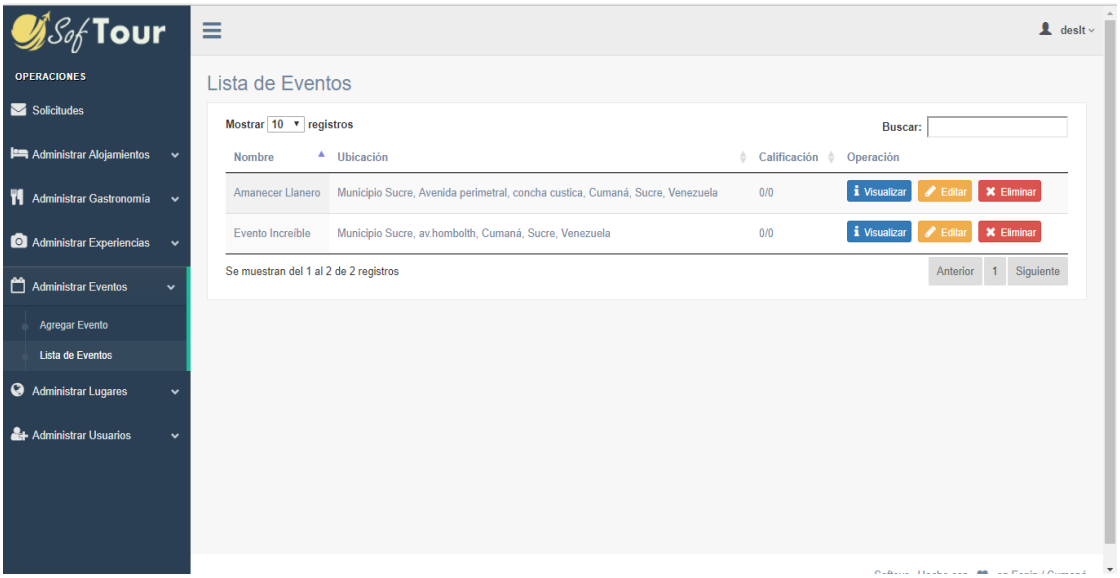


Figura L. 11.Imagen el sistema muestra el Listado de Eventos modificado.

Conclusión de la prueba

El objetivo para el cual fue desarrollado la prueba se cumplió.

Prueba de interfaz y usabilidad

Basado en pruebas realizadas con anterioridad se concluye que esta interfaz por pertenecer al conjunto de interfaz de usuario Administrador, cumple con buenas prácticas de diseño de interfaz y de usabilidad.

Apéndice M. Ciclo de Versión Administrador: Iteración Ingresar Experiencia.

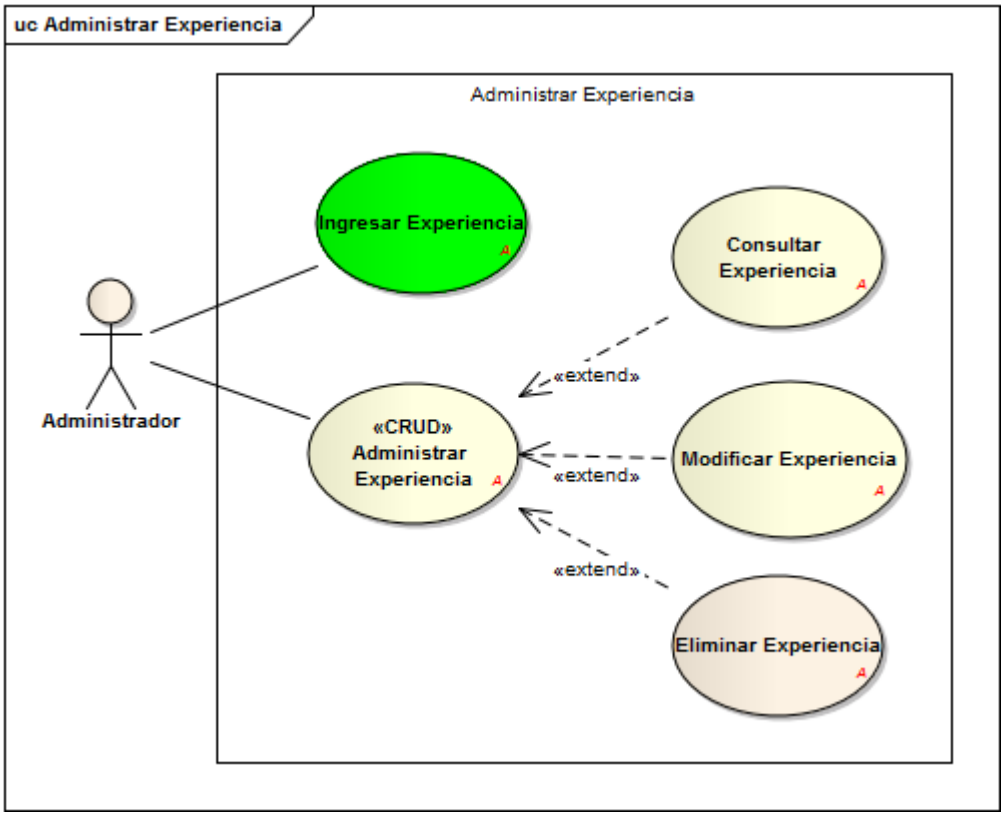


Figura M. 1.Caso de uso: Ingresar Experiencia por usuario Administrador.

Tabla M. 1. Descripción Caso de uso Ingresar Alojamiento por usuario Administrador.

Nombre del Caso de uso	Ingresar experiencia.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en ingresar los atributos de la experiencia de manera satisfactoria.	
Precondición	Para ingresar una experiencia, no debe estar registrada en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Agregar Experiencia".
	2	El sistema muestra el formulario correspondiente.
	3	El Administrador ingresa los atributos de la experiencia.
	4	El sistema almacena los atributos de la experiencia y finaliza el caso de uso.
Postcondición	Se ingresa los atributos de la experiencia en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no ingresar la experiencia y selecciona cancelar el caso de uso, seleccionando otra opción en el sistema.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. Para que una experiencia pueda ser ingresada, no debe estar registrada en la base de datos del sistema.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

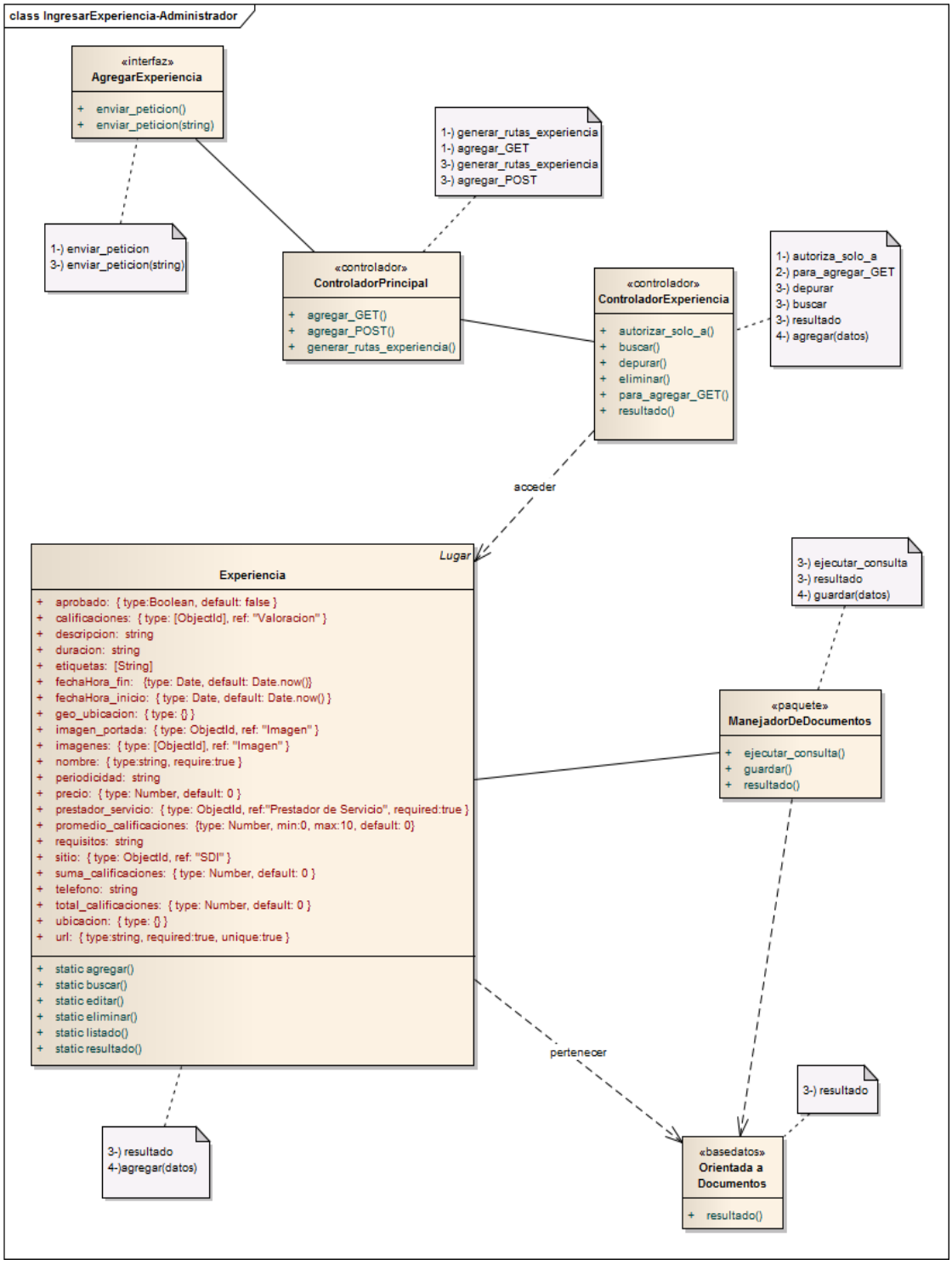


Figura M. 2.Diagrama de Clases: Ingresar Experiencia por usuario Administrador

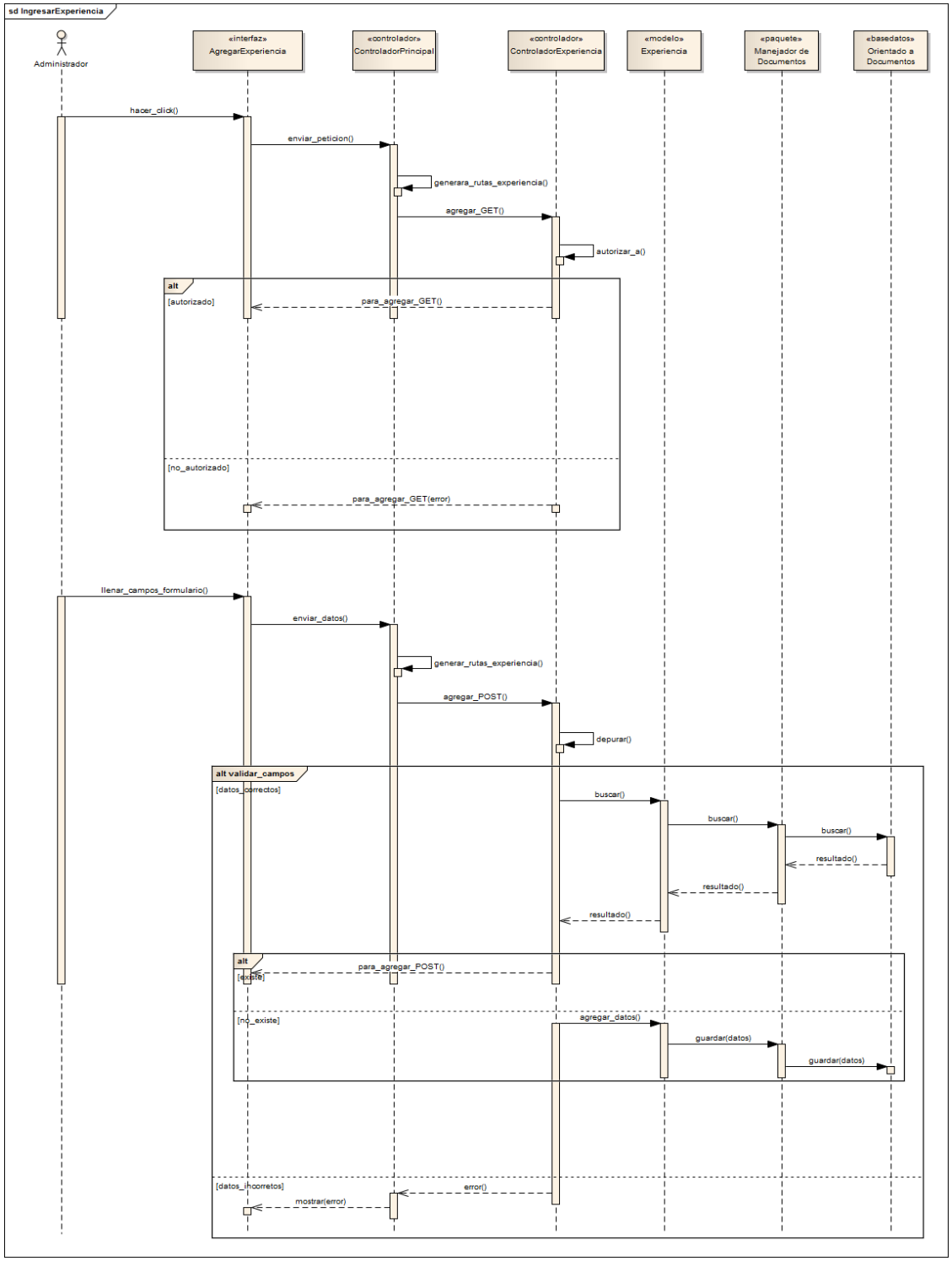


Figura M. 3.Diagrama de secuencia: Ingresar Experiencia por usuario Administrador

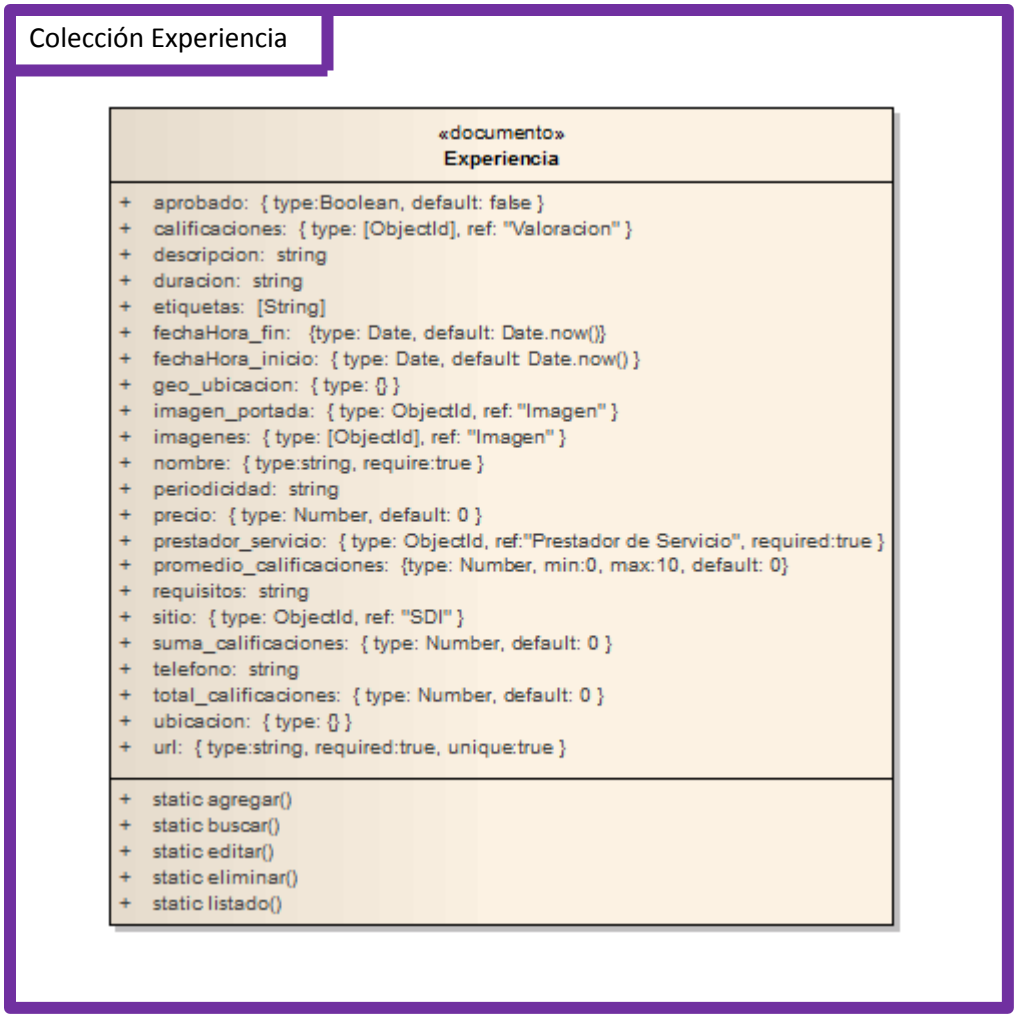


Figura M. 4. Colección Experiencia: Ingresar Experiencia por usuario Administrador

A continuación, se muestran los modelos de plantillas para el proceso de Ingresar Experiencia.

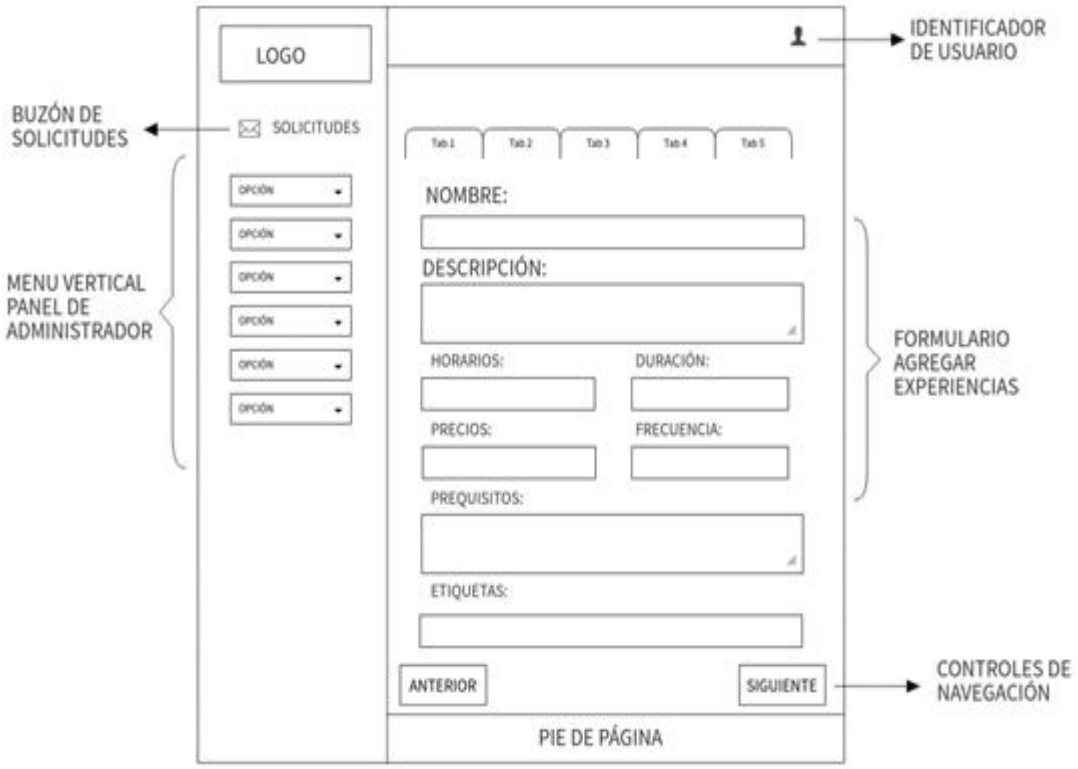


Figura M. 5. Plantilla de interfaz 1: Ingresar Experiencia por usuario Administrador.

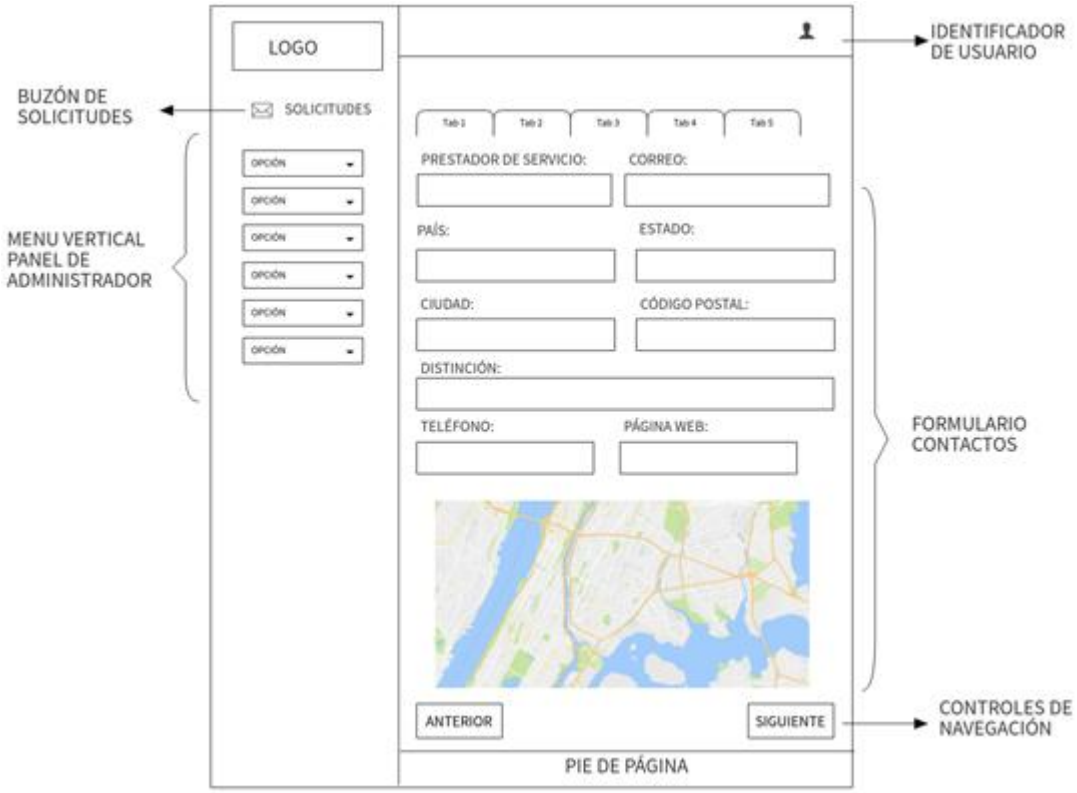


Figura M. 6. Plantilla de interfaz 2: Ingresar Experiencia por usuario Administrador.

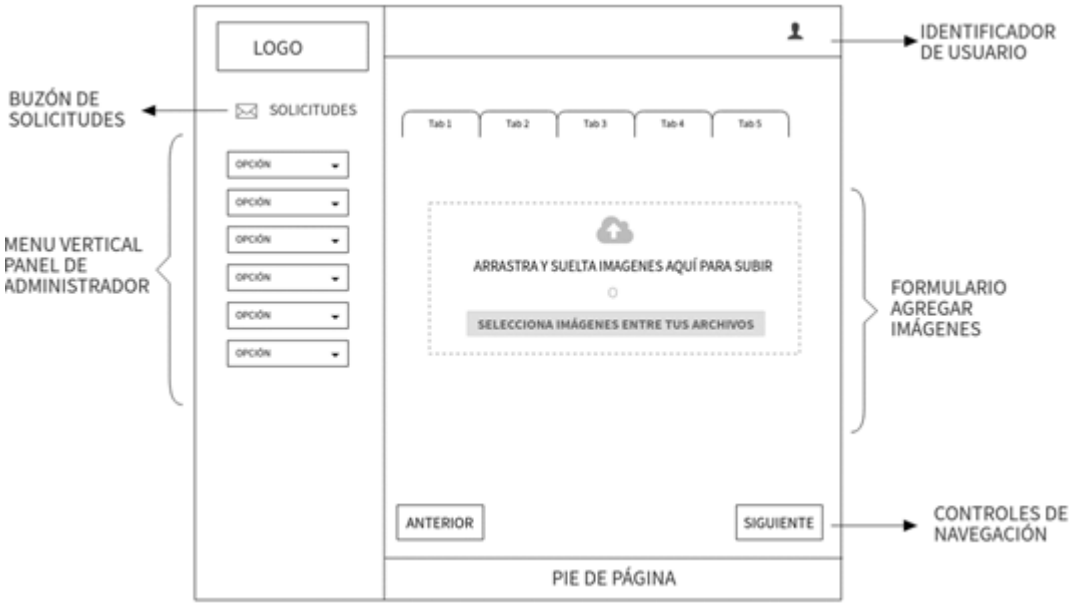


Figura M. 7.Plantilla de interfaz 3: Ingresar Experiencia por usuario Administrador.

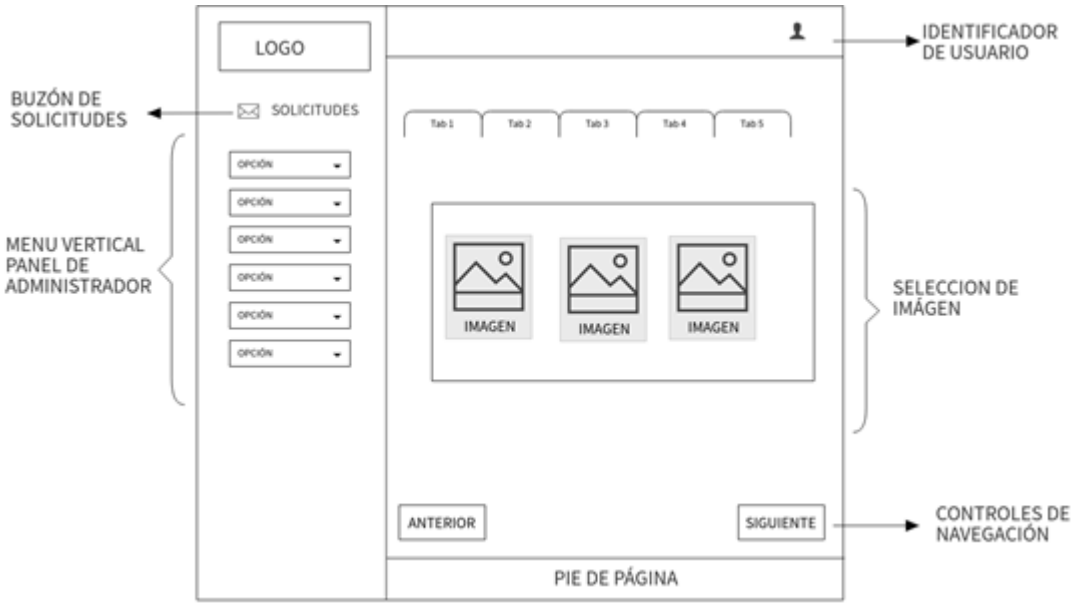


Figura M. 8. Plantilla de interfaz 4: Ingresar Experiencia por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Agregar Experiencia y el sistema responde mostrando el formulario correspondiente.

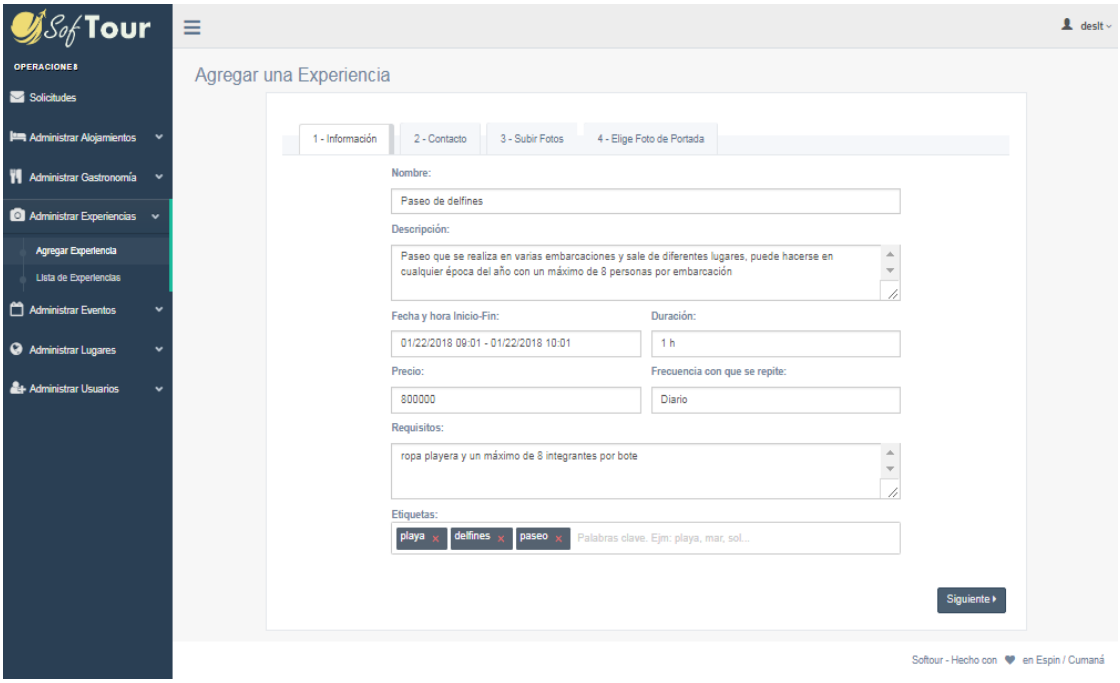


Figura M. 9.Interfaz 1 de proceso de Agregar Experiencia por usuario Administrador

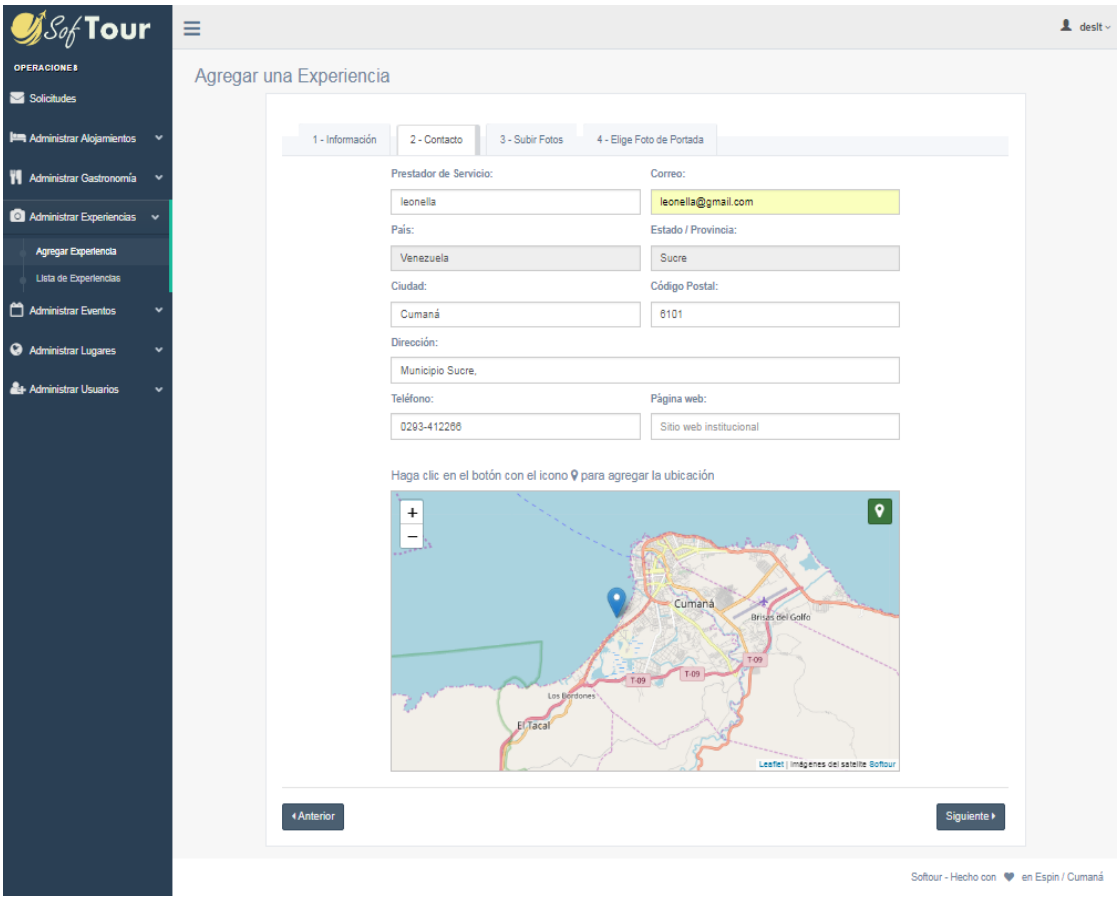


Figura M. 10.Interfaz 2 de proceso de Agregar Experiencia por usuario Administrador

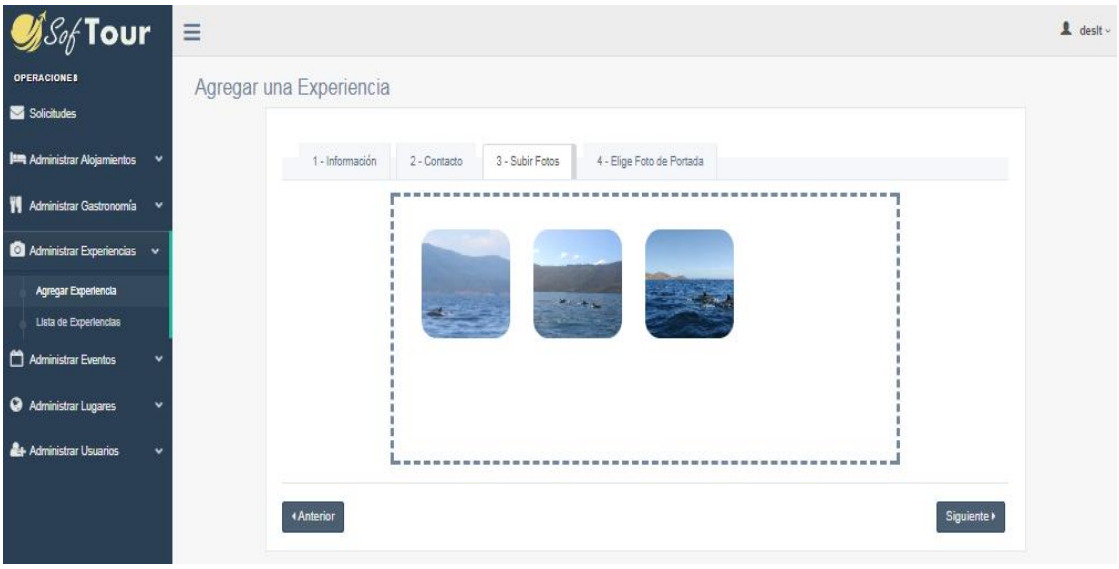


Figura M. 11.Interfaz 3 de proceso de Agregar Experiencia por usuario Administrador

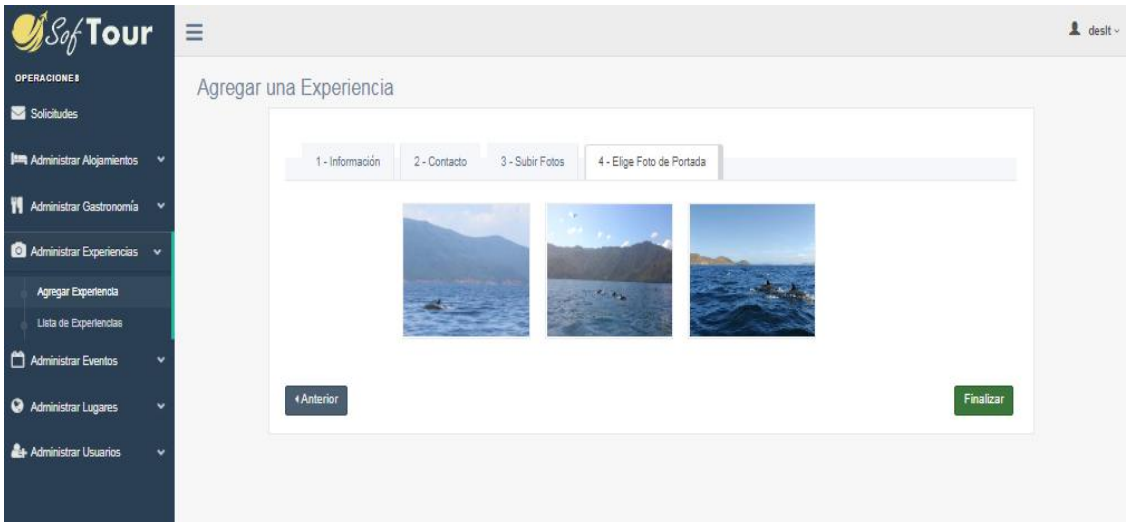


Figura M. 12.Interfaz 4 de proceso de Agregar Experiencia por usuario Administrador

Código desarrollado para la funcionalidad Agregar Experiencia por usuario

administrador

```
ingresar_POST(req, res, next) {
    this._validar.campos(req.body, false, (error, resultados) => {
        if(error) next(error);
        res.locals.datos = resultados;

        if (res.locals.datos.error) {
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;

            this._respuesta.para_ingresar_POST(req, res);
        }

        else {
            res.locals.datos.value.geo_ubicacion = {
                "type": "Point",
                "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
            };
            res.locals.datos.value.ubicacion = {
                "direccion": res.locals.datos.value.direccion,
                "pais": res.locals.datos.value.pais,
                "estado": res.locals.datos.value.estado,
                "ciudad": res.locals.datos.value.ciudad,
                "codigo_postal": res.locals.datos.value.codigo_postal
            };

            this._Experiencia.buscar({"url": res.locals.datos.value.url}, (error, experiencia) => {
                if (experiencia) {
                    res.locals.datos.error = { "details": [ {
                        "path": "nombre",
                        "message": 'Ya existe una experiencia con este nombre.'
                    } ] };
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_ingresar_POST(req, res);
                }

                else{
                    if(req.session.passport.user.tipo_usr == "Administrador")
                        res.locals.datos.value.aprobado = true;

                    this._Experiencia.ingresar(res.locals.datos.value, (error, creado) => {
                        if(error) {
                            res.locals.datos.error = error;

                            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                            this._respuesta.para_ingresar_POST(req, res);
                        }

                        res.locals.datos = {"value": creado};
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_ingresar_POST(req, res);
                    });
                }
            });
        }
    });
}
```

Figura M. 13. Código desarrollado para el proceso Ingresar Experiencia por usuario Administrador.

Prueba agregar experiencia

Mediante esta función el usuario Administrador puede llenar la base de datos con experiencias de determinado lugar que realice un prestador de servicio y de esta manera brindar la publicidad en el sistema.

Para corroborar que este módulo funciona de manera óptima se le aplico la técnica de caja negra basándose en los siguientes datos y casos de prueba:

Tabla M. 2. Clases de equivalencia Agregar Experiencia por Administrador

Condición de entrada	Tipo	Clases de equivalencias válidas	Clases de equivalencias no válidas
Nombre	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Descripción	Alfanumérico	1: Cadena de caracteres, números.	
Fecha y hora inicio-fin	Selección		2: Campo en blanco
Duración	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
precio	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Frecuencia con que se repite	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Requisitos	Alfanumérico	1: Cadena de caracteres, números.	
Etiquetas	Alfanumérico	1: Cadena de caracteres, números.	
Prestador de servicio	Selección		2: Campo en blanco
Correo	Correo	3: símbolos, caracteres, números	2: Campo en blanco 4: formato

			invalido
País	Alfabético	1: Cadena de caracteres	5: Símbolos 6: números
Estado/provincia	Alfabético	1: Cadena de caracteres	5: Símbolos 6: números
País	Alfabético	1: Cadena de caracteres	5: Símbolos 6: números
Ciudad	Alfabético	1: Cadena de caracteres	5: Símbolos 6: números
Código postal	Numérico	6: números	1: Cadena de caracteres
Dirección	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Teléfono	Numérico	6: números	1: Cadena de caracteres
Pagina web	Alfanumérico	1: Cadena de caracteres, números.	
Selección mapa	Selección		
Subir foto	Imagen	7: JPG, GIF, PNG	2: Campo en blanco
Seleccionar foto	selección		

Tabla M. 3. Casos de prueba Agregar Experiencia por Administrador.

Condición de entrada	A1	A2	A3
Nombre	Buceo con tiburones	Recorrido casco histórico	Parapente
Descripción	Es probable que hayas buceado con tiburones desde el principio. La verdad es que tendrás suerte de poder ver un tiburón bajo el agua. Representa un extraño privilegio el observarlos. De hecho, ¿sabías	En el recorrido con guía por el casco histórico de Cumaná, podrá enterarse al pasear por la colina del castillo y el casco urbano de Cumaná de muchos datos fascinantes de la historia y la actualidad de	

	que estadísticamente tienes más probabilidades de que te caiga un rayo que de ser mordido por un tiburón?	Cumaná.	
Fecha y hora inicio-fin	16-10/16-10	01-02/17-02	12-12/12-12
Duración	4 hora	2 horas	4 horas
precio	1000 Bs	500000	150000 Bs
Frecuencia con que se repite	Diario	Diario	Diario
Requisitos	Saber nadar.		No tener operaciones recientes
Etiquetas	Aguas; buceo	Caminar; conocer	Aire; vuelo; deporte
Prestador de servicio	selección	Selección	Selección
Correo	buceo@gmail.com	manuel@hotmail.com	volar@parapente.com
País	Venezuela	Venezuela	Venezuela
Estado/provincia	Sucre	Sucre	Sucre
Ciudad	Cumaná	Cumaná	Carúpano
Código postal	6101	hgfd	5555
Dirección	Playa san Luis	Calle bolívar #25	
Teléfono	029345584	0293455847	02947778854
Pagina web	Buceo.com		Parapente.es
Selección mapa	selección	Selección	selección
Subir foto	PNG, JPG		PDF
Seleccionar foto	Selección	Selección	selección
Clases Equivalencia	1,3,4,5,6,7	1,2,3,4,5	1,2,3,4
Resultado	Registro satisfactorio	Registro no satisfactorio por datos no válidos para el sistema.	Registro no satisfactorio.

En el caso de pruebas A1 para Ingresar Experiencia el registro fue satisfactorio,

se ingresaron datos que fueron acordes a los permitidos por el sistema y la función cumplió su fin. A continuación, se muestran en las Figuras del caso de prueba A1.

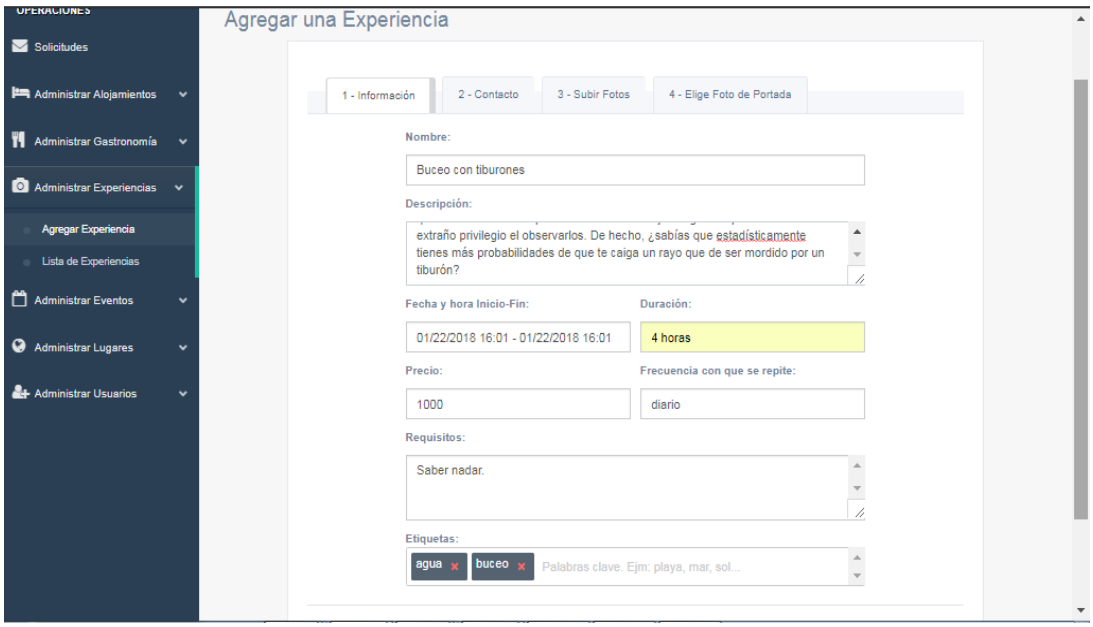


Figura M. 14.Caso de prueba A1 Ingresar Experiencia por Administrador. Paso 1

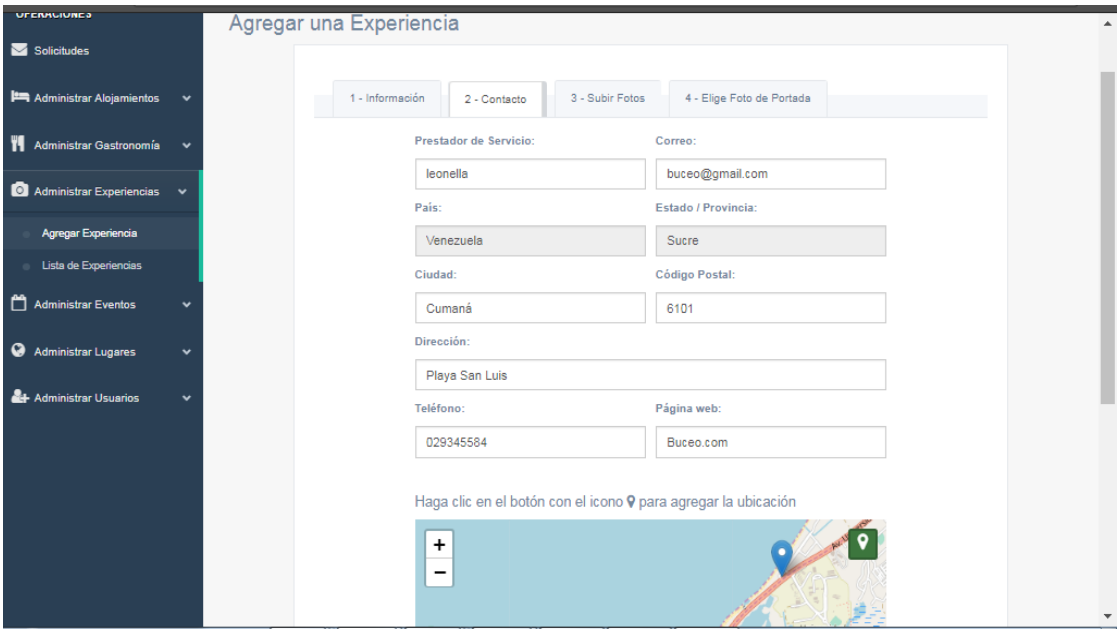


Figura M. 15. Caso de prueba A1 Ingresar Experiencia por Administrador. Paso 2

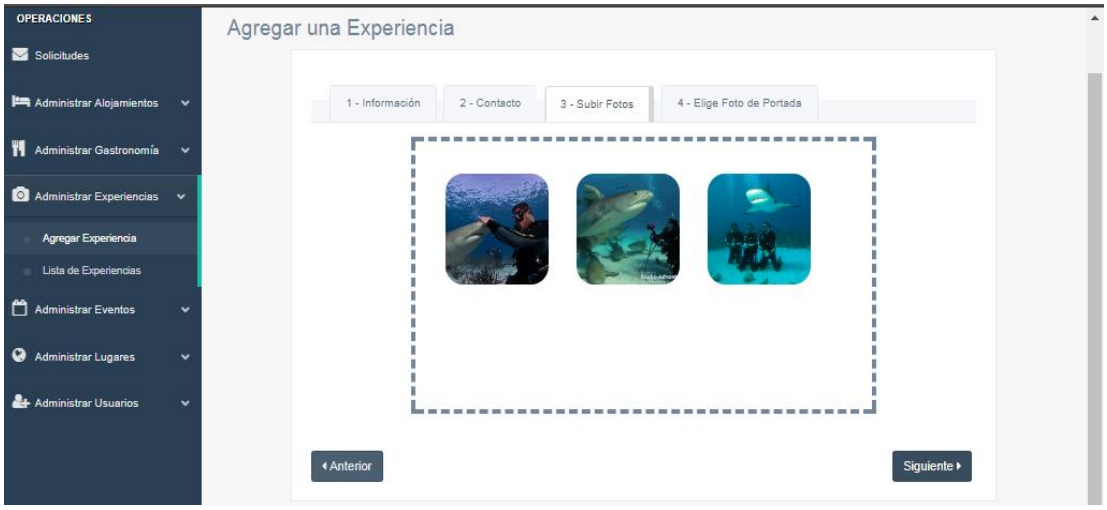


Figura M. 16. Caso de prueba A1 Ingresar Experiencia por Administrador. Paso 3

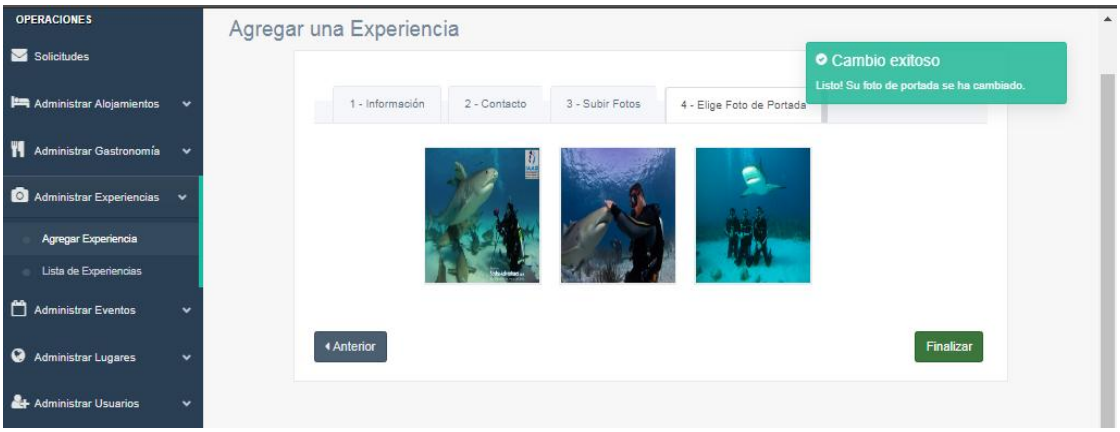


Figura M. 17.Caso de prueba A1 Ingresar Experiencia por Administrador. Paso 4

Para el caso de pruebas A2 el sistema permitió superar el paso 1 si ingresar requisitos, algo contemplado en el diseño de registro, mas no permitió continuar el registro por ingresar datos no validos en el código postal.

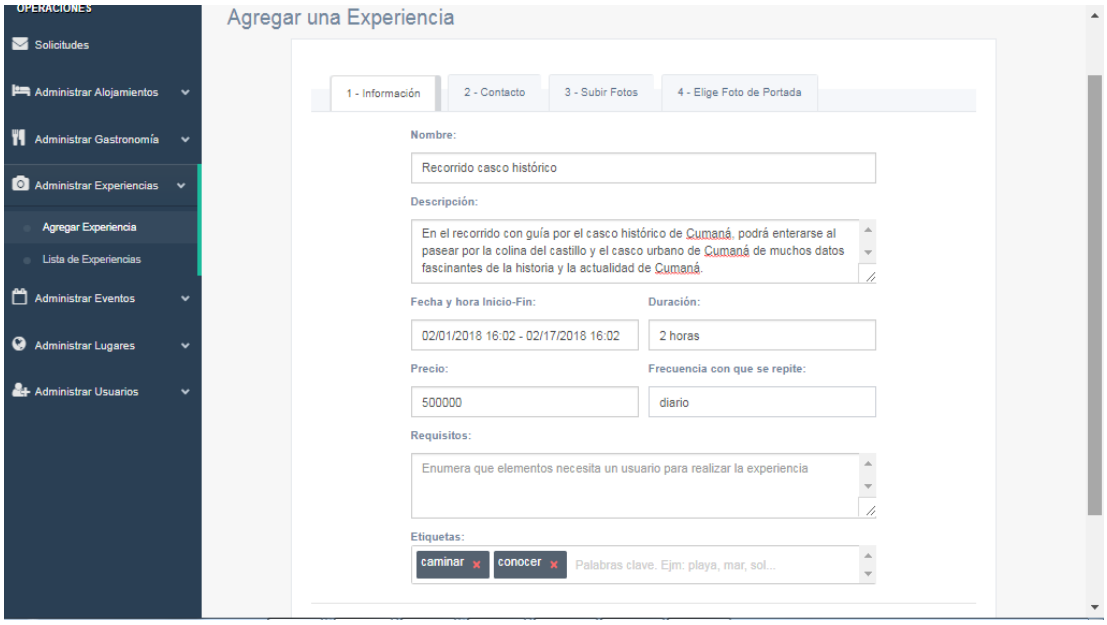


Figura M. 18.Caso de prueba A2 Ingresar Experiencia por Administrador. Paso 1

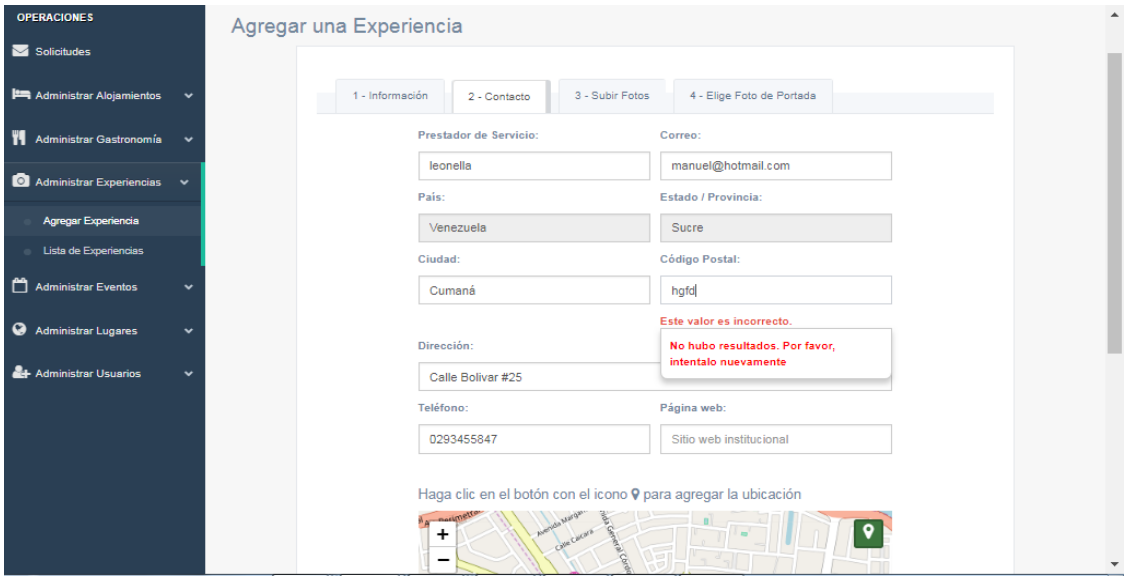


Figura M. 19.Caso de prueba A2 Ingresar Experiencia por Administrador. Paso 2

En el caso de pruebas A3 de Agregar Experiencias el sistema permitió continuar registro sin el usuario agregar una descripción de la Experiencia, característica permitida por el sistema, pero no permitió continuar por falta de una dirección.

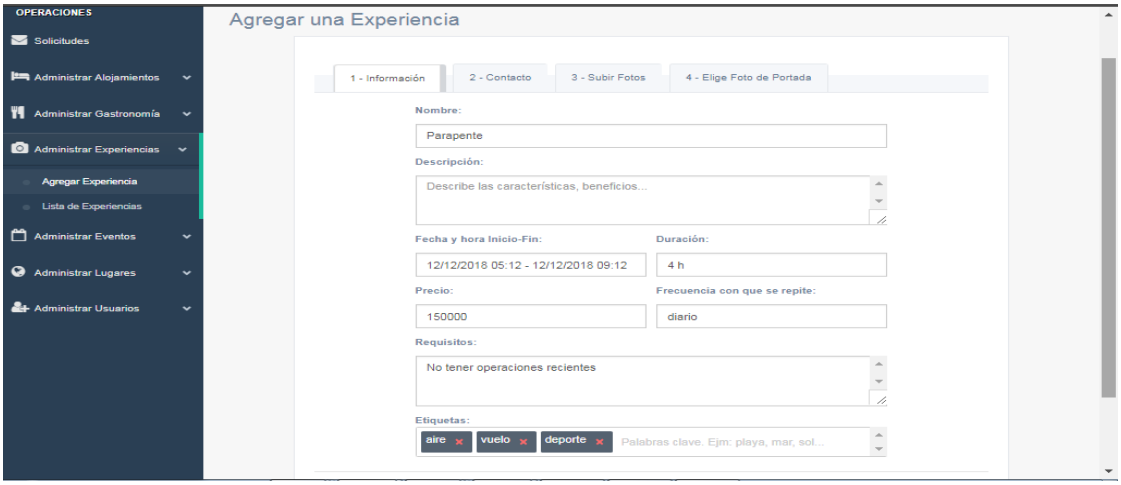


Figura M. 20.Caso de prueba A3 Ingresar Experiencia por Administrador. Paso 1

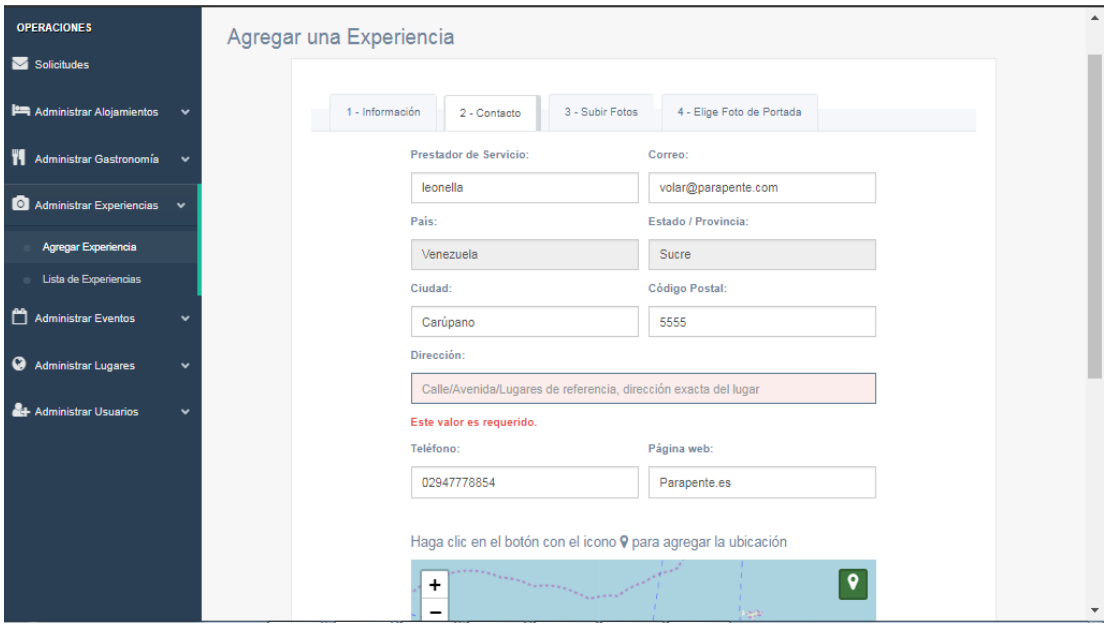


Figura M. 21. Caso de prueba A3 Ingresar Experiencia por Administrador. Paso 2

Conclusión de la prueba

La función cumple con el objetivo para el cual fue desarrollada, además de estar validada en diferentes aspectos.

Pruebas de interfaz y usabilidad

Para realizar esta prueba se realizó un análisis detallado para validar el cumplimiento de los principios heurísticos de usabilidad en donde se determinó que cuenta con diferentes principios, como lo son minimalismo en sus procesos, uso de metáforas para el reconocimiento de otras funciones, uso de colores para el acierto y desacierto de caracteres, señalización de los procesos en donde se encuentra el usuario, asegurando así el uso de buenas prácticas.

Apéndice N. Ciclo de Versión Administrador: Iteración Listar Experiencia.

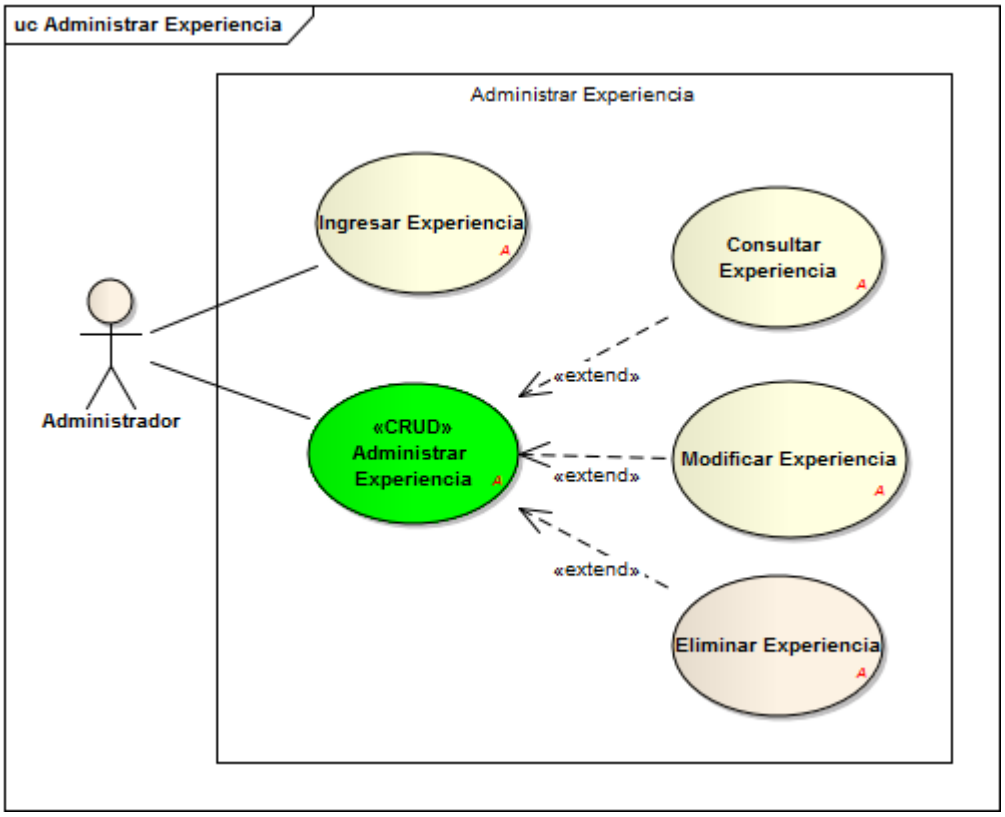


Figura N. 1.Caso de uso: Elaborar Listado de Experiencia por Administrador.

Tabla N. 1. Descripción Caso de uso Elaborar Listado de Experiencia por usuario Administrador.

Nombre del Caso de uso	Elaborar listado de experiencias.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en elaborar un listado de las experiencias registradas en la de base de datos del sistema, de manera satisfactoria.	
Precondición	Para elaborar el listado de las experiencias, deben estar registradas en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Elaborar listado de experiencias".
	2	El sistema muestra el listado de las experiencias almacenados en la base de datos del sistema, de acuerdo a un orden predeterminado, brindando las siguientes opciones de extensión: a. "Consultar experiencia". b. "Modificar experiencia" c. "Eliminar experiencia"
	3	El Administrador visualiza el listado de las experiencias y selecciona alguna de las opciones de extensión propuestas.
	4	El sistema extiende <<extend>> hacia el caso de uso que representa la opción seleccionada.
	6	El sistema finaliza el caso de uso.
Postcondición	Se elabora el listado de experiencias existentes en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no seleccionar ninguna opción de extensión, se cancela el caso de uso, seleccionando otra opción en el sistema.
	4.1.	El sistema finaliza el caso de uso.
Reglas del negocio	1. El listado de experiencias debe ser elaborado de acuerdo a un orden preestablecido. 2. Se podrá seleccionar, desde el listado general de experiencias, una experiencia en específica para realizar las operaciones de consultar, modificar o eliminar.	
Inclusiones	Ninguna.	
Extensiones	1. "Consultar experiencia". 2. "Modificar experiencia". 3. "Eliminar experiencia".	

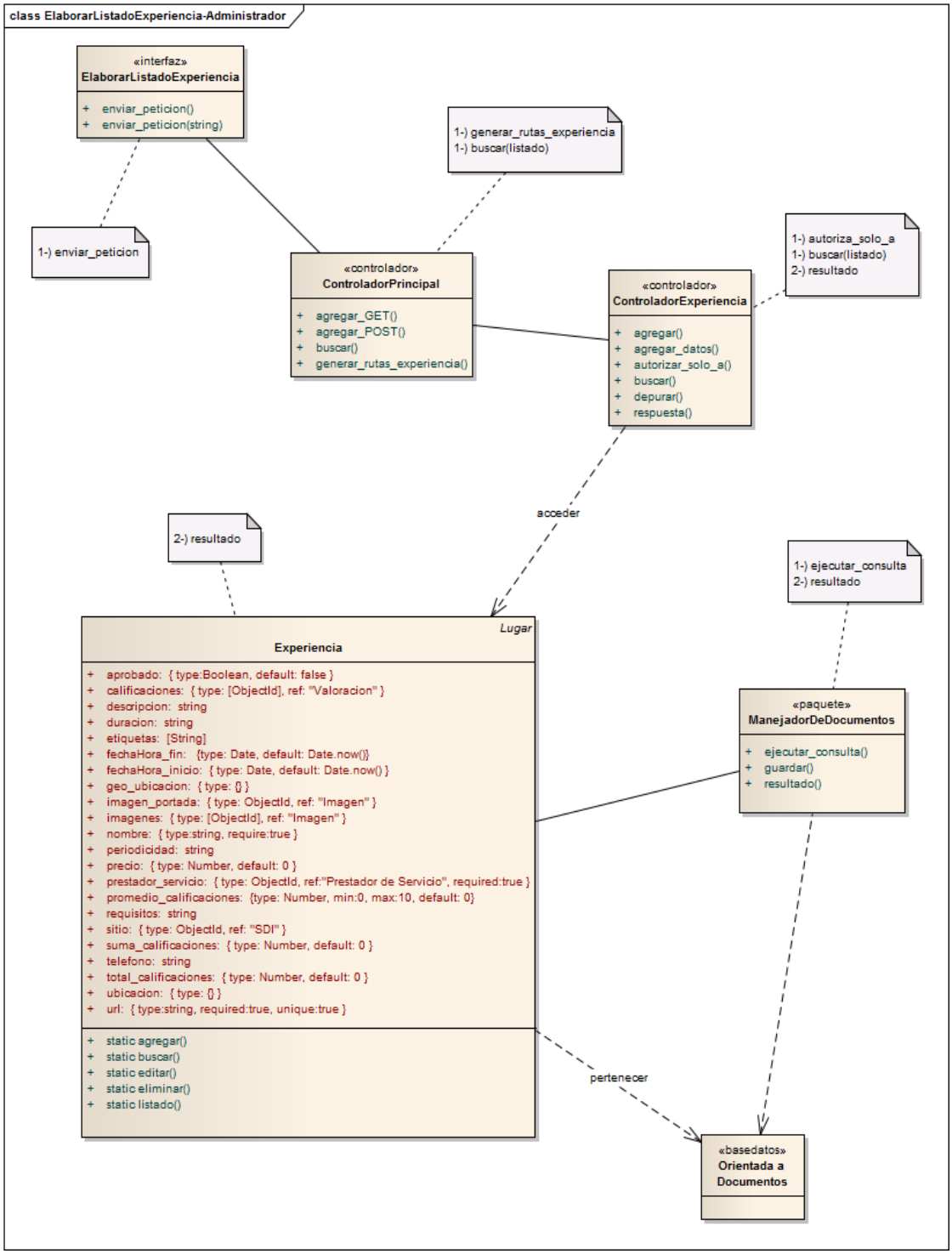


Figura N. 2.Diagrama de Clases: Elaborar Listado de Experiencia por usuario Administrador.

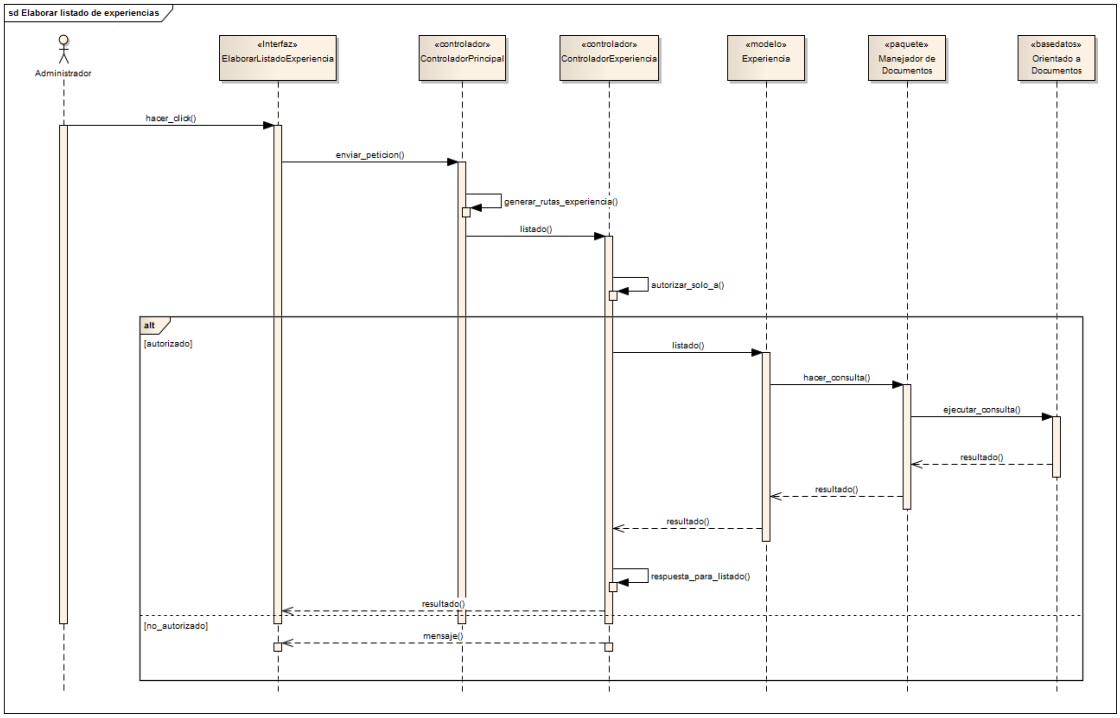


Figura N. 3.Diagrama de secuencia: Elaborar Listado de Experiencia por usuario Administrador.

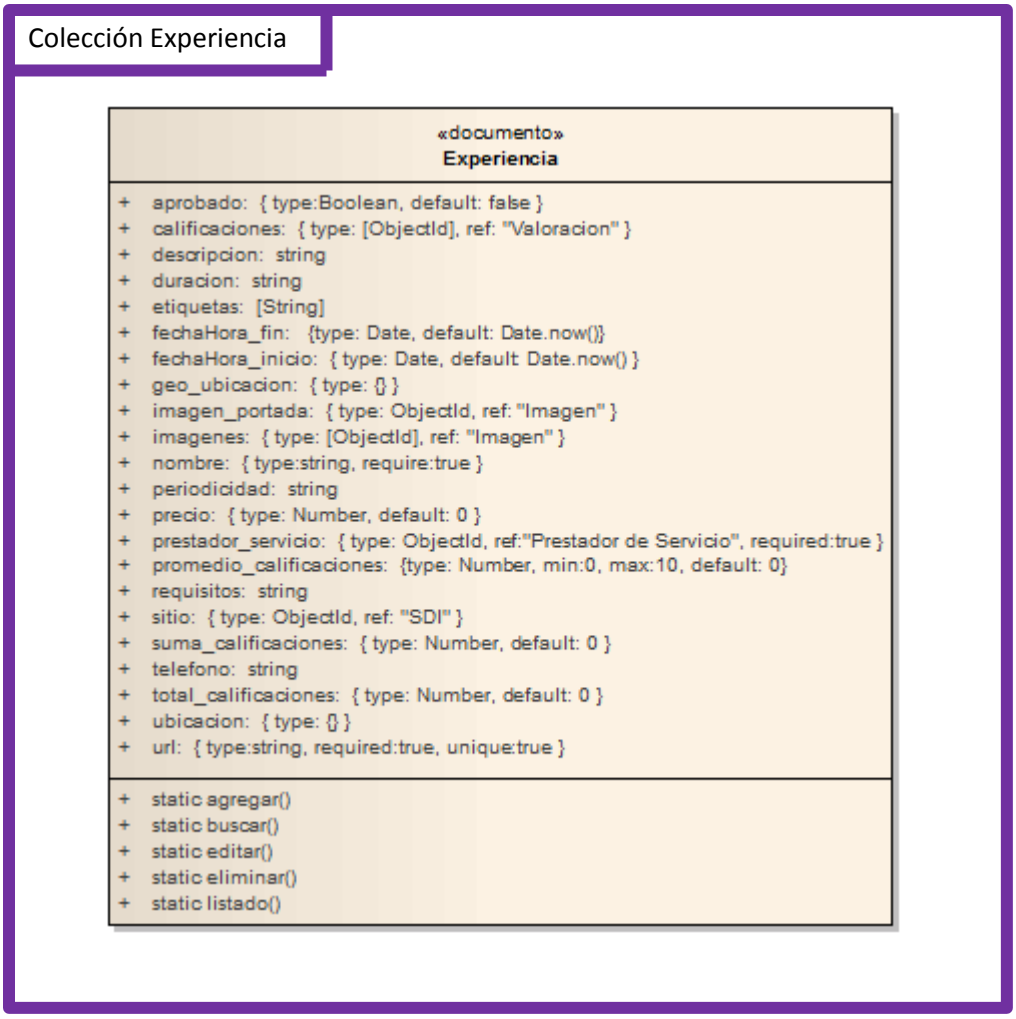


Figura N. 4. Colección Experiencia: Eliminar experiencia por usuario Administrador

A continuación, se muestran el modelo de plantilla para el proceso de listar experiencia.

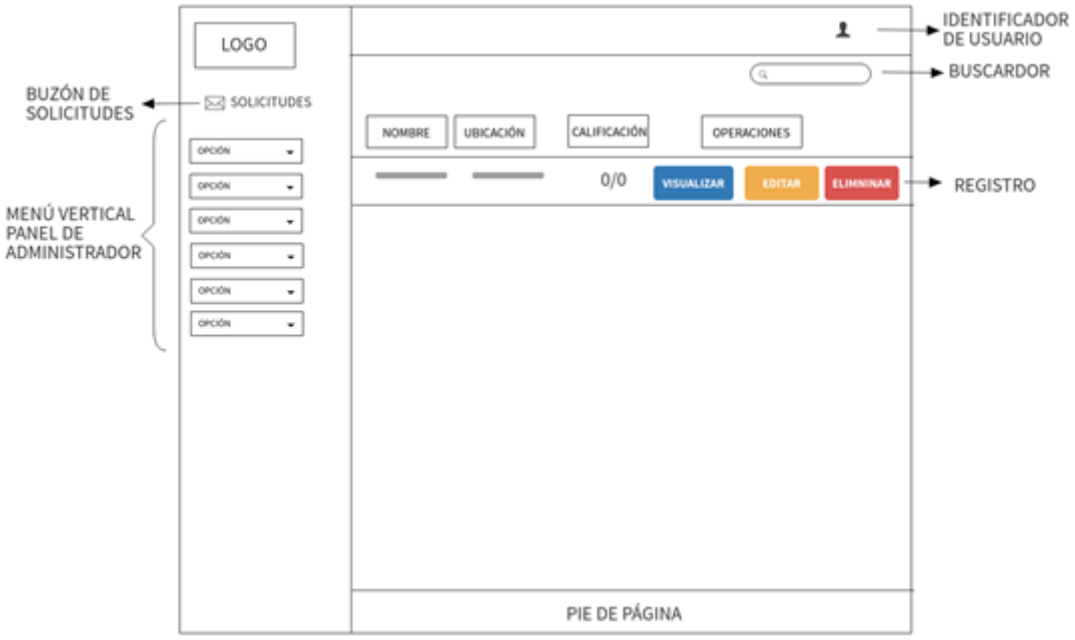


Figura N. 5.Plantilla de interfaz: Listar Experiencia por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Lista de Eventos y el sistema responde mostrando la interfaz correspondiente.

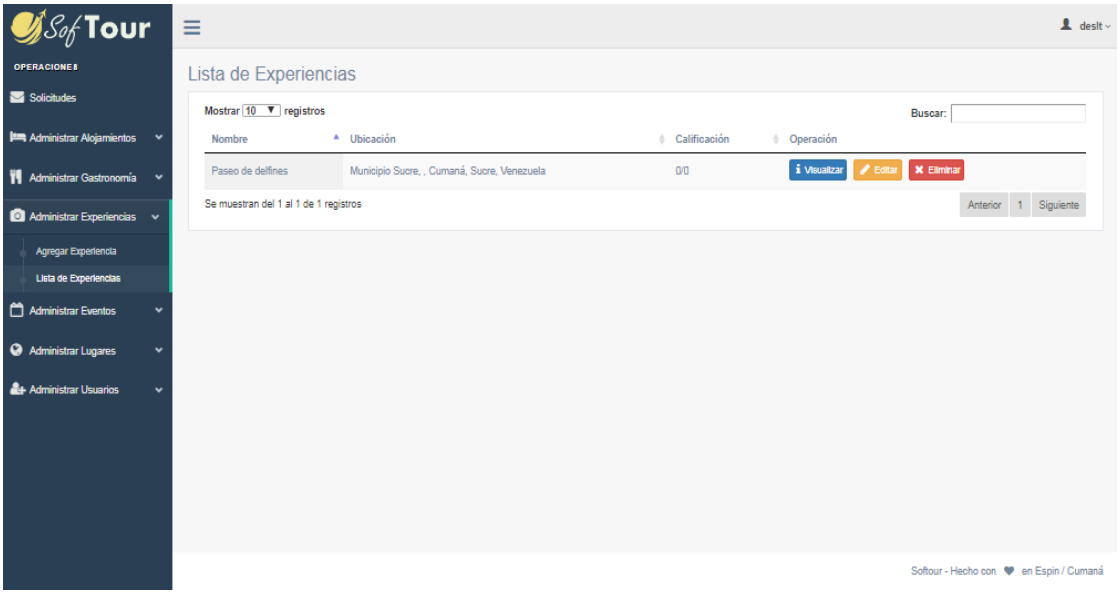


Figura N. 6. Interfaz Listado de Experiencia por usuario Administrador.

Código desarrollado para la funcionalidad Listado de Eventos por usuario administrador

```
listado(req, res, next) {
    let query = {"aprobado": true};

    if(req.session.passport.user.tipo_usr == "PrestadorServicio")
        query.prestador_servicio = req.session.passport.user.id;

    this._Experiencia.listado({"consulta": query}, (error, lista) => {

        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_listado(req, res);
        }

        res.locals.datos = {"value": lista};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_listado(req, res);
    })
}
```

Figura N. 7. Código desarrollado para el proceso de Listar Experiencia por usuario Administrador.

Apéndice O. Ciclo de Versión Administrador: Iteración Consultar Experiencia.

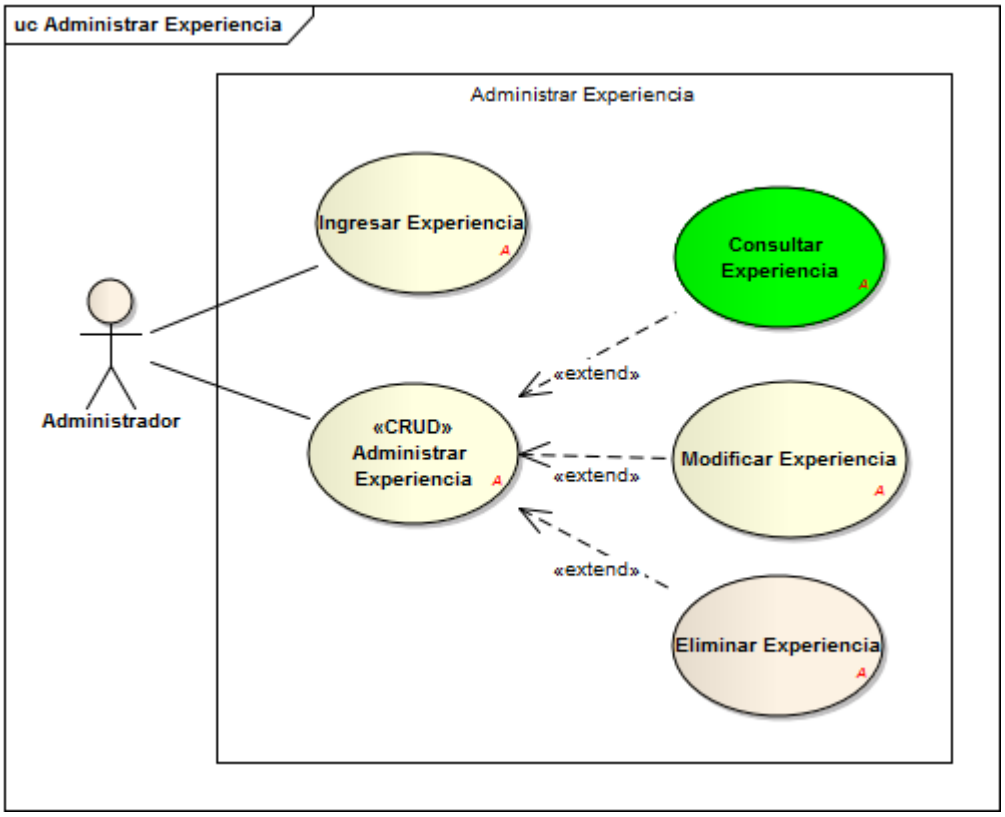


Figura O. 1.Caso de uso: Consultar Experiencia por Administrador

Tabla O. 1. Descripción Caso de uso Consultar Experiencia por usuario Administrador

Nombre del Caso de uso	Consultar experiencia.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en consultar los atributos de la experiencia de manera satisfactoria.	
Precondición	Para consultar una experiencia, debe estar registrada en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Consultar”.
	2	El sistema ejecuta la búsqueda de la experiencia, mostrando un formulario con la consulta.
	3	El Administrador visualiza la consulta de los atributos de la experiencia.
	4	El sistema finaliza el caso de uso.
Postcondición	Se consultan los atributos de la experiencia de la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
Reglas del negocio	1. La experiencia debe estar registrada para que pueda ser consultada.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

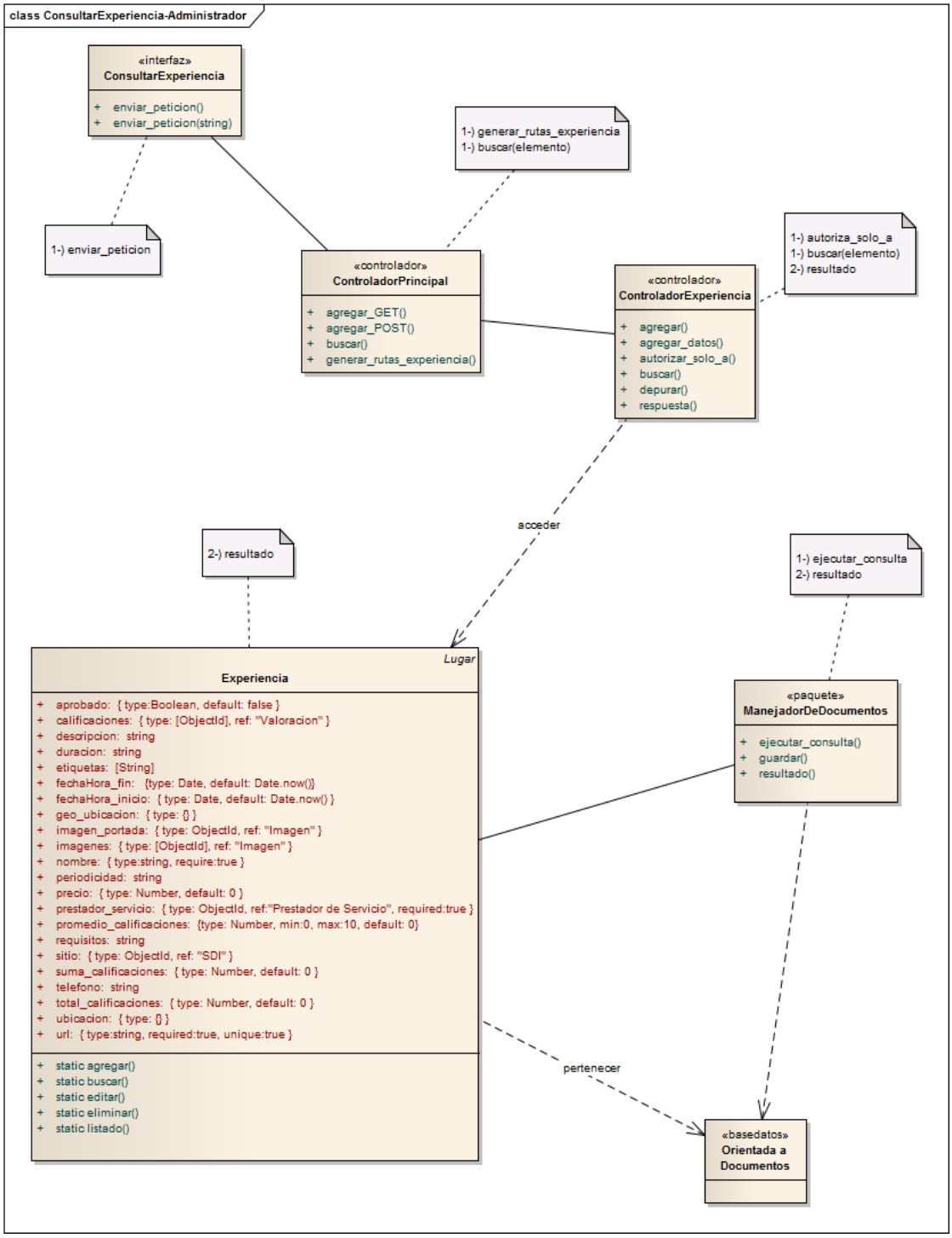


Figura O. 2. Diagrama de Clases: Consultar Experiencia por usuario Administrador.

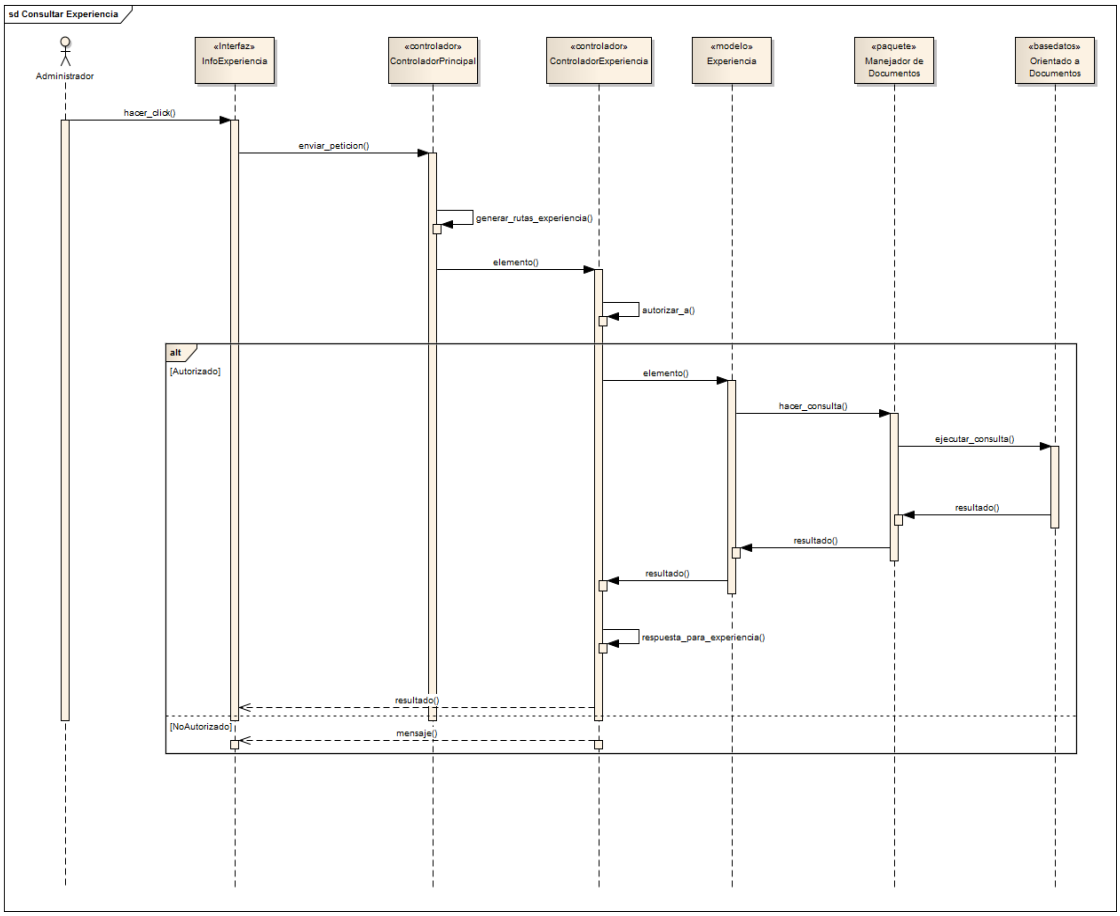


Figura O. 3.Diagrama de secuencia: Consultar Experiencia por usuario Administrador.

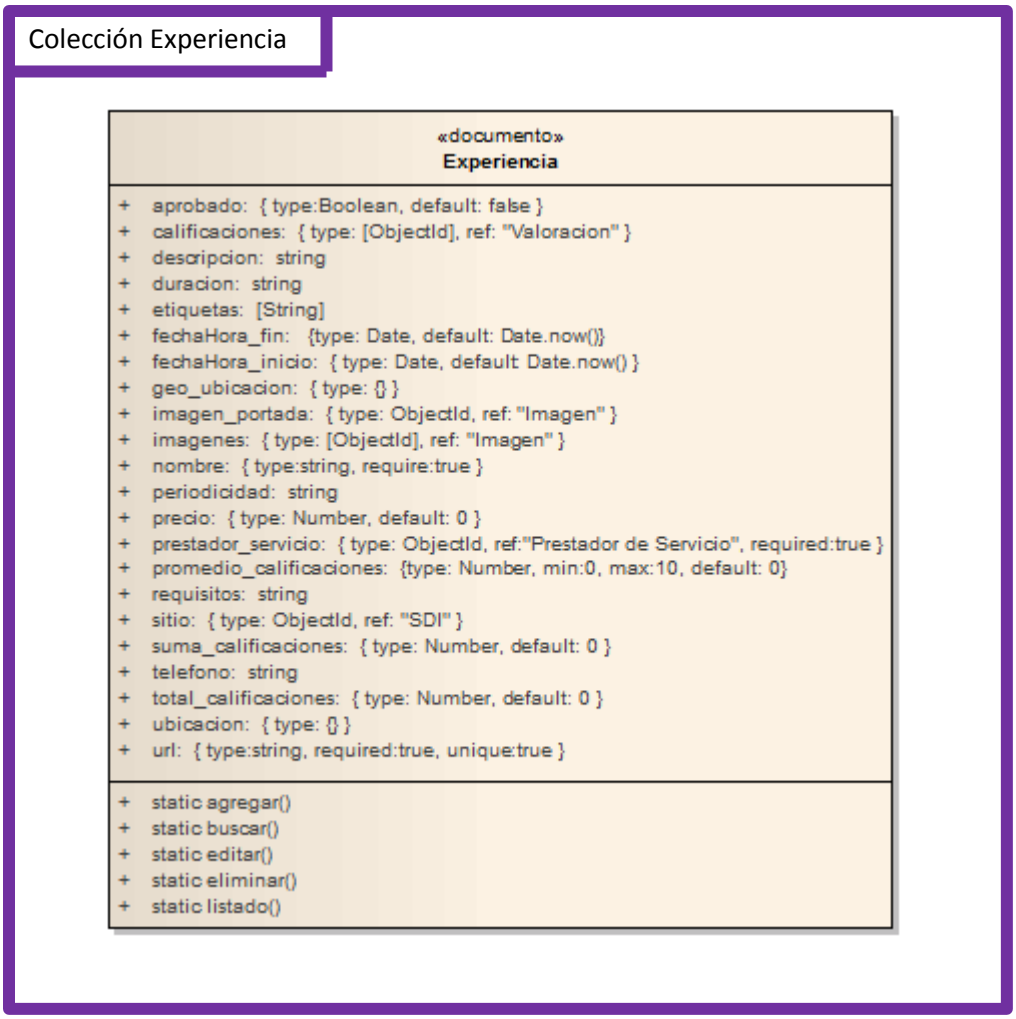


Figura O. 4.Colección Experiencia: Consultar Experiencia por usuario Administrador.

A continuación, se muestran el modelo de plantilla para el proceso de consultar experiencia.

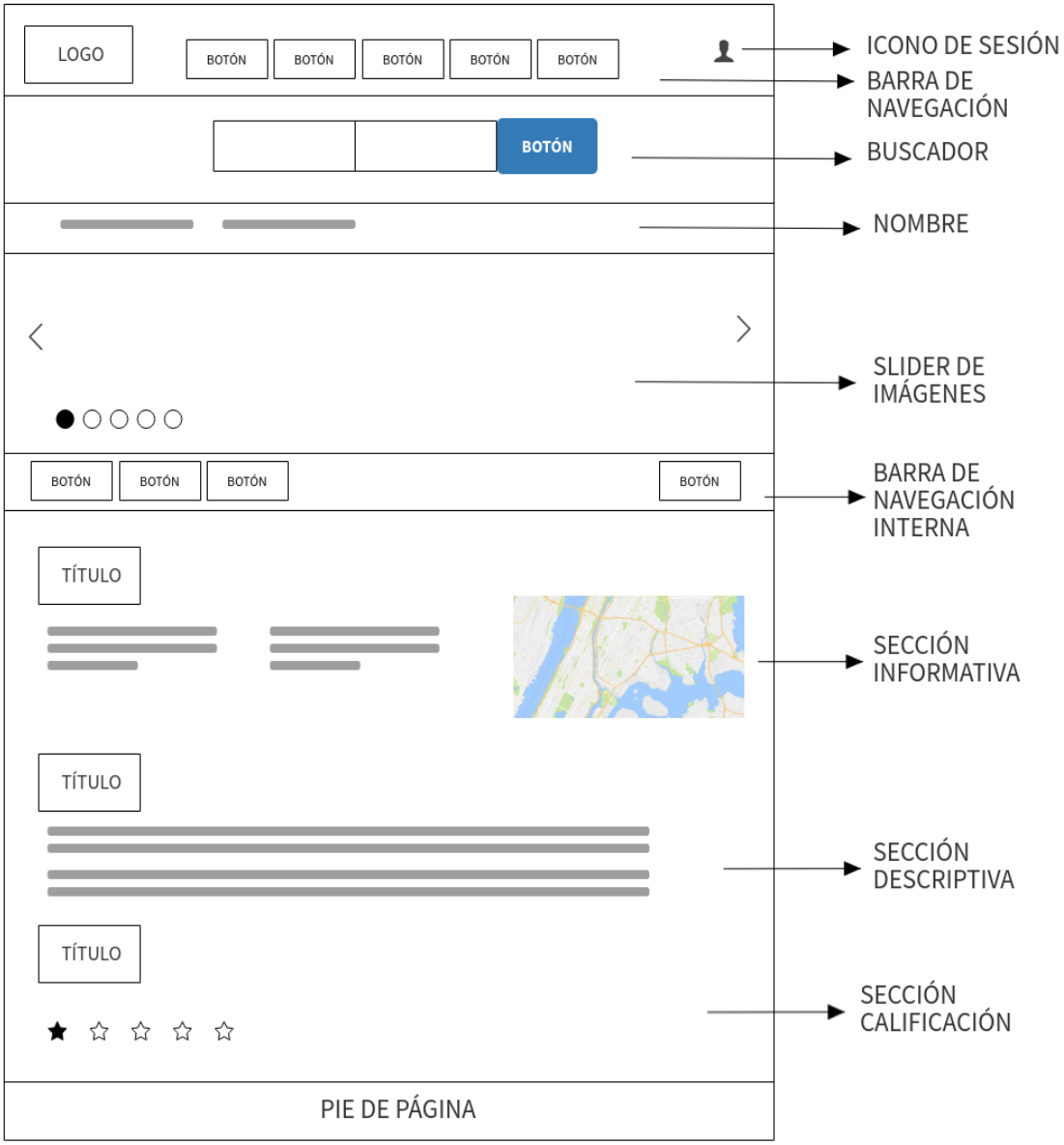


Figura O. 5.Plantilla de interfaz: Consultar Experiencia por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Consultar Experiencia y el sistema responde mostrando la interfaz con las especificaciones de la actividad.

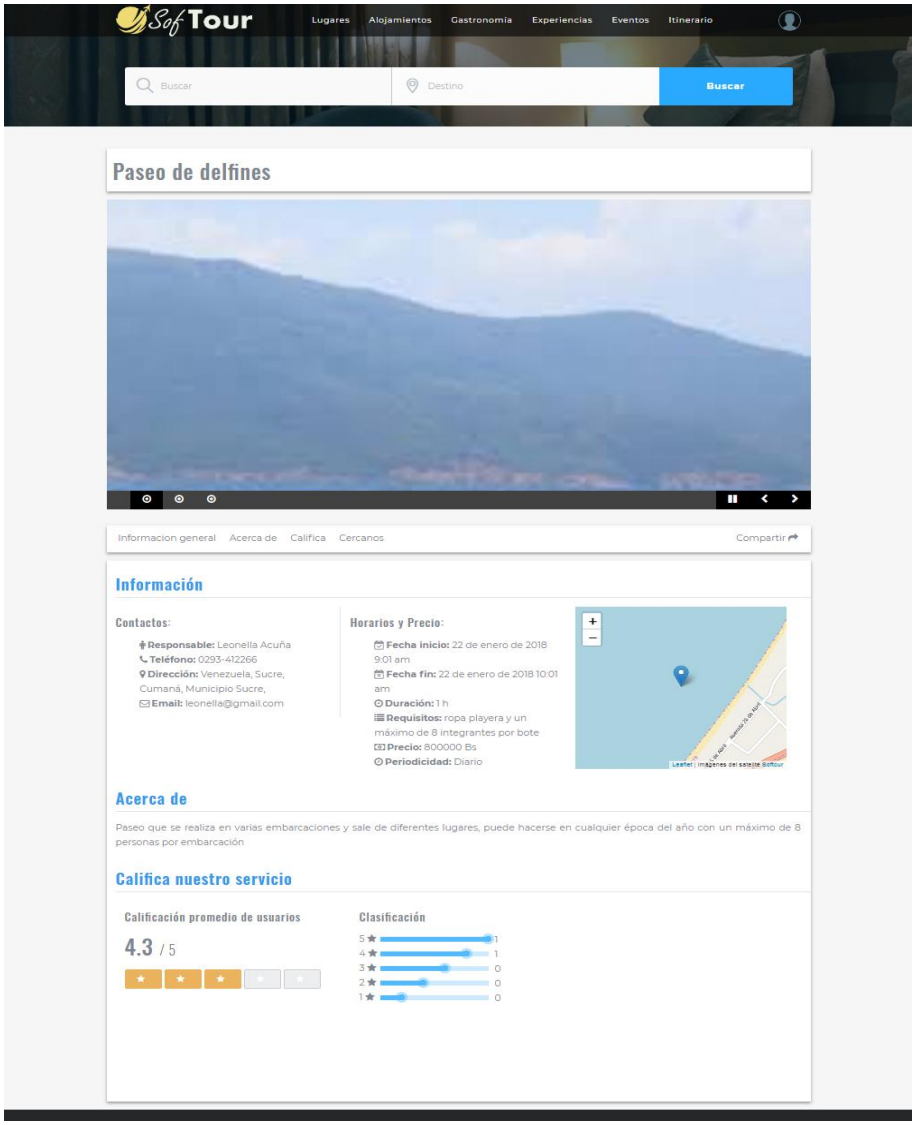


Figura O. 6.Interfaz de proceso de Consultar Experiencia por usuario Administrador.

Código desarrollado para la funcionalidad Consultar Detalle Alojamiento por usuario administrador

```
consultar(req, res, next) {
    this._Experiencia.buscar_consulta({"url": req.params.url.toLowerCase()}, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": buscado};
        if (req.isAuthenticated())
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_consultar(req, res);
    })
}
```

Figura O. 7. Código desarrollado para el proceso Consultar Experiencia por usuario Administrador.

Prueba visualizar experiencia

La función Visualizar experiencia consiste en consultar los datos de una experiencia agregada con anterioridad al sistema. Para aplicar pruebas a este módulo se utilizaron datos ingresados con anterioridad.

Se procede a seleccionar una experiencia, luego se selecciona visualizar, tal como se muestra en las siguientes imágenes

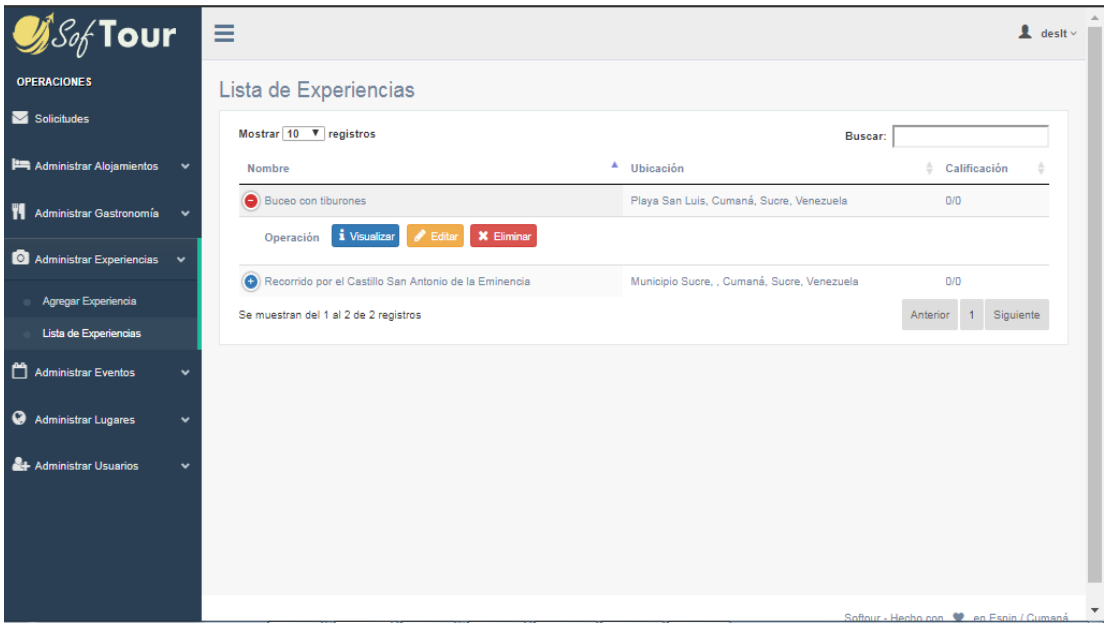


Figura O. 8.Prueba Consultar Experiencia por el usuario Administrador. Paso 1

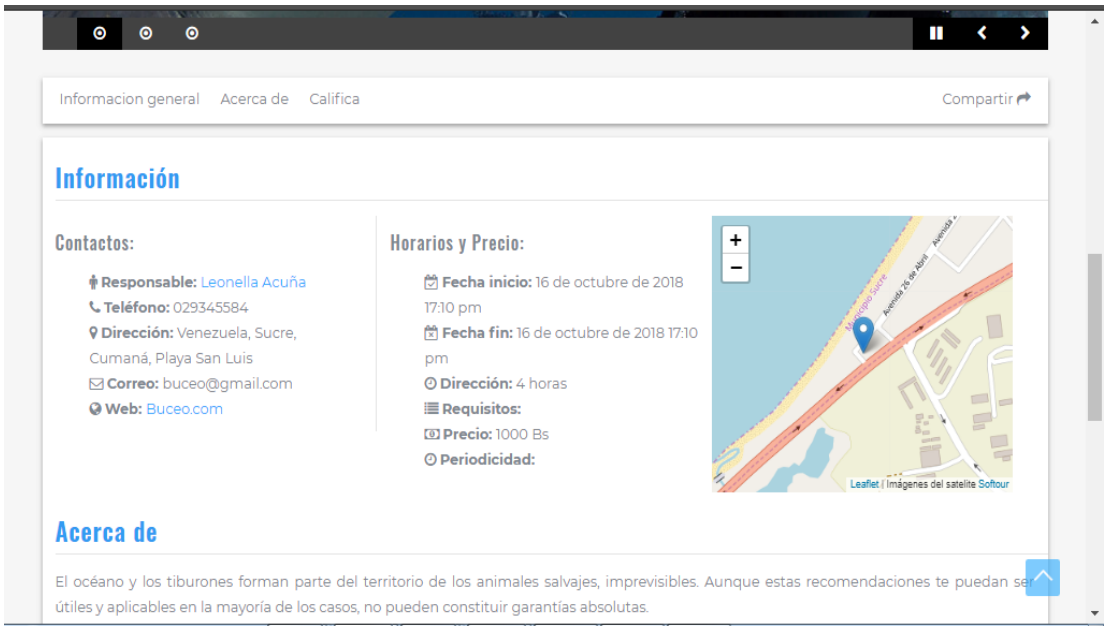


Figura O. 9. Prueba Consultar Experiencia por el usuario Administrador. Paso 2

Conclusión de la prueba

La función cumple con el objetivo para la cual fue desarrollada.

Pruebas de interfaz y funcionalidad

La interfaz mantiene consistencia con las demás interfaces que han evaluado.

Apéndice P. Ciclo de Versión Administrador: Iteración Modificar Experiencia.

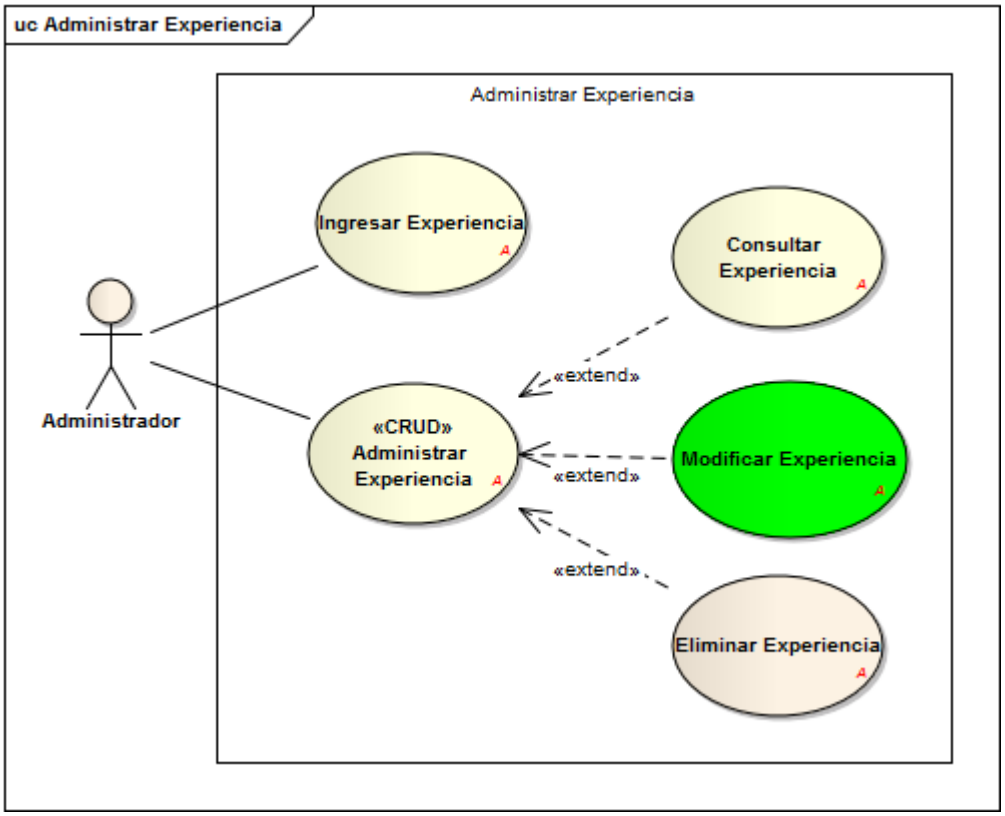


Figura P. 1.Caso de uso: Modificar Experiencia por Administrador.

Tabla P. 1. Descripción Caso de uso Modificar Experiencia por usuario Administrador

Nombre del Caso de uso	Modificar experiencia.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en modificar los atributos de la experiencia de manera satisfactoria.	
Precondición	Para modificar una experiencia, debe estar registrada en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Modificar”.
	2	El sistema ejecuta la búsqueda de la experiencia, mostrando un formulario con la consulta en modo edición.
	3	El Administrador visualiza la consulta de los atributos de la experiencia que desea modificar, los modifica y selecciona aceptar.
	4	El sistema realiza las modificaciones en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se modifican los atributos de la experiencia en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no modificar la experiencia y selecciona cancelar el caso de uso, seleccionando otra opción en el sistema.
	4.1.	El sistema finaliza el caso de uso.
Reglas del negocio	1. La experiencia debe estar registrada para que pueda ser modificada.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

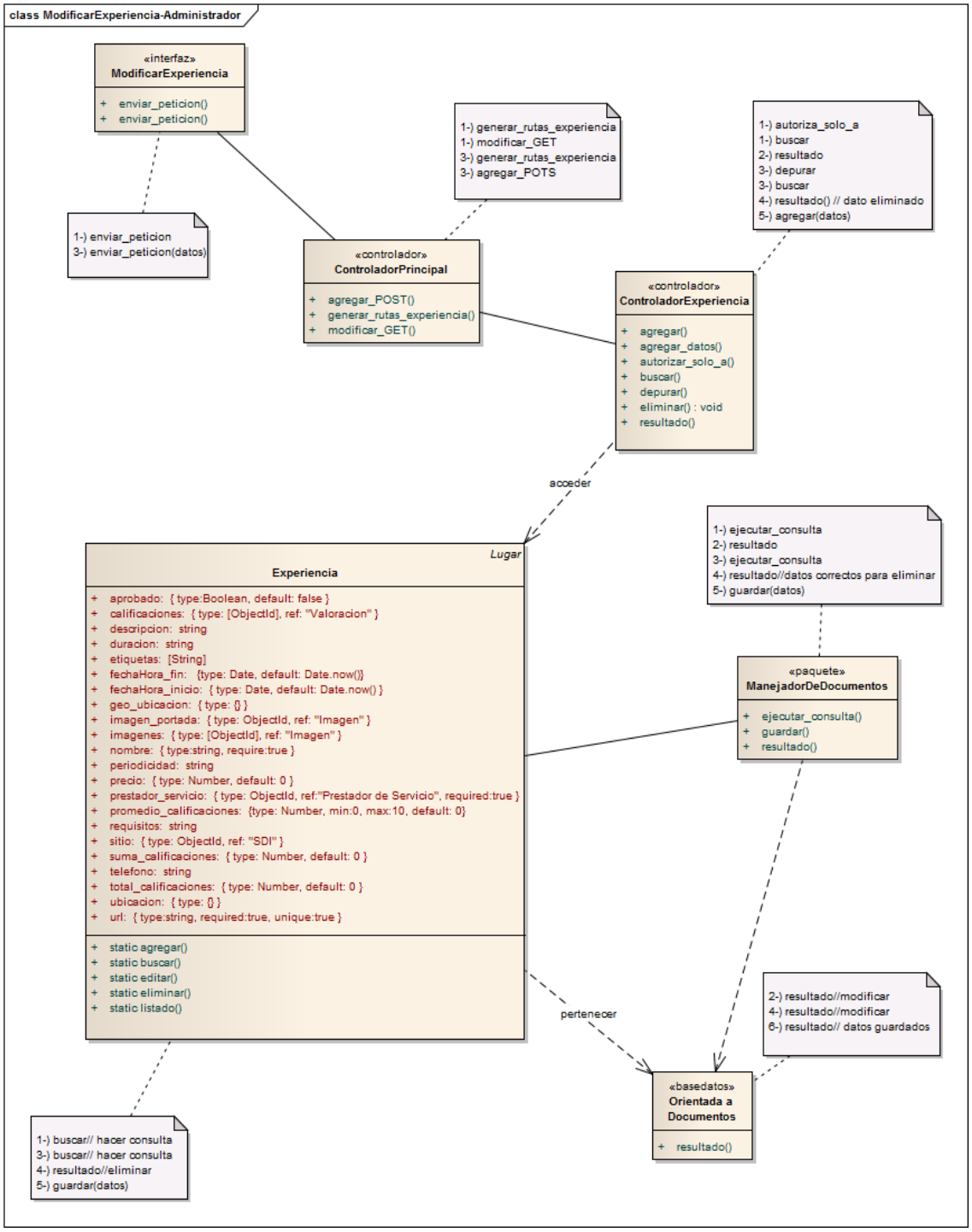


Figura P. 2.Diagrama de Clases: Modificar Experiencia por usuario Administrador.

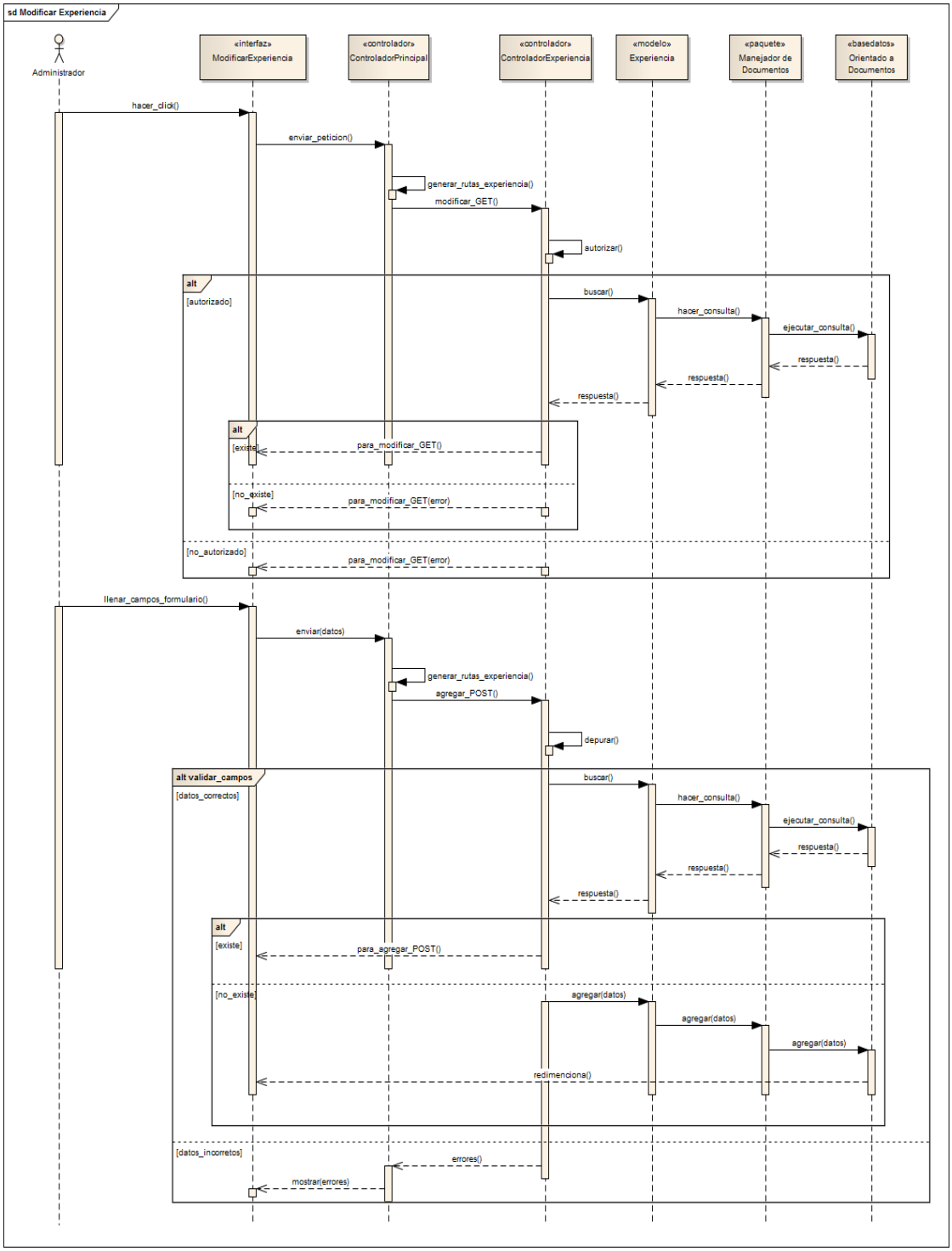
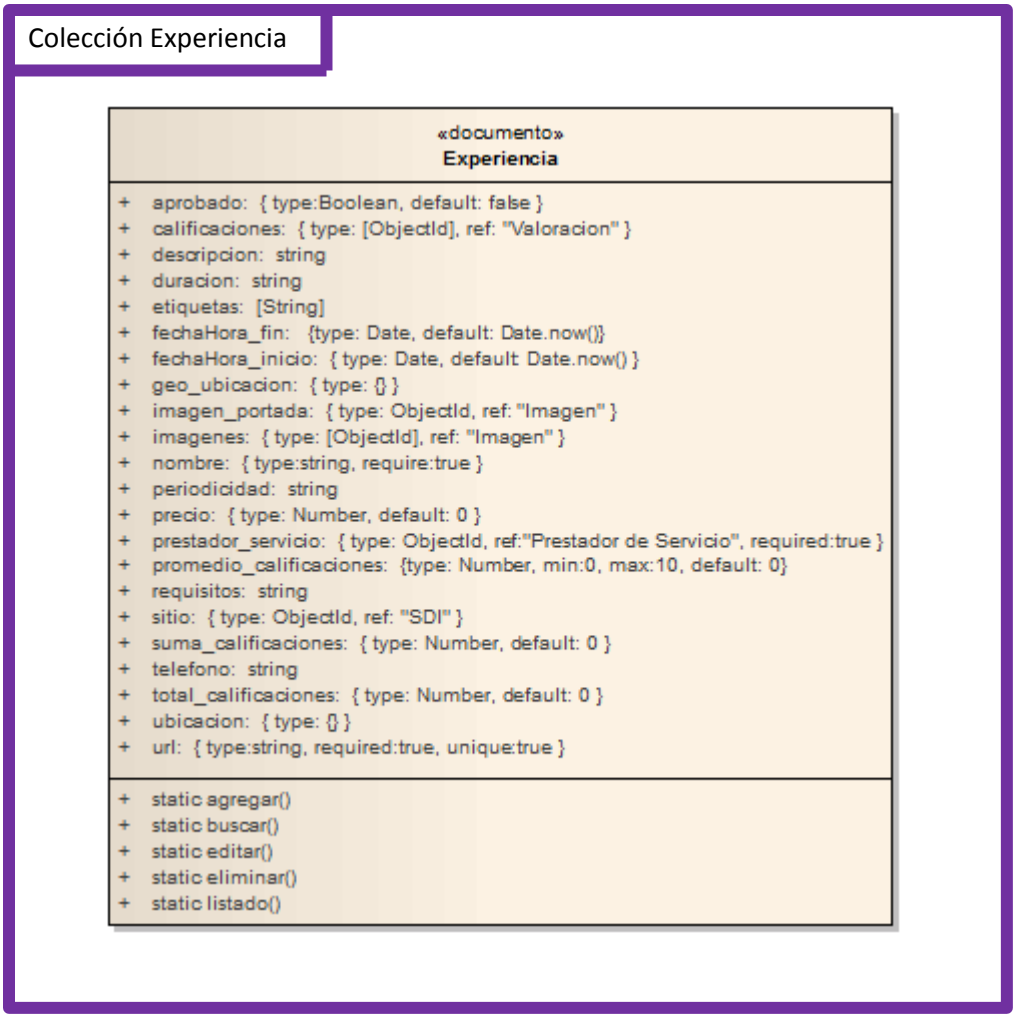


Figura P. 3.Diagrama de secuencia: Modificar Experiencia por usuario Administrador



I

Figura P. 4. Colección Experiencia: Modificar experiencia por usuario Administrador.

A continuación, se muestran los modelos de plantillas para el proceso de modificar experiencia.

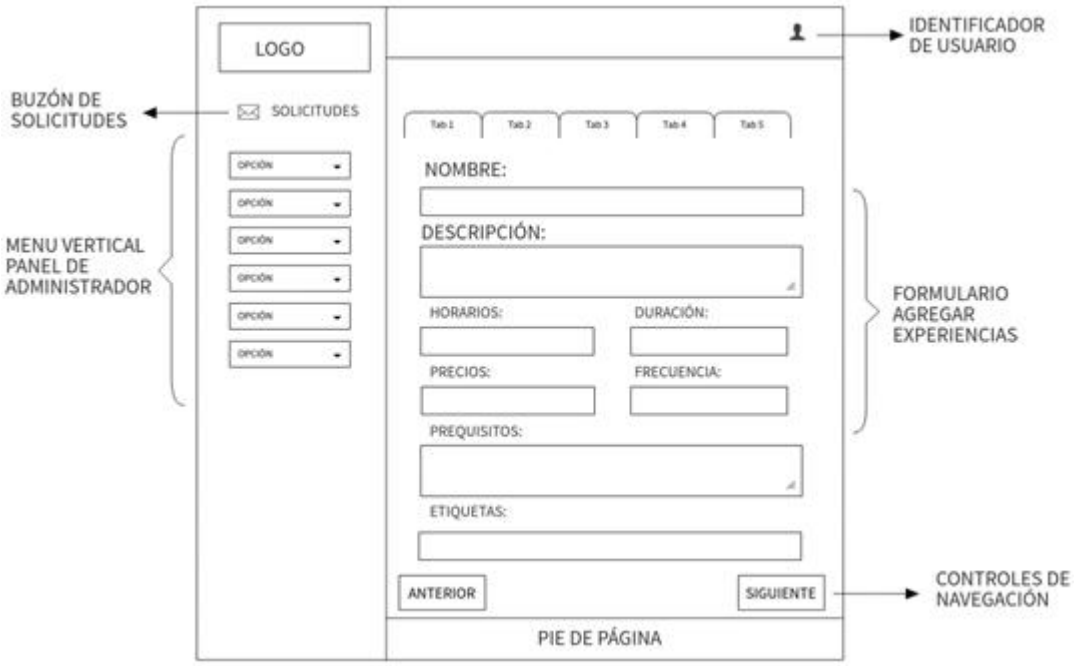


Figura P. 5. Plantilla de interfaz 1: Modificar Experiencia por usuario Administrador.

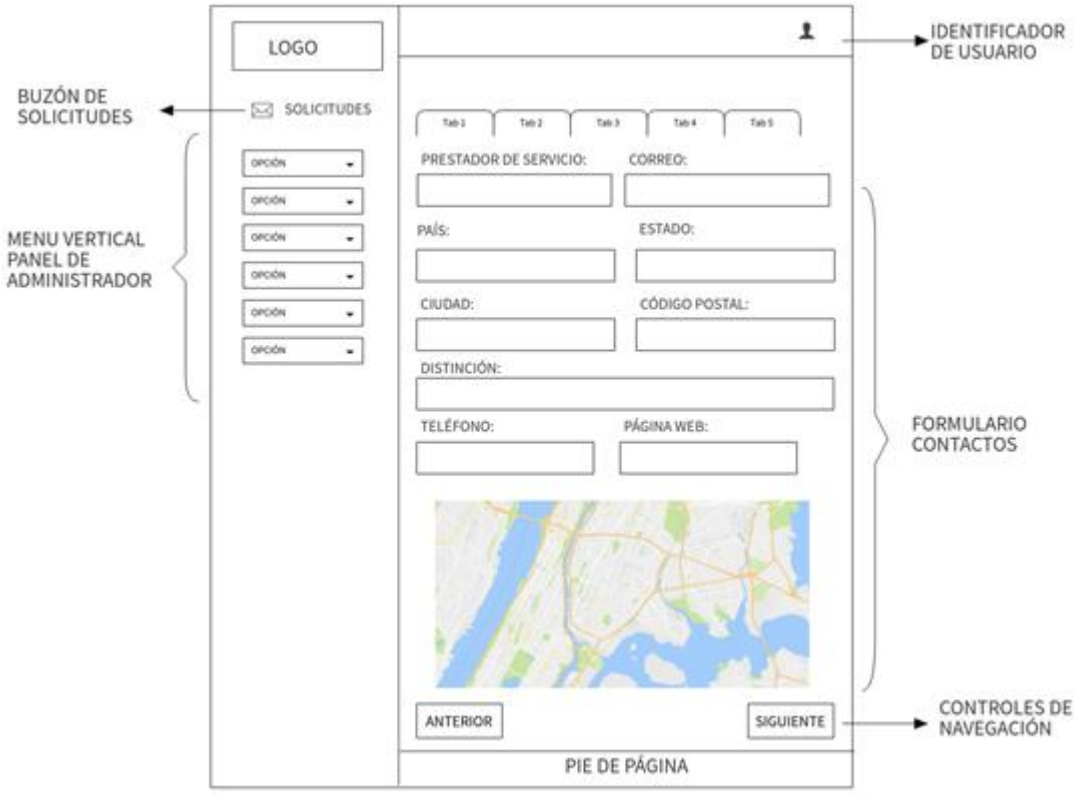


Figura P. 6.Plantilla de interfaz 2: Modificar Experiencia por usuario Administrador.

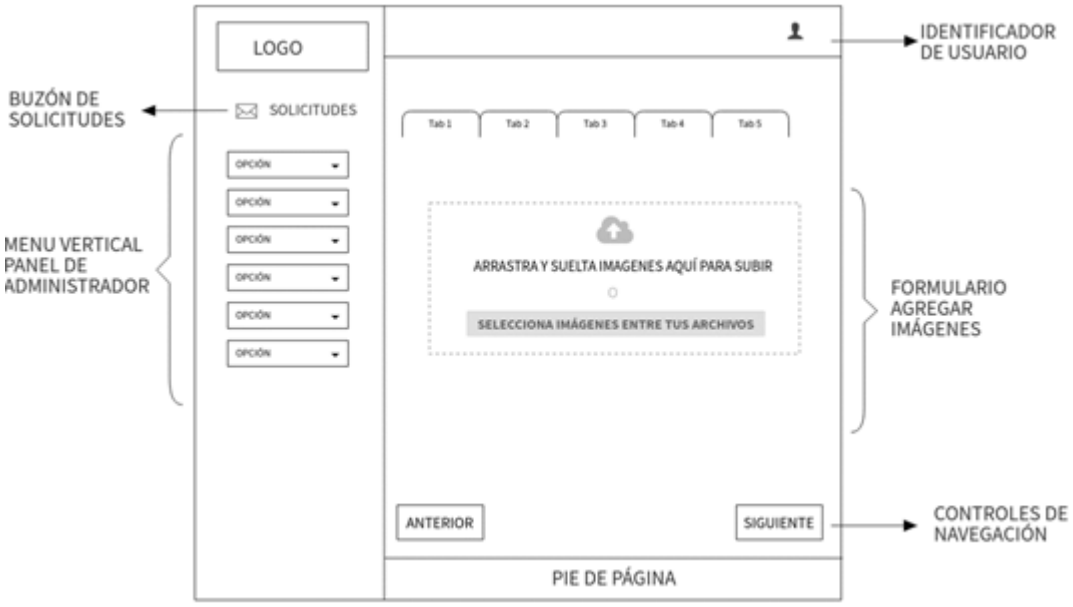


Figura P. 7.Plantilla de interfaz 3: Modificar Experiencia por usuario Administrador.

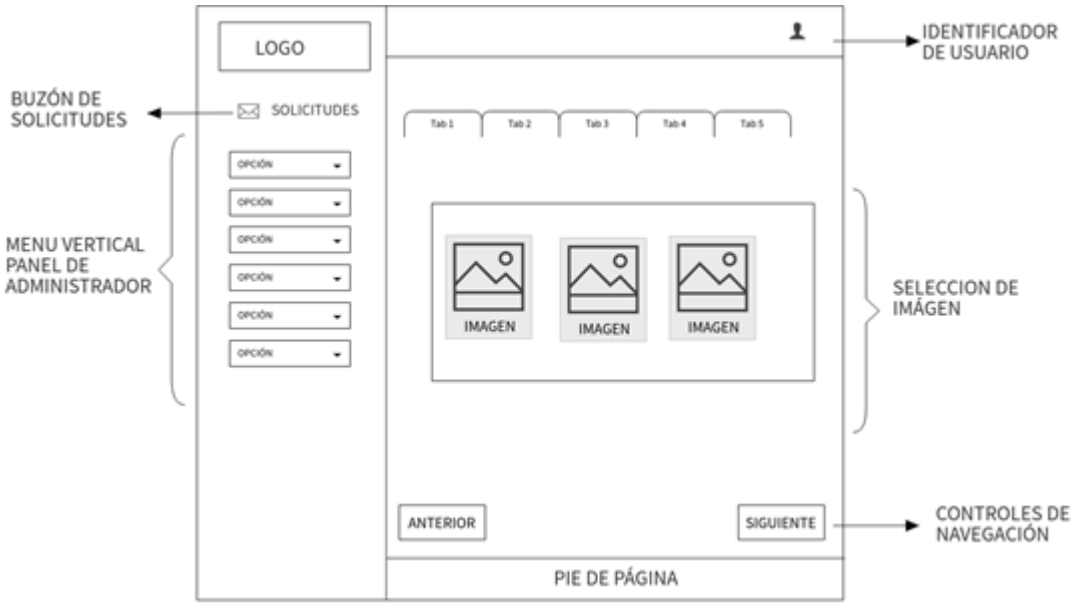


Figura P. 8.Plantilla de interfaz 4: Modificar Experiencia por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Modificar Experiencia y el sistema responde mostrando el formulario correspondiente.

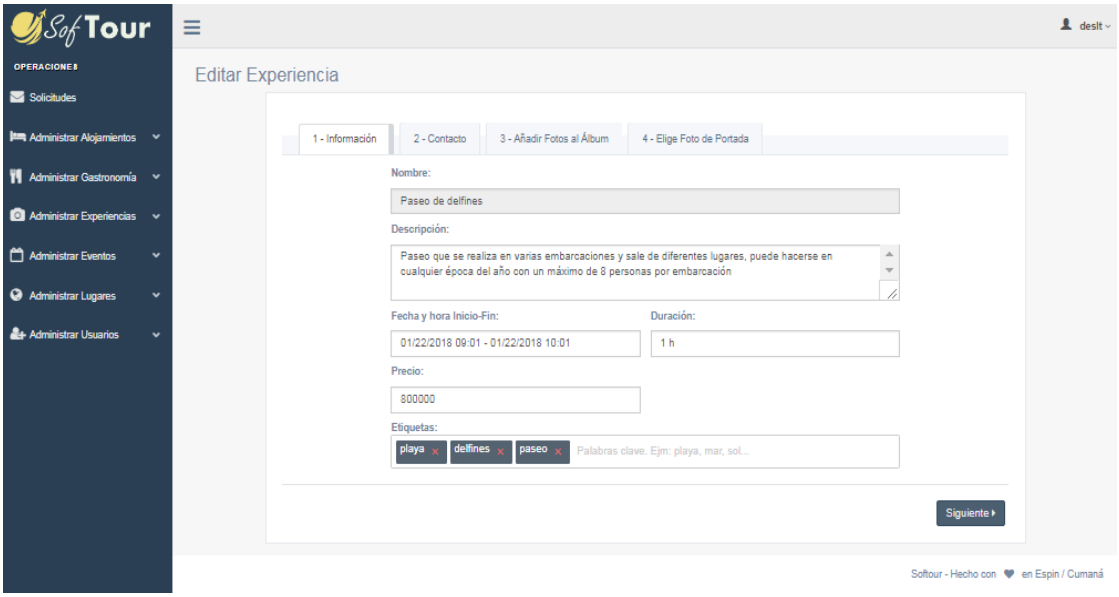


Figura P. 9.Interfaz 1 de proceso de Modificar Experiencia por usuario Administrador.

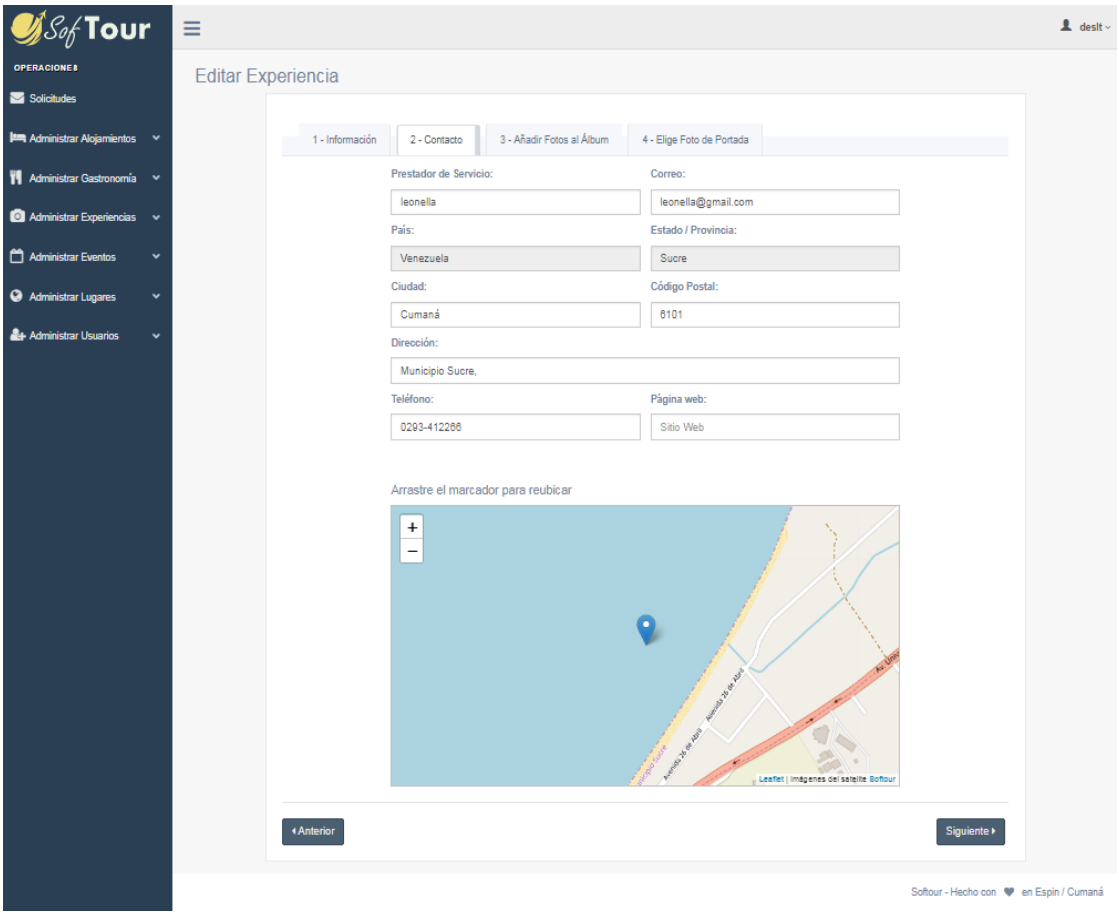


Figura P. 10.Interfaz 2 de proceso de Modificar Experiencia por usuario Administrador.

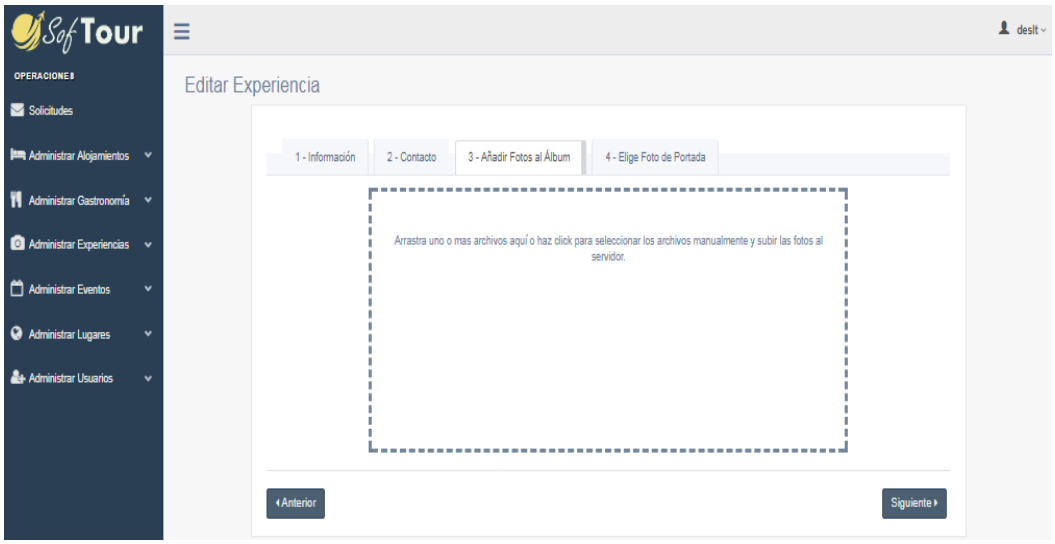


Figura P. 11.Interfaz 3 de proceso de Modificar Experiencia por usuario Administrador

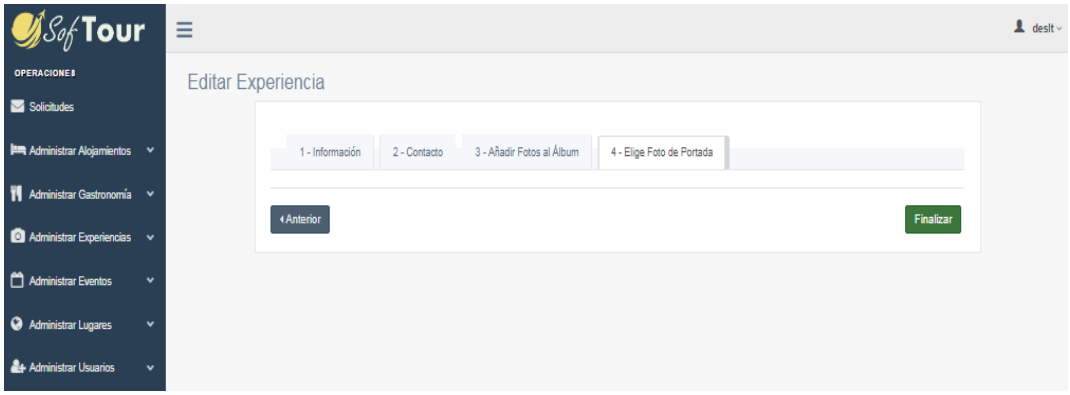


Figura P. 12.Interfaz 4 de proceso de Modificar Experiencia por usuario Administrador.

Código desarrollado para la funcionalidad Modificar Alojamiento por usuario administrador

```
modificar(req, res, next) {
    this._validar.campos(req.body, true, (error, resultados) => {
        if(error) return next(error);
        res.locals.datos = resultados;

        if(res.locals.datos.error){
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_modificar(req, res);
        }

        else {
            this._Experiencia.buscar({"url": req.params.url.toLowerCase()}, (error, buscado) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
                    this._respuesta.para_modificar(req, res);
                }

                if(!buscado) return next(new this._error_NoEncontrado());
                res.locals.datos.value.geo_ubicacion = {
                    "type": "Point",
                    "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
                };
                res.locals.datos.value.ubicacion = {
                    "direccion": res.locals.datos.value.direccion,
                    "pais": res.locals.datos.value.pais,
                    "estado": res.locals.datos.value.estado,
                    "ciudad": res.locals.datos.value.ciudad,
                    "codigo_postal": res.locals.datos.value.codigo_postal
                };

                res.locals.datos.value.url = req.params.url.toLowerCase();
                this._Experiencia.modificar(res.locals.datos.value, (error, modificado) => {
                    if(error) {
                        res.locals.datos.error = error;
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_modificar(req, res);
                    }
                    if(!modificado) return next(new this._error_NoEncontrado());

                    res.locals.datos = {"value": modificado};
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_modificar(req, res);
                });
            });
        }
    });
}
```

Figura P. 13. Código desarrollado para el Modificar Experiencia por usuario Administrador.

Prueba editar experiencia

La función Editar experiencia funciona de forma correcta cuando se editan experiencias agregadas previamente en el sistema, para realizar esta prueba se tomaron en cuenta las experiencias agregadas en la prueba realizada a la función Ingresar experiencia.

Para comprobar la funcionalidad se ingresó a algunas experiencias creadas con anterioridad, se modificó y se procedió a visualizar si se guardaron los cambios realizados.

A continuación, se muestran imágenes de una de las pruebas que se le realizó a la función para validar que cumpliera con su objetivo, la información a cambiar será:

Información: El océano y los tiburones forman parte del territorio de los animales salvajes, imprevisibles. Aunque estas recomendaciones te puedan ser útiles y aplicables en la mayoría de los casos, no pueden constituir garantías absolutas.

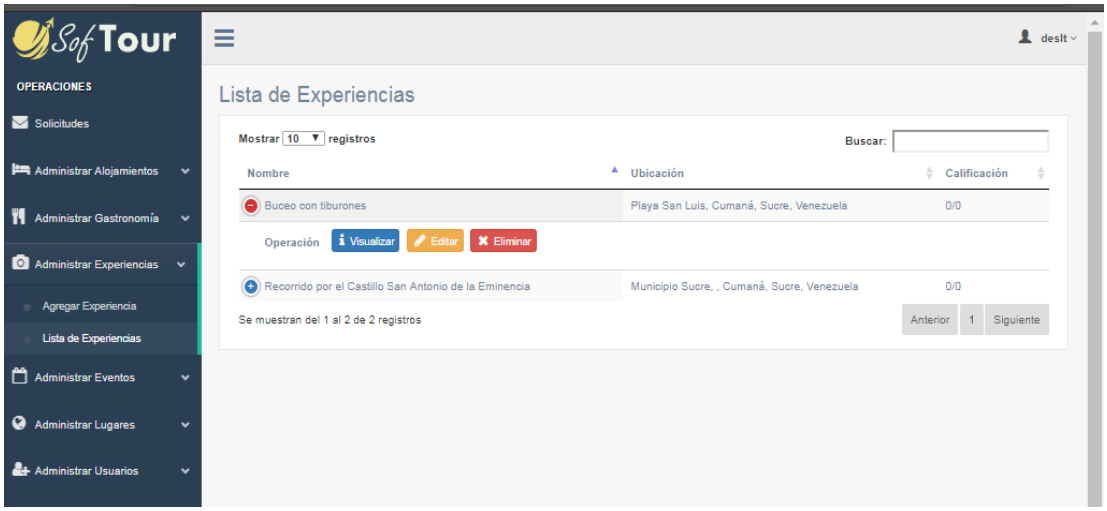


Figura P. 14.Prueba Modificar Experiencia por Administrador. Paso 1.

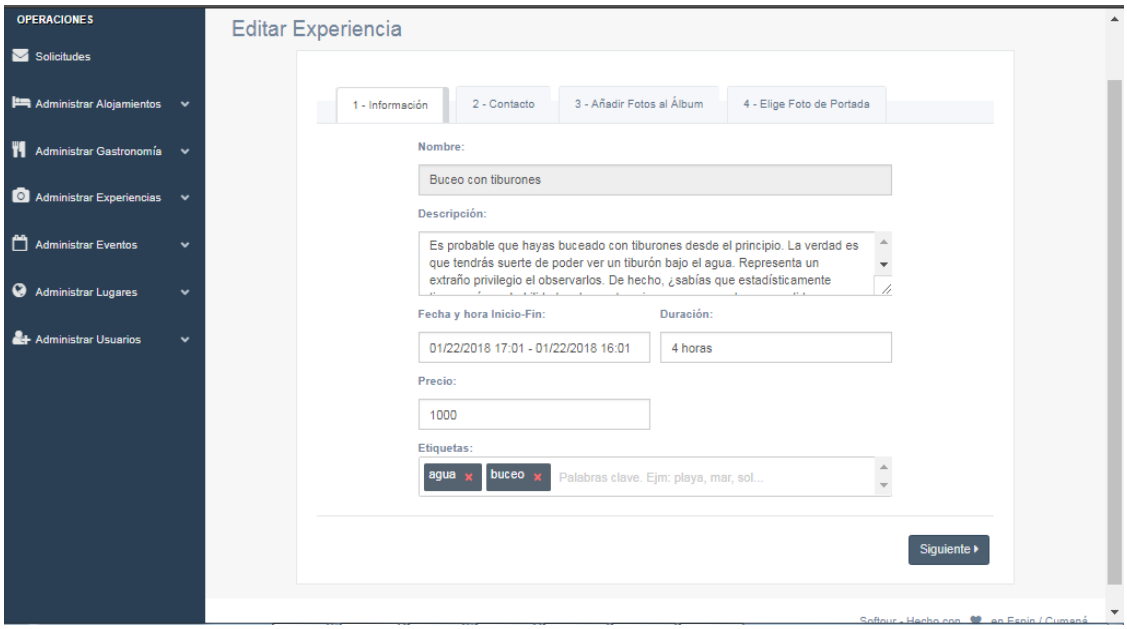


Figura P. 15.Prueba Modificar Experiencia por Administrador. Paso 2.

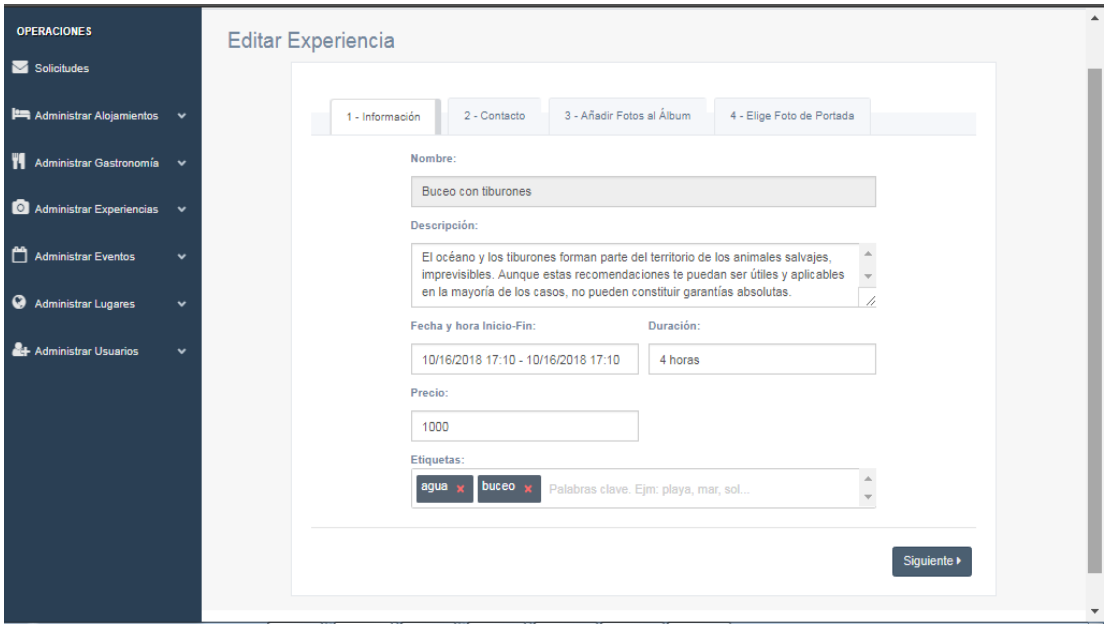


Figura P. 16.Prueba Modificar Experiencia por Administrador. Paso 3.

Conclusión de prueba

La función cumple con el objetivo para la cual fue desarrollada.

Prueba de interfaz y usabilidad

Esta interfaz pertenece al conjunto de interfaz de usuario que se evaluó con anterioridad y que fue diseñada bajo la premisa de cumplir con estándares de usabilidad. Se puede constatar que son consistentes a pesar de las diferentes funciones.

Apéndice Q. Ciclo de Versión Administrador: Iteración Eliminar Experiencia.

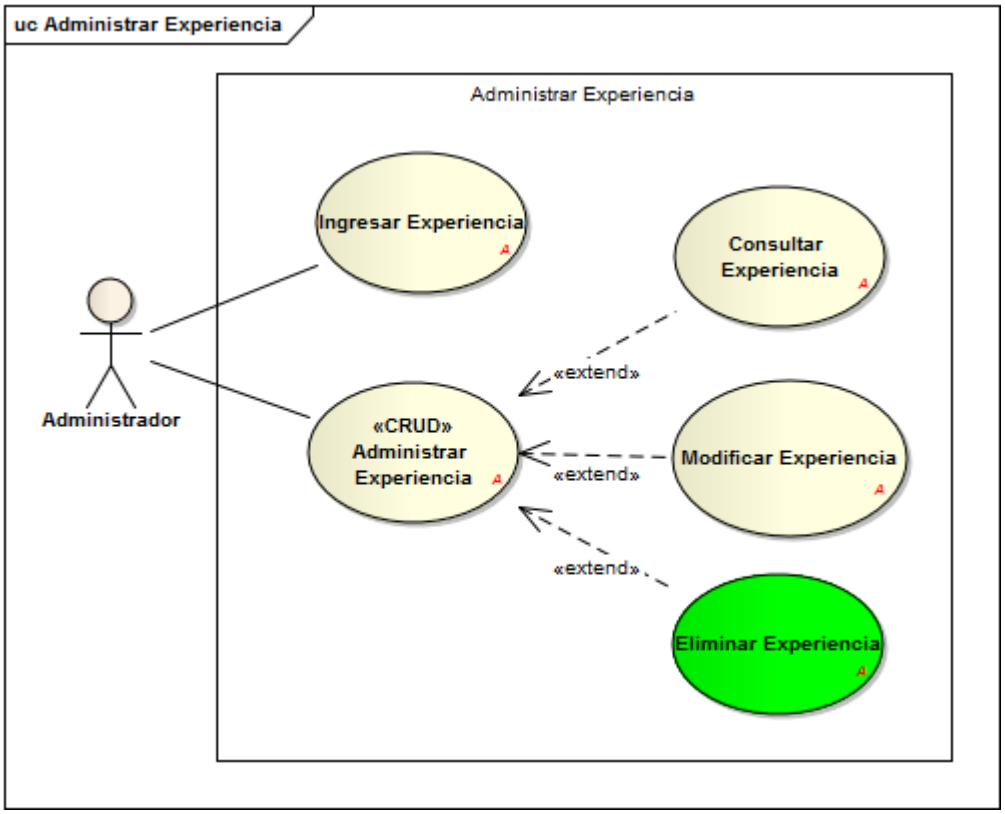


Figura Q. 1.Caso de uso: Eliminar Experiencia por Administrador.

Tabla Q. 1. Descripción Caso de uso Eliminar Experiencia por usuario Administrador.

Nombre del Caso de uso	Eliminar experiencia.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en eliminar una experiencia de manera satisfactoria.	
Precondición	Para eliminar una experiencia, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Eliminar”.
	2	El sistema ejecuta la búsqueda de la experiencia, a eliminar.
	3	El Administrador visualiza la experiencia que desea eliminar.
	4	El sistema muestra un mensaje “¿Está seguro de eliminar la experiencia seleccionada?”
	5	El administrador selecciona la opción “Si, estoy seguro”
	6	El sistema realiza la eliminación de la experiencia en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se elimina la experiencia de la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	5.1.	El Administrador decide no continuar con el caso de uso y selecciona cancelar el caso de uso, seleccionando otra opción del sistema.
	5.2.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. La experiencia debe estar registrado para que pueda ser eliminada.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

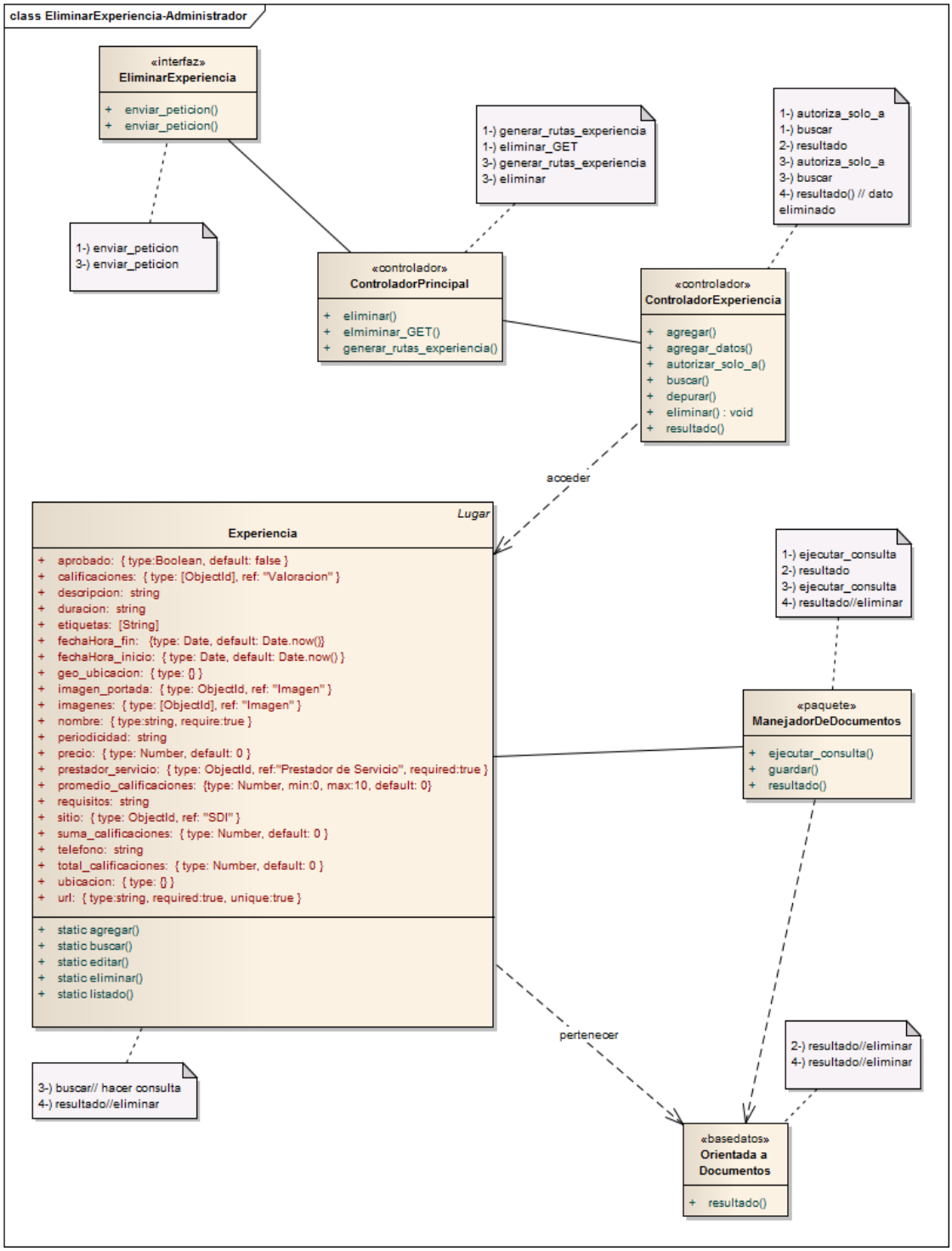


Figura Q. 2. Diagrama de Clases: Eliminar Experiencia por usuario Administrador.

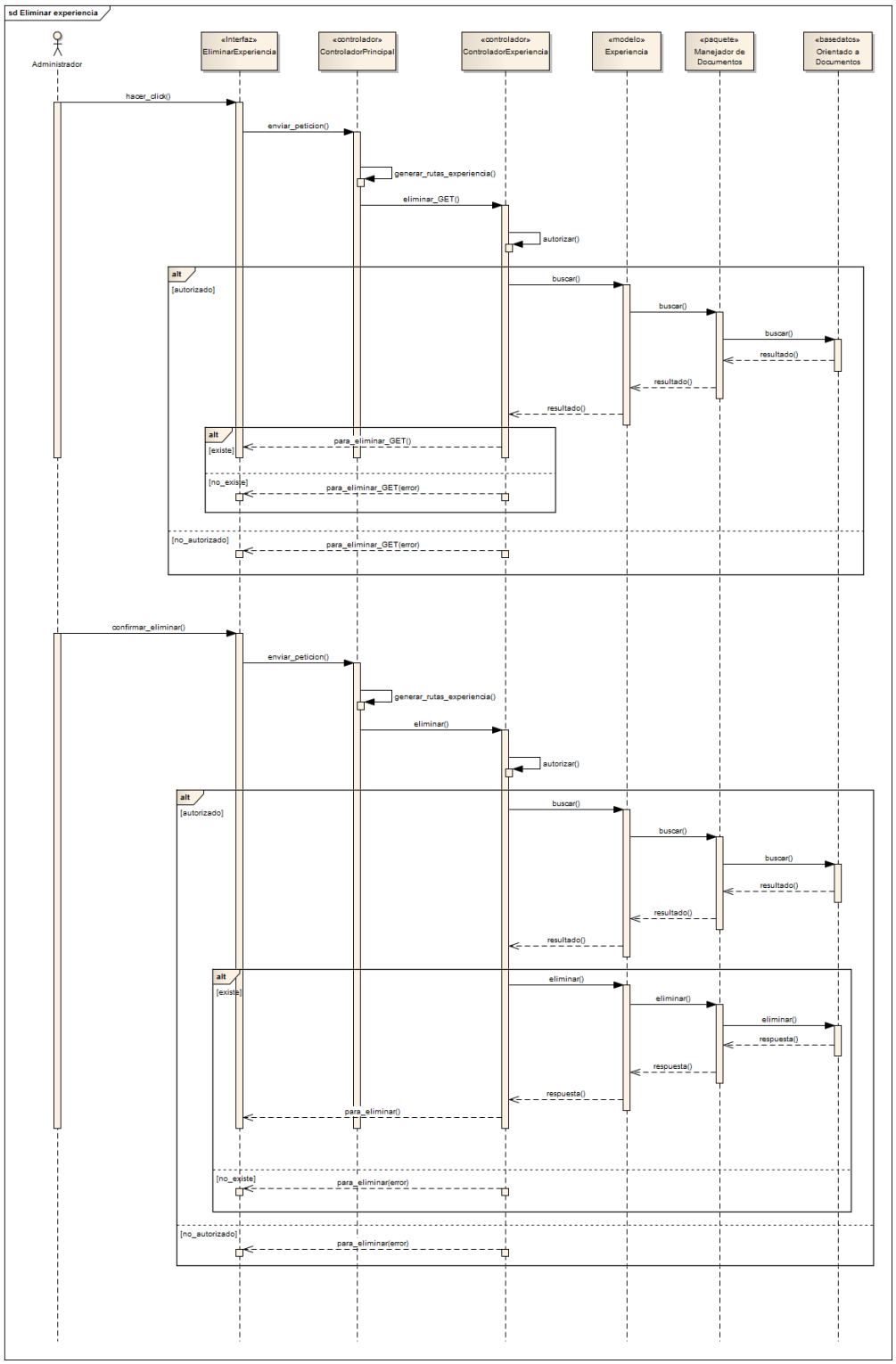


Figura Q. 3.Diagrama de secuencia: Eliminar Experiencia por usuario Administrador.

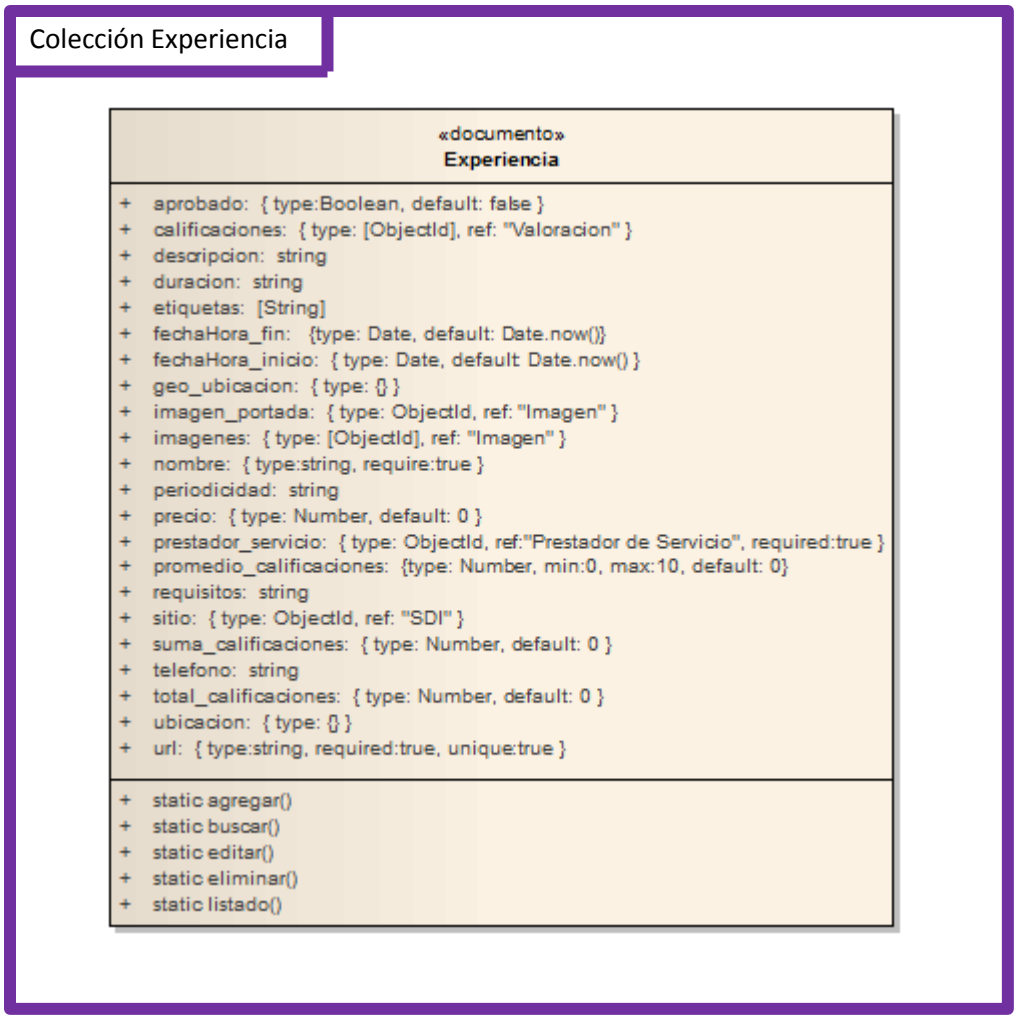


Figura Q. 4.Colección Experiencia: Eliminar Experiencia por usuario Administrador.

A continuación, se muestran el modelo de plantilla para el proceso de eliminar experiencia.

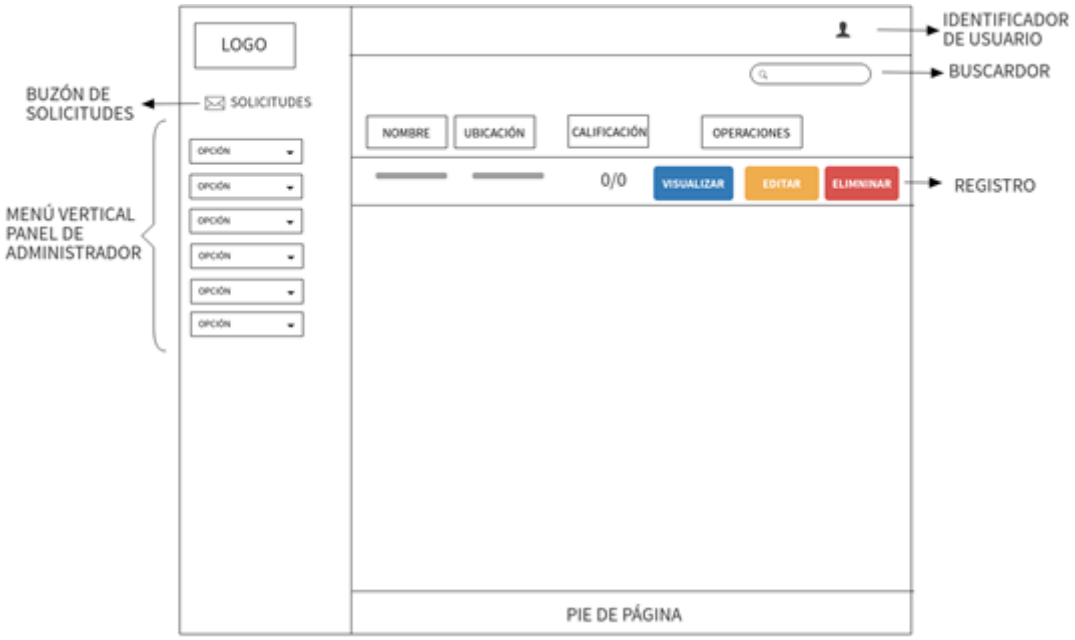


Figura Q. 5.Plantilla de interfaz 1: Eliminar Experiencia por usuario Administrador.

Figura Q. 6.Plantilla de interfaz 2: Eliminar Experiencia por usuario Administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Eliminar Experiencia y el sistema responde mostrando la interfaz de confirmación de acción.

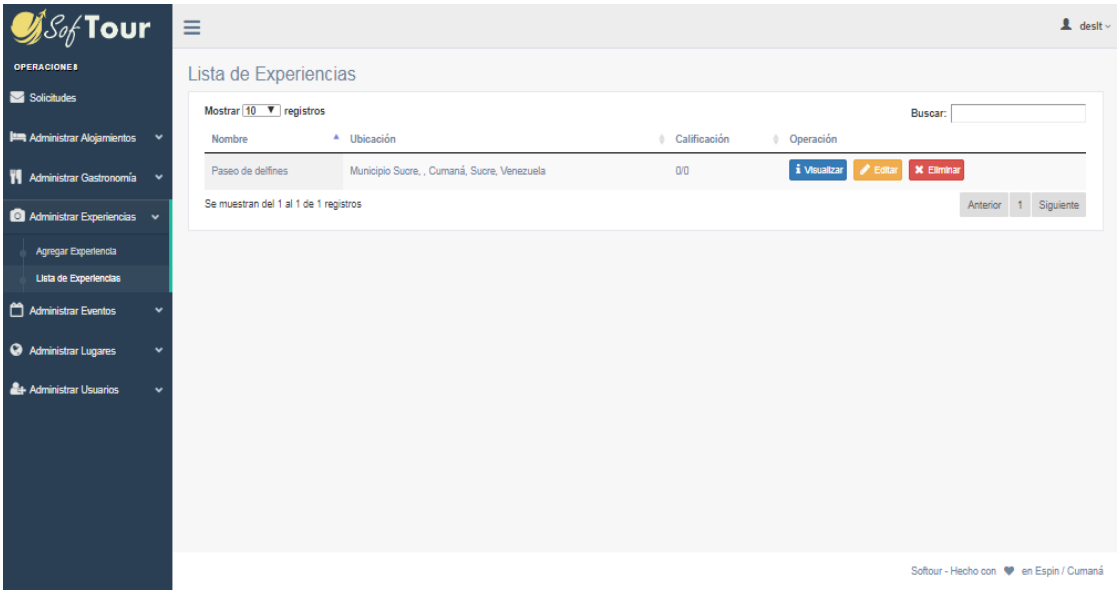


Figura Q. 7.Interfaz 1 de proceso de Eliminar Experiencia por usuario Administrador.

Figura Q. 8.Interfaz 2 de proceso de Eliminar Experiencia por usuario Administrador.

Parte del código desarrollado para la funcionalidad Eliminar Evento por usuario administrador

```
eliminar(req, res, next) {
    this._Experiencia.eliminar({"url": req.params.url.toLowerCase()}, (error, eliminado) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_eliminar(req, res);
        }
        if(!eliminado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": {"url": "listado"}};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_eliminar(req, res);
    });
}
```

Figura Q. 9.Código desarrollado para el proceso Eliminar Experiencia por usuario Administrador.

Prueba eliminar experiencia

Para continuar el proceso de pruebas se procede a someter a la técnica de caja negra al módulo Eliminar experiencia, la función de este módulo es eliminar de forma total una experiencia agregada a la base de datos del sistema.

La prueba se llevó a cabo, ubicándose en el listado de experiencias agregadas y procediendo a utiliza el botón eliminar, si la experiencia es eliminada, la función se cumple de forma satisfactoria.

A continuación, se muestran imágenes de una de las pruebas realizadas:

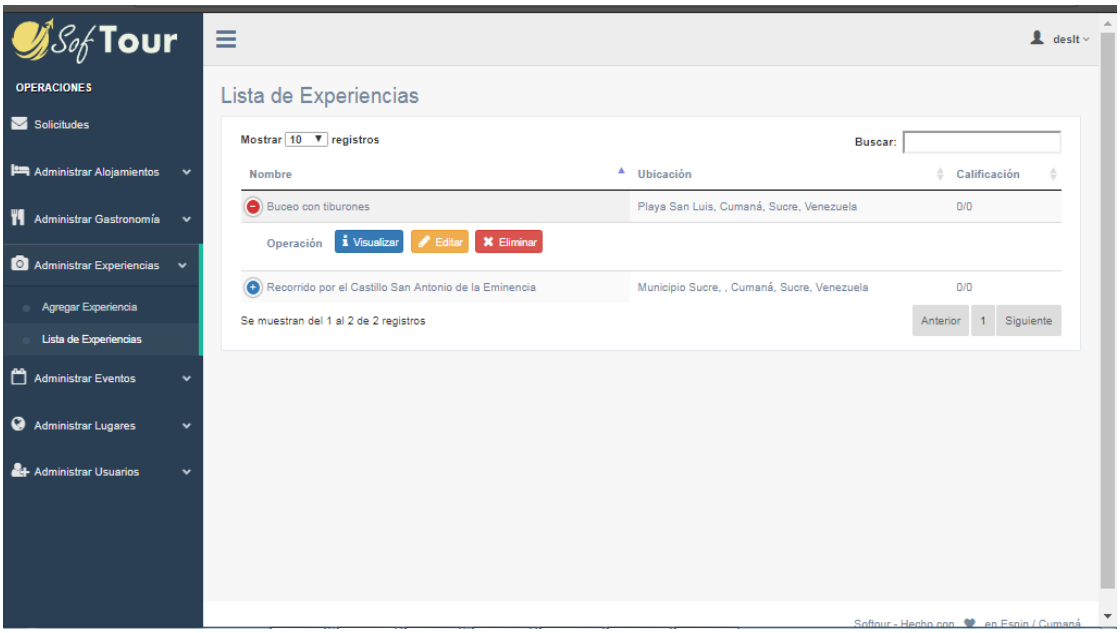


Figura Q. 10.Prueba Eliminar Experiencia por Administrador. Paso 1.

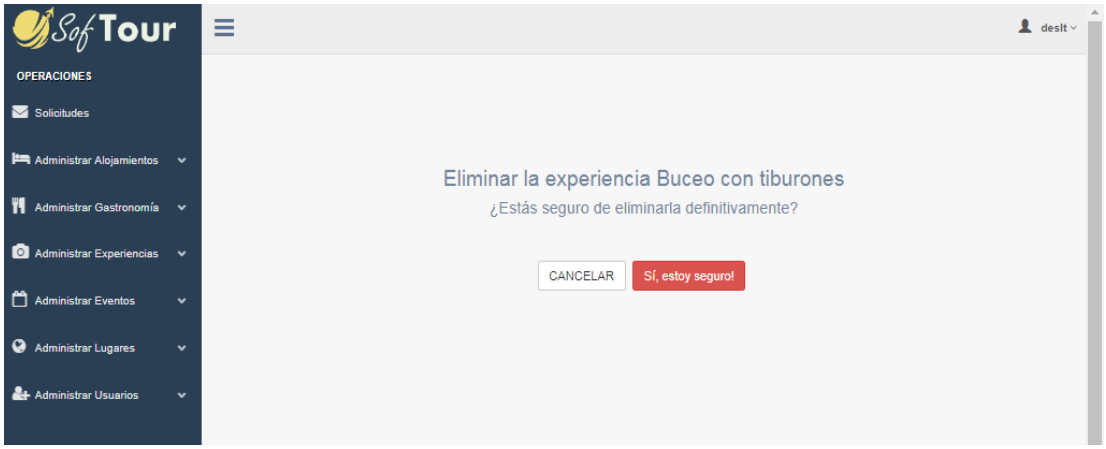


Figura Q. 11.Prueba Eliminar Experiencia por Administrador. Paso 2.

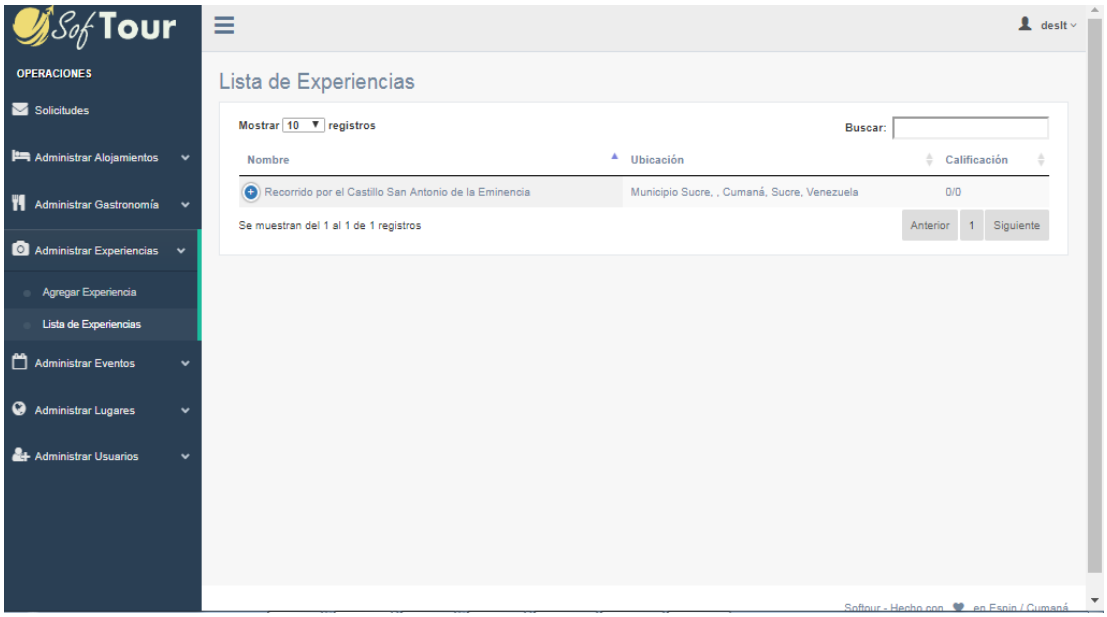


Figura Q. 12. Prueba Eliminar Experiencia por Administrador. Paso 3.

Conclusión de prueba

La función se ejecutó de forma correcta.

Prueba interfaz y usabilidad

La interfaz pertenece al conjunto de interfaz de usuario Administrador que se evaluó anteriormente, cumpliendo con estándares de usabilidad.

Apéndice R. Ciclo de Versión Administrador: Iteración Ingresar Alojamiento.

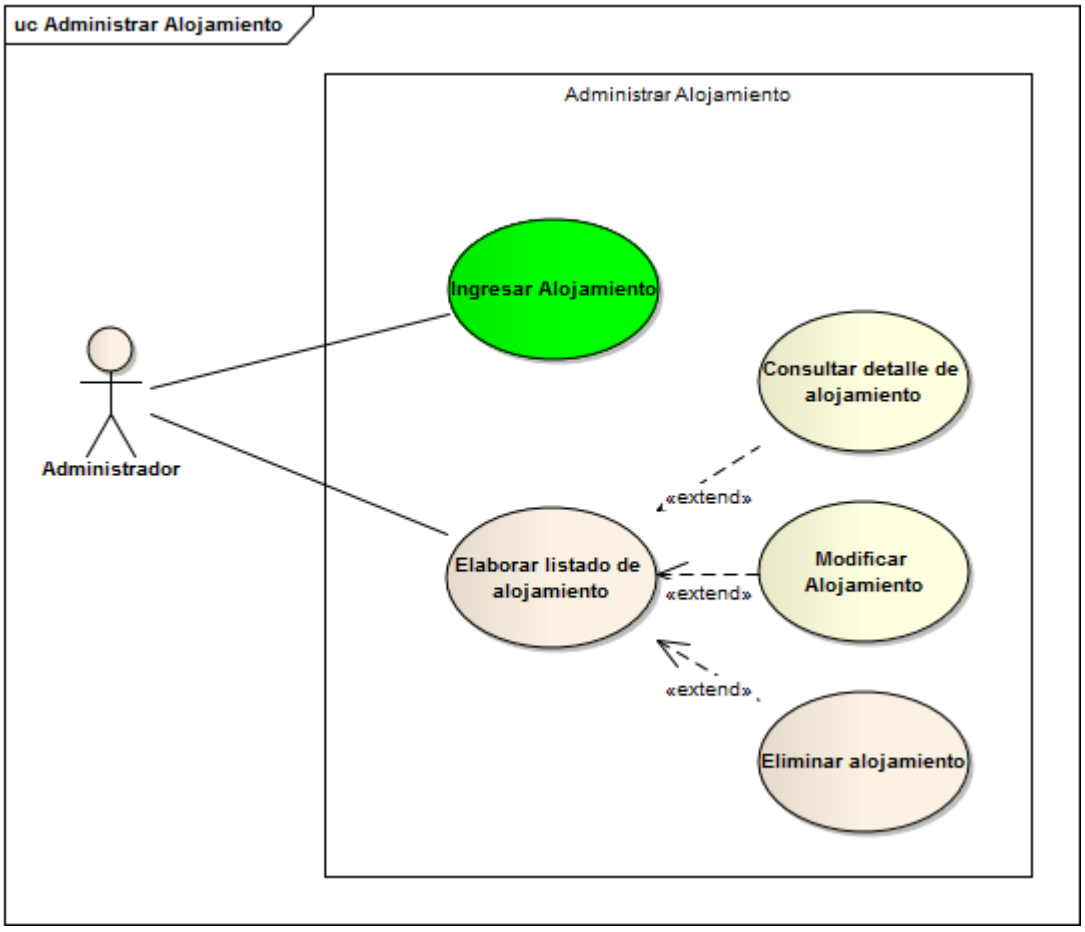


Figura R. 1.Caso de uso: Ingresar Alojamiento por Administrador.

Tabla R. 1. Descripción Caso de uso Ingresar Alojamiento por usuario Administrador.

Nombre del Caso de uso	Ingresar alojamiento.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en ingresar los atributos del alojamiento de manera satisfactoria.	
Precondición	Para ingresar un alojamiento, no debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Agregar alojamiento".
	2	El sistema muestra el formulario correspondiente.
	3	El Administrador ingresar los atributos del alojamiento.
	4	El sistema almacena los atributos del alojamiento y finaliza el caso de uso.
Postcondición	Se ingresa los atributos del alojamiento en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no ingresar el alojamiento y selecciona cancelar el caso de uso, ya sea seleccionando cualquier vista del sistema.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. Para que un alojamiento pueda ser ingresado no debe estar registrado en la base de datos del sistema.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

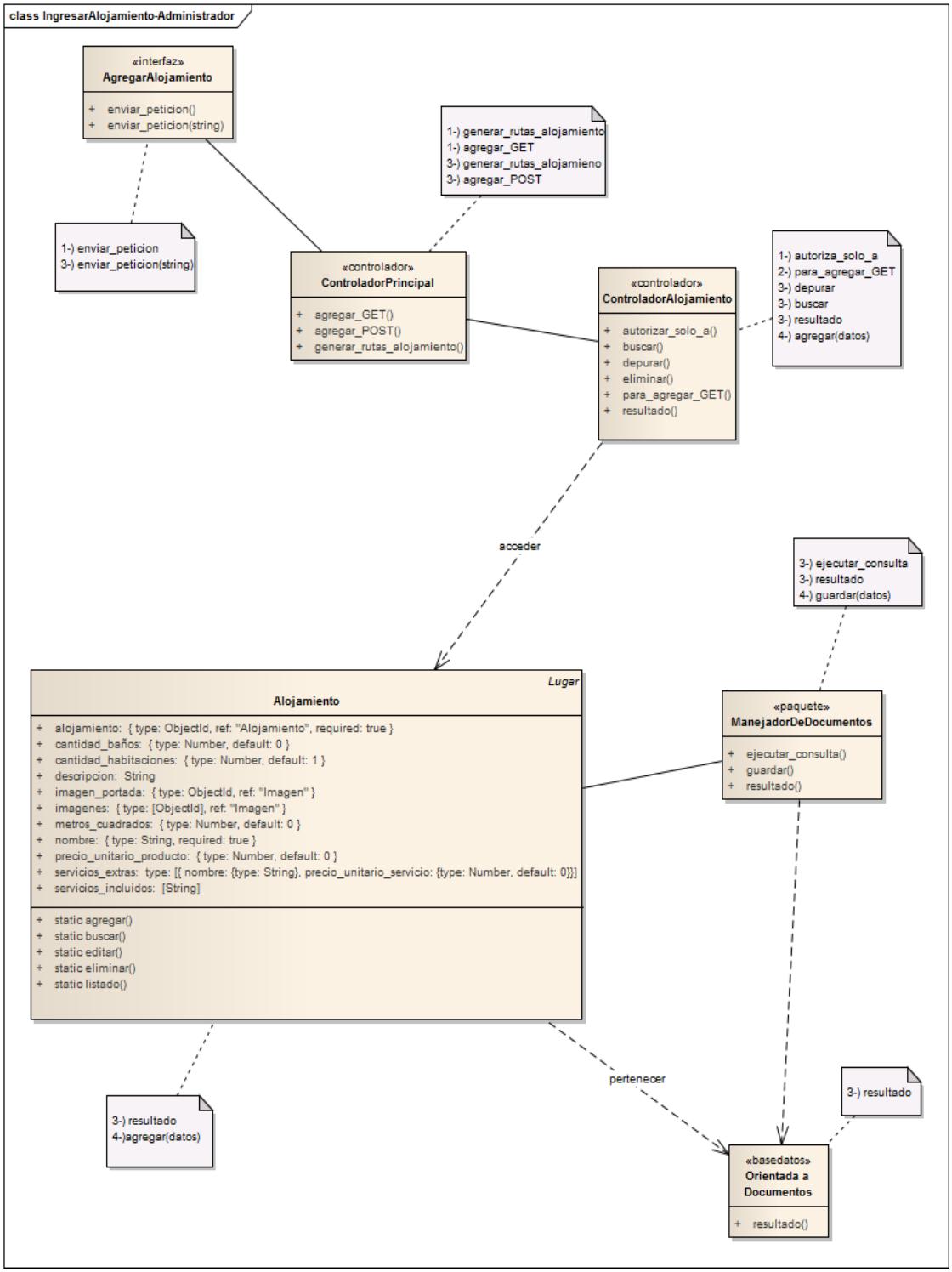


Figura R. 2.Diagrama de Clases: Ingresar Alojamiento por usuario Administrador.

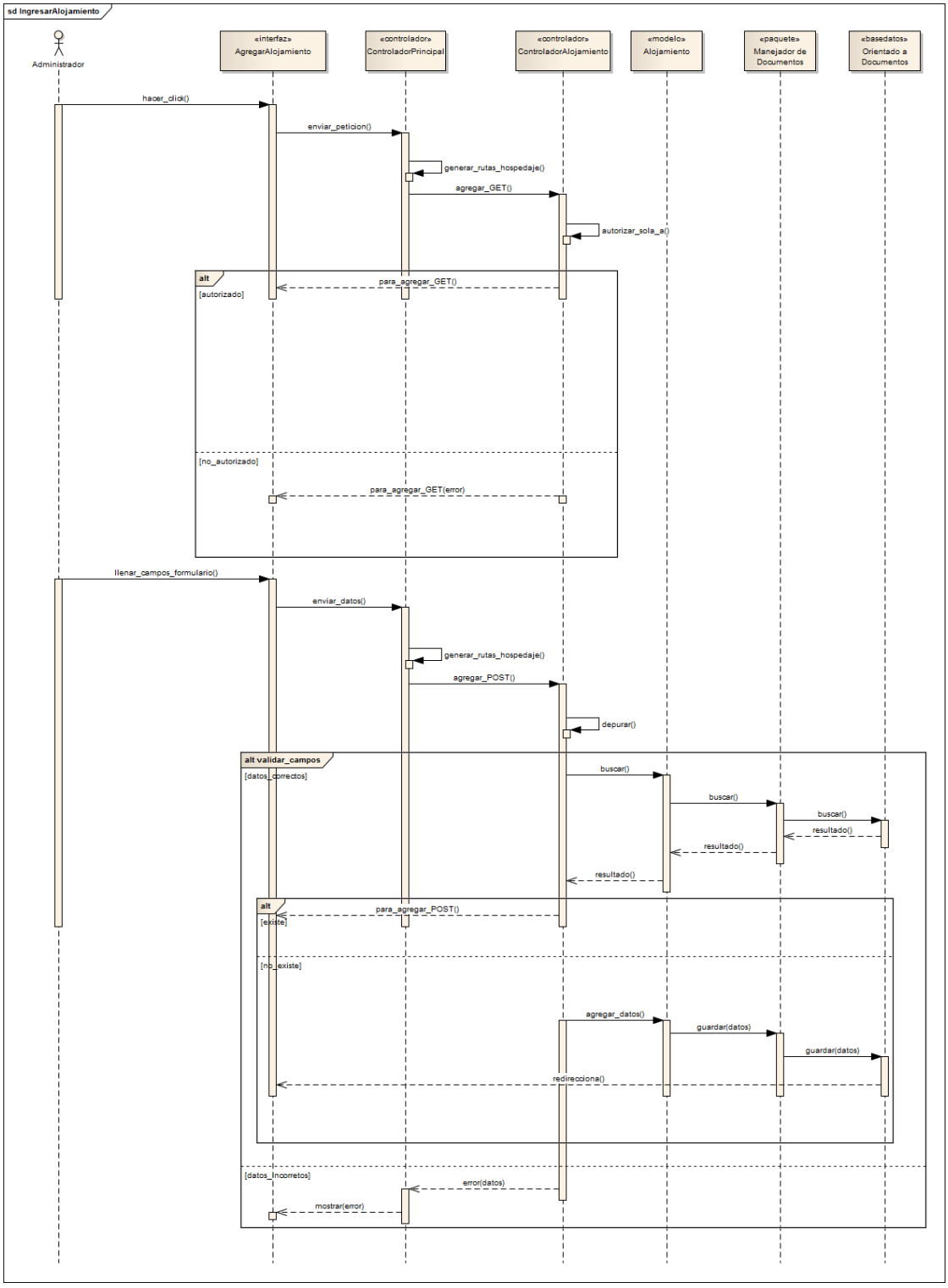


Figura R. 3.Diagrama de secuencia: Ingresar Alojamiento por usuario Administrador.

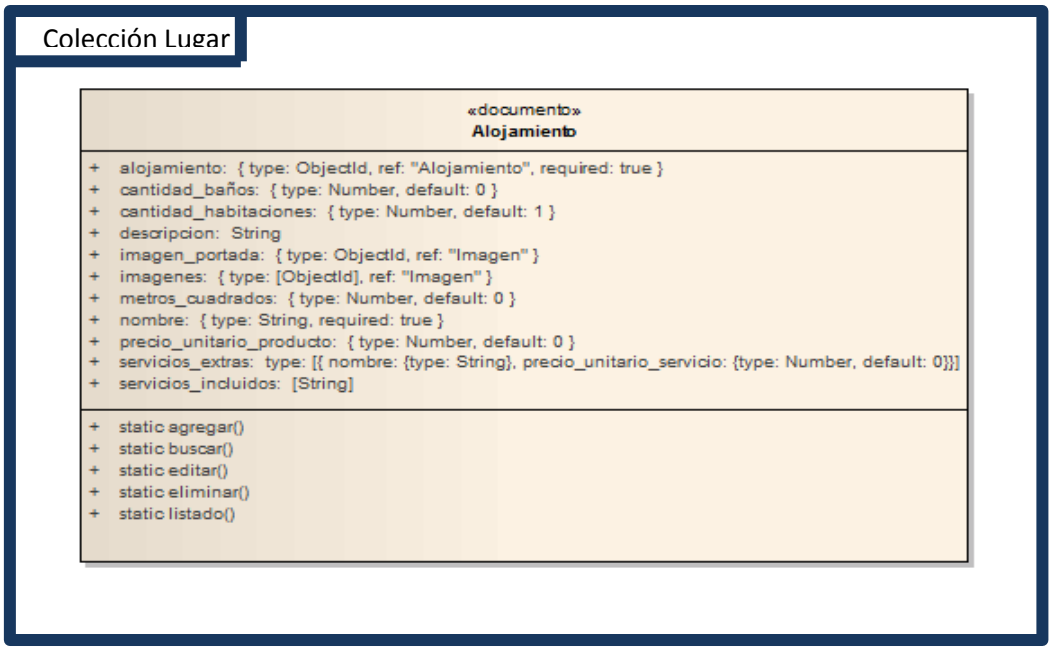


Figura R. 4. Colección Alojamiento 1: Ingresar Alojamiento por usuario Administrador.

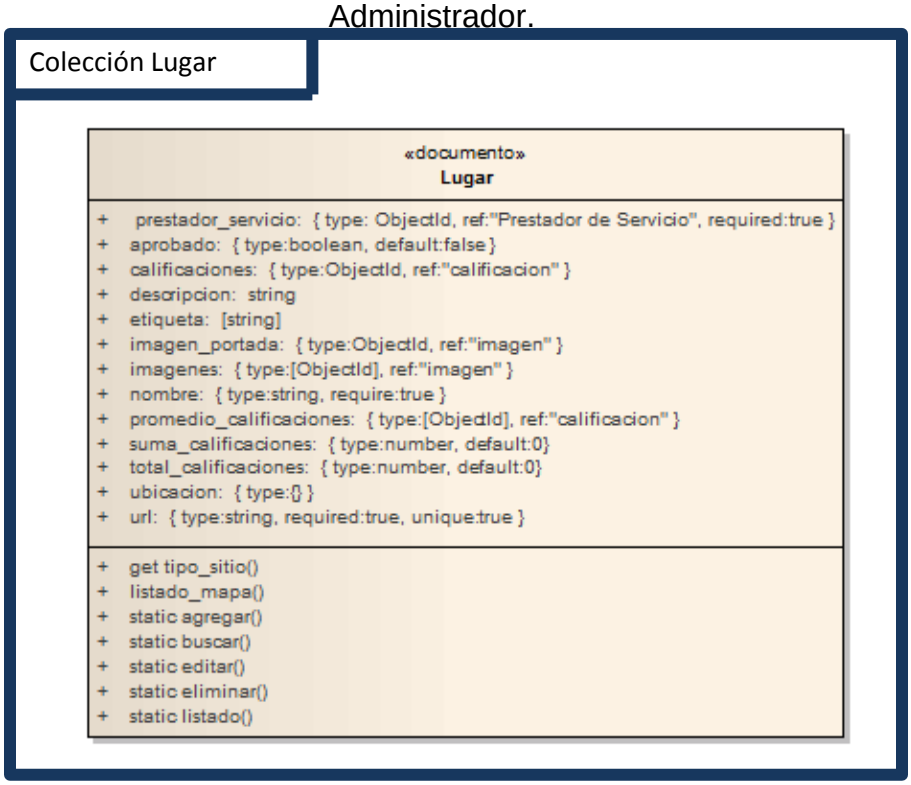


Figura R. 5. Colección Alojamiento 2: Ingresar Alojamiento por usuario Administrador.

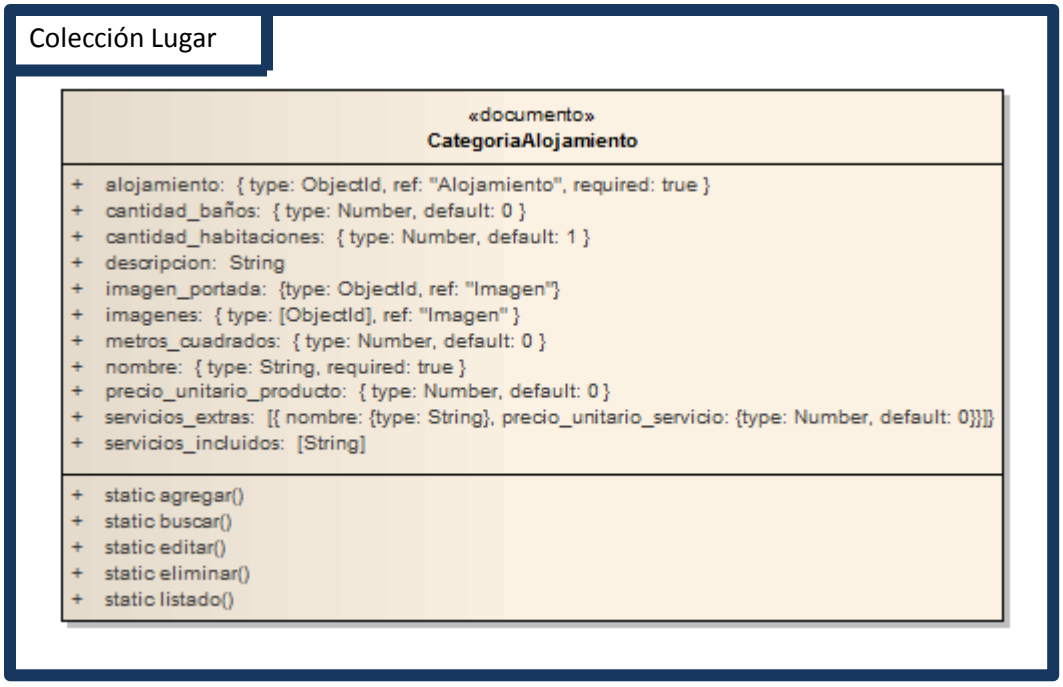


Figura R. 6. Colección Alojamiento 3: Ingresar Alojamiento por usuario Administrador.

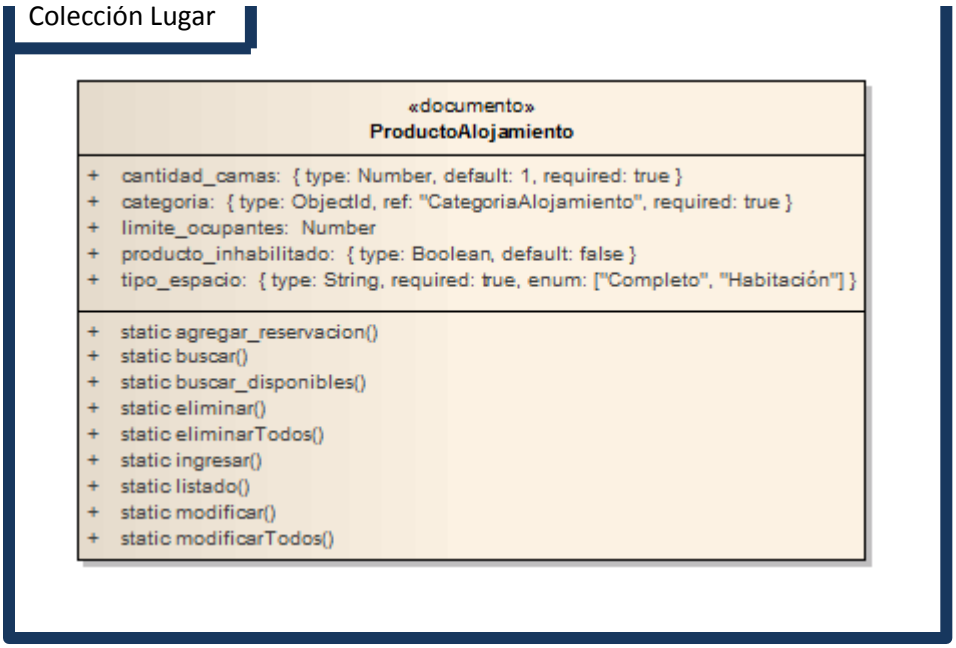


Figura R. 7. Colección Alojamiento 4: Ingresar Alojamiento por usuario Administrador.

A continuación, se muestran los modelos de plantillas para el proceso de Ingresar Alojamiento.

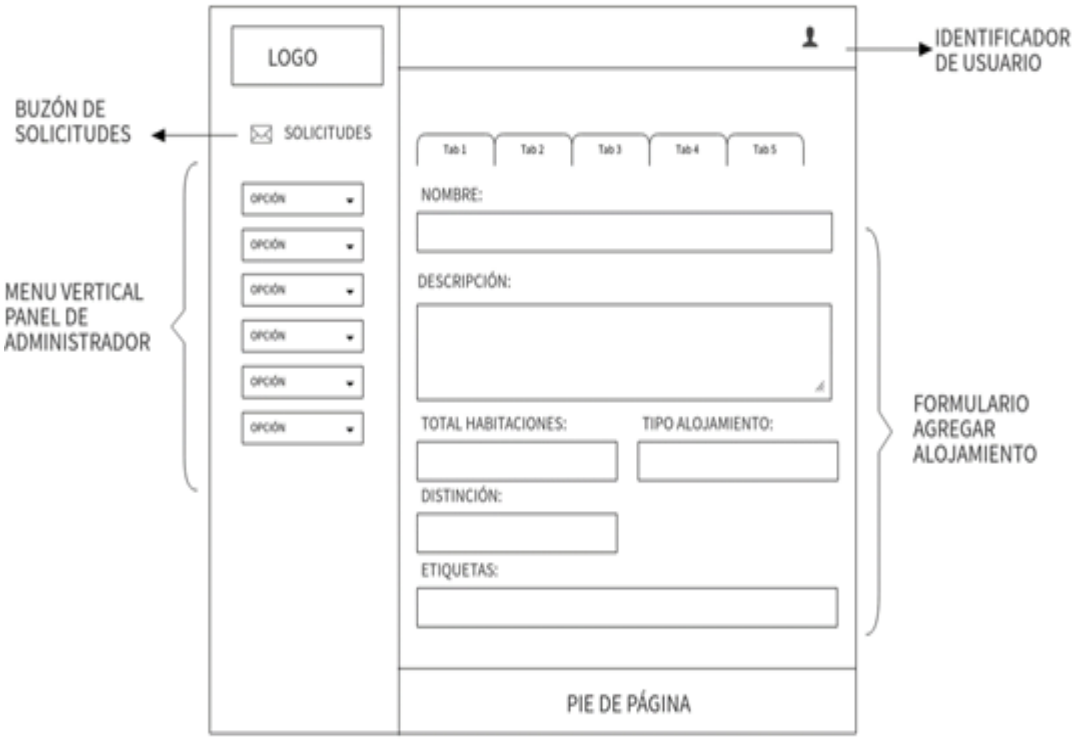


Figura R. 8.Plantilla de interfaz 1: Ingresar Alojamiento por usuario Administrador.

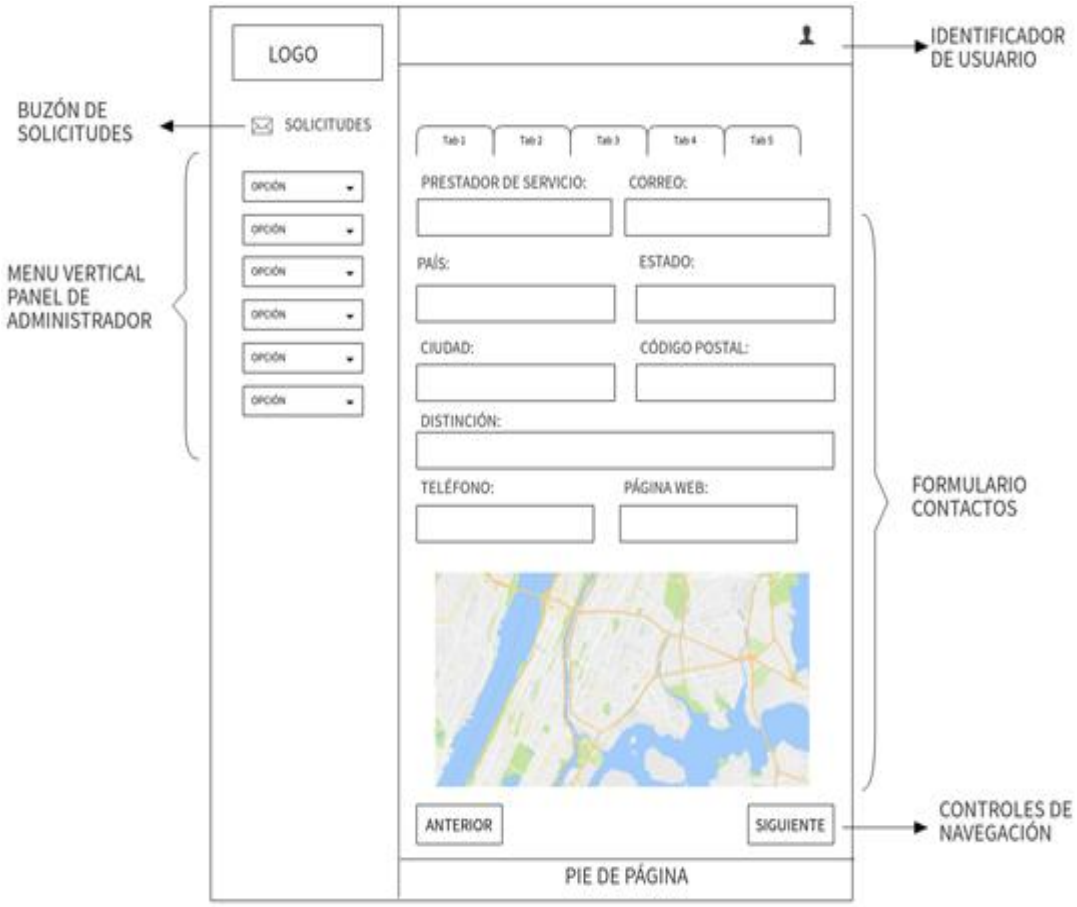


Figura R. 9.Plantilla de interfaz 2: Ingresar Alojamiento por usuario Administrador.

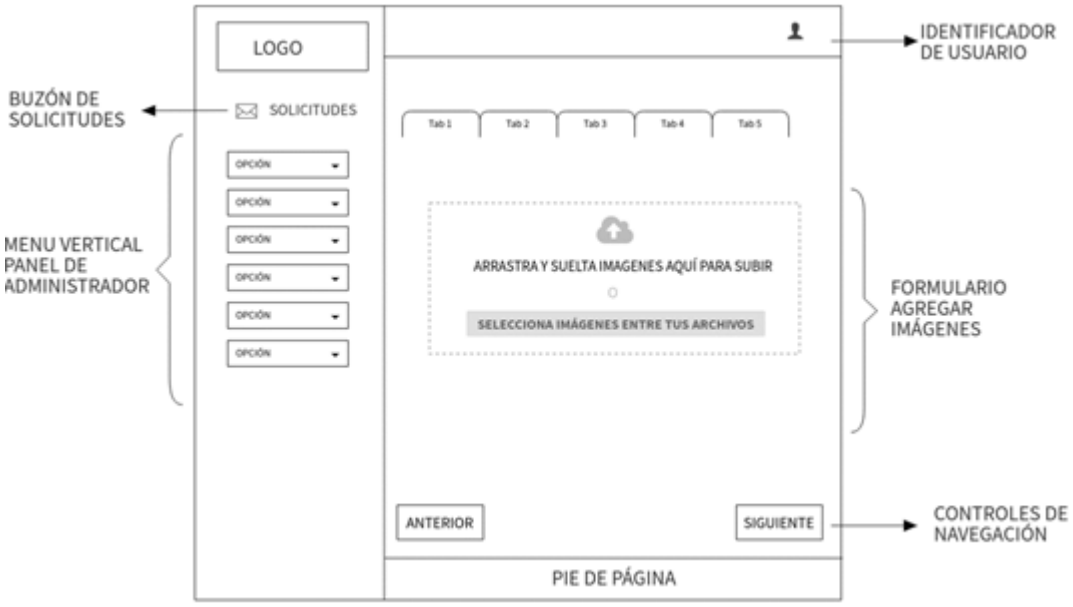


Figura R. 10.Plantilla de interfaz 3: Ingresar Alojamiento por usuario Administrador.

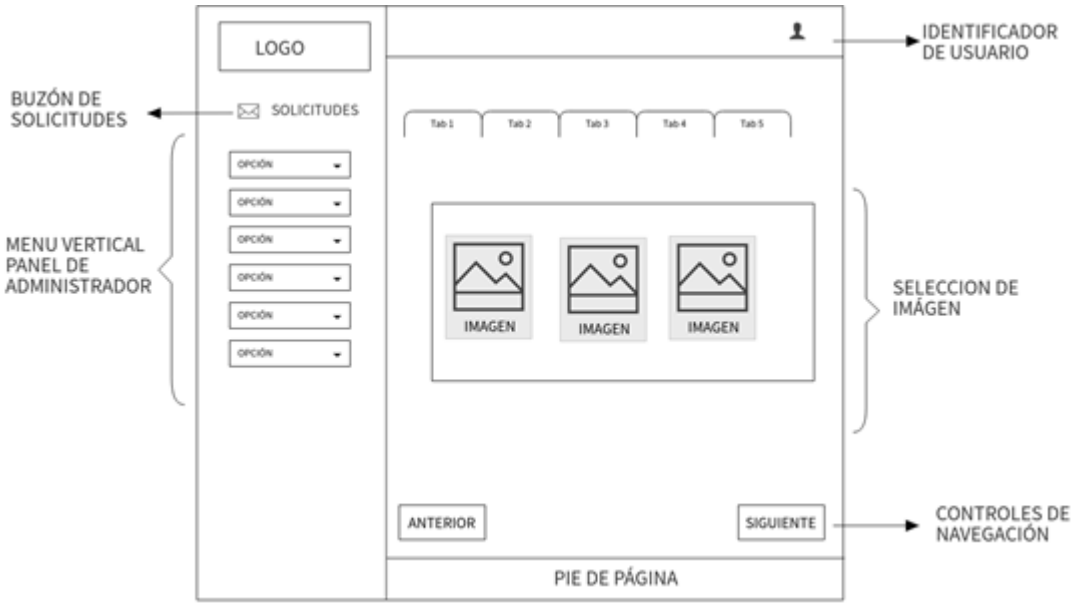


Figura R. 11.Plantilla de interfaz 4: Ingresar Alojamiento por usuario Administrador.

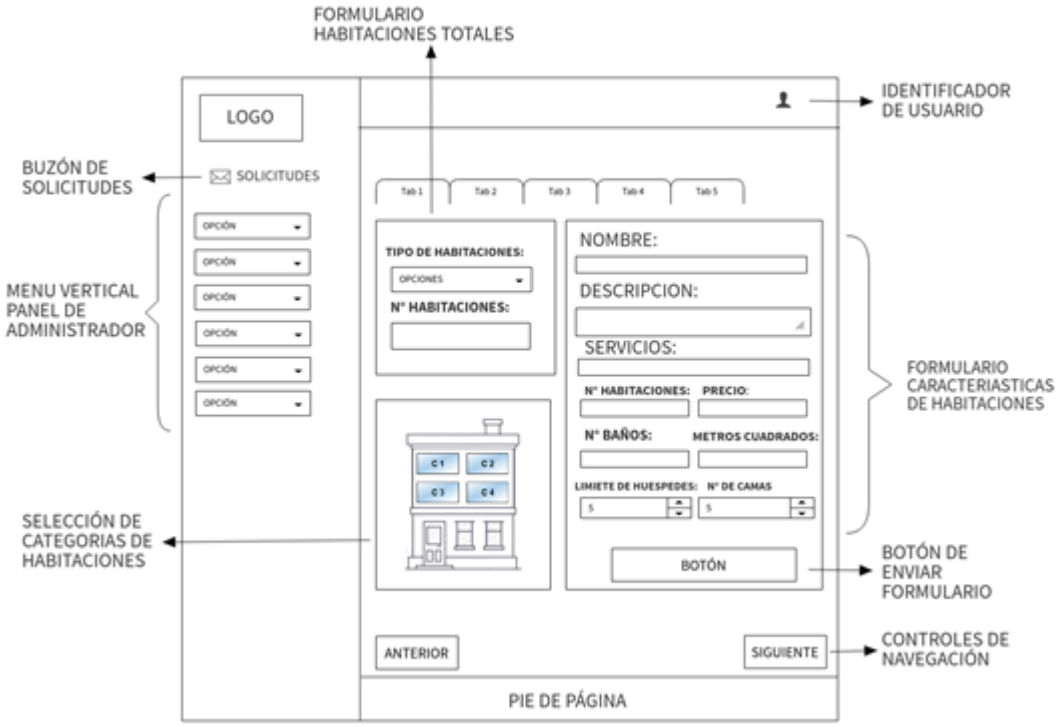


Figura R. 12.Plantilla de interfaz 5: Ingresar Alojamiento por usuario Administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Agregar Alojamiento y el sistema responde mostrando el formulario correspondiente.

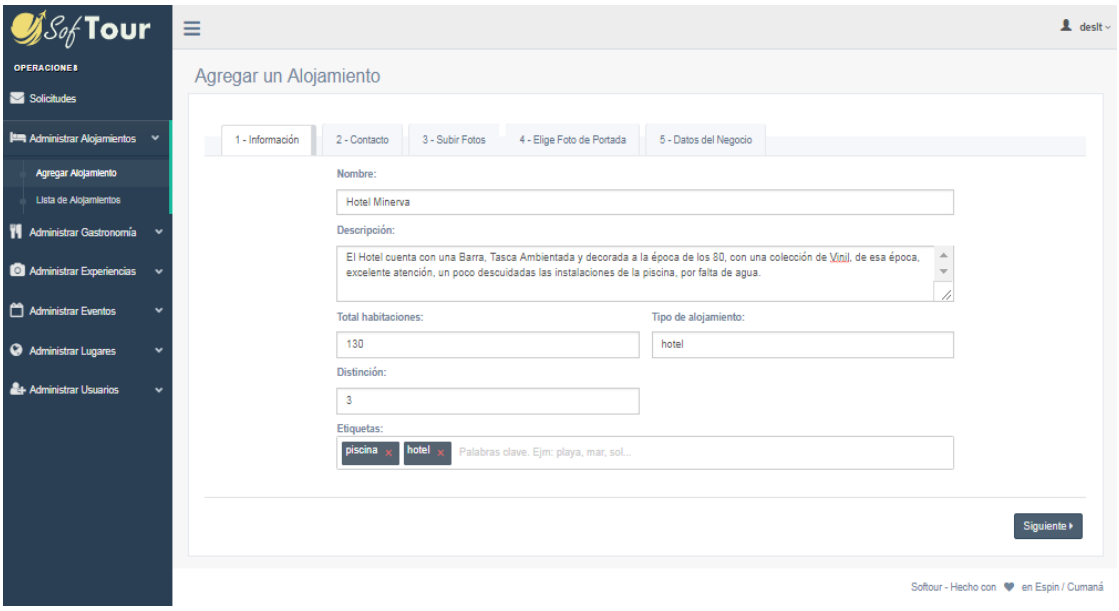


Figura R. 13.Interfaz 1 de proceso de Agregar Alojamiento por usuario Administrador.

El administrador ingresa los atributos del lugar, lo cual se visualiza en diferentes pestañas: Información, Contacto, Subir fotos, Elige foto de portada y Datos del negocio.

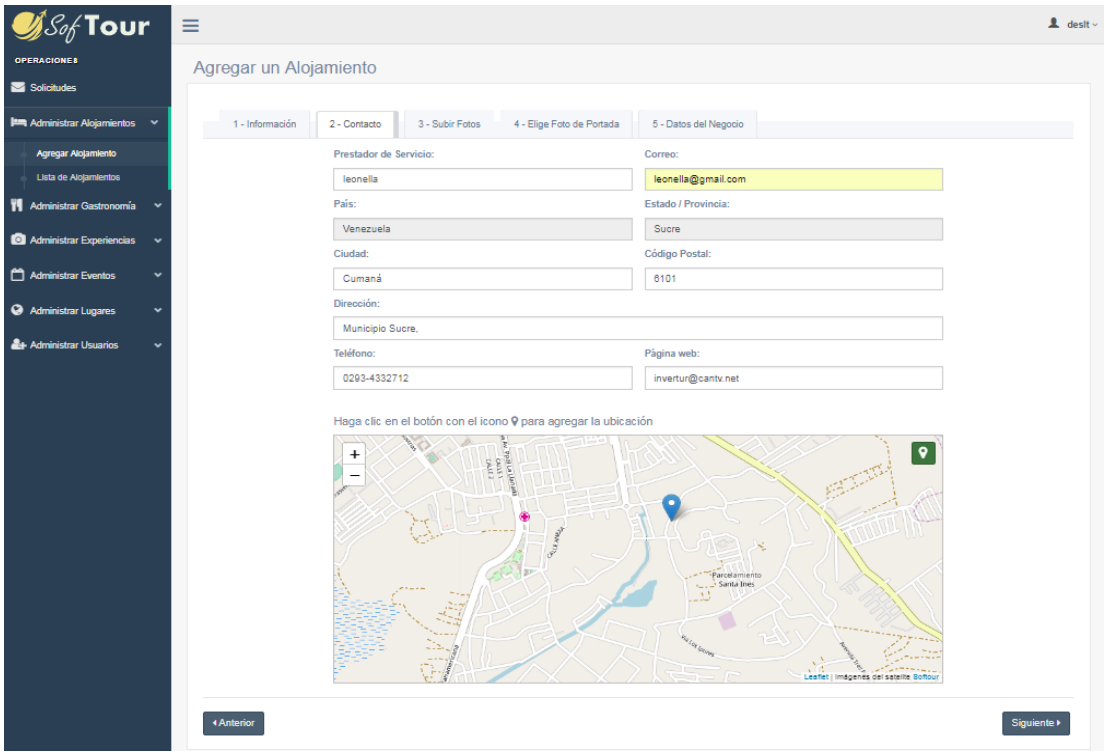


Figura R. 14.Interfaz 2 de proceso de Agregar Alojamiento por usuario Administrador.

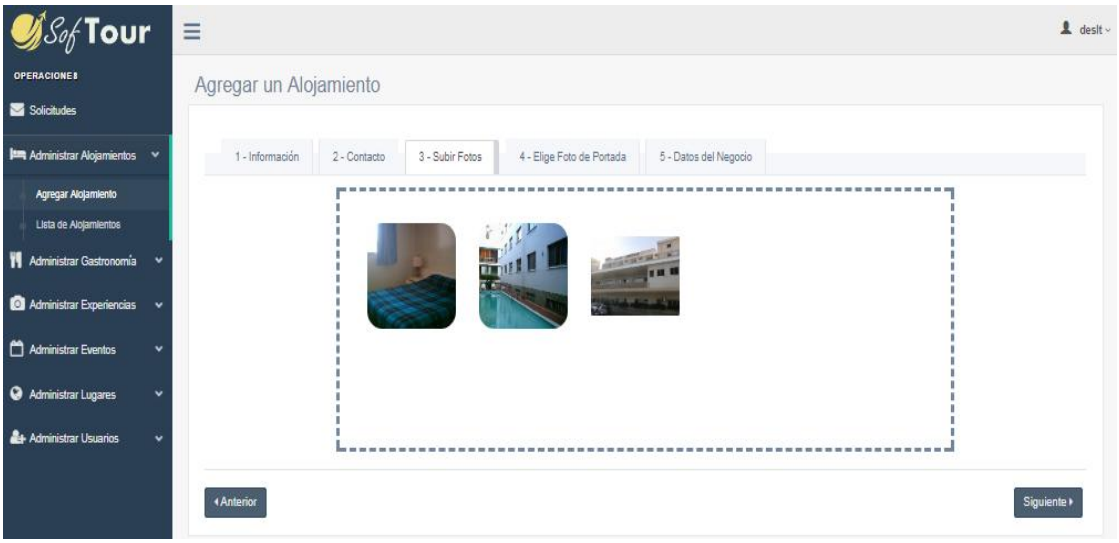


Figura R. 15.Interfaz 3 de proceso de Agregar Alojamiento por usuario Administrador.

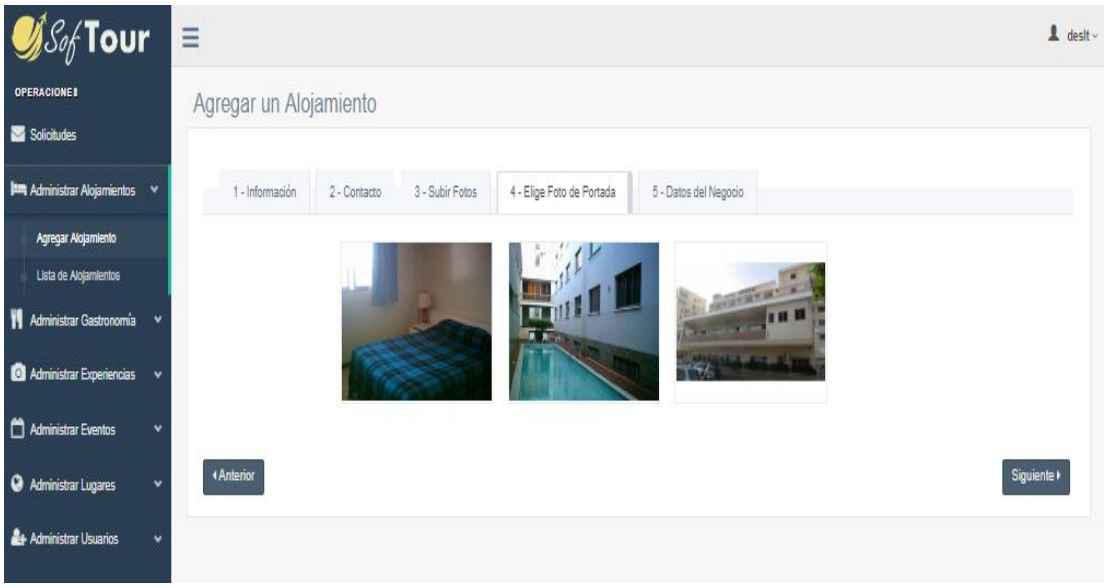


Figura R. 16.Interfaz 4 de proceso de Agregar Alojamiento por usuario Administrador.

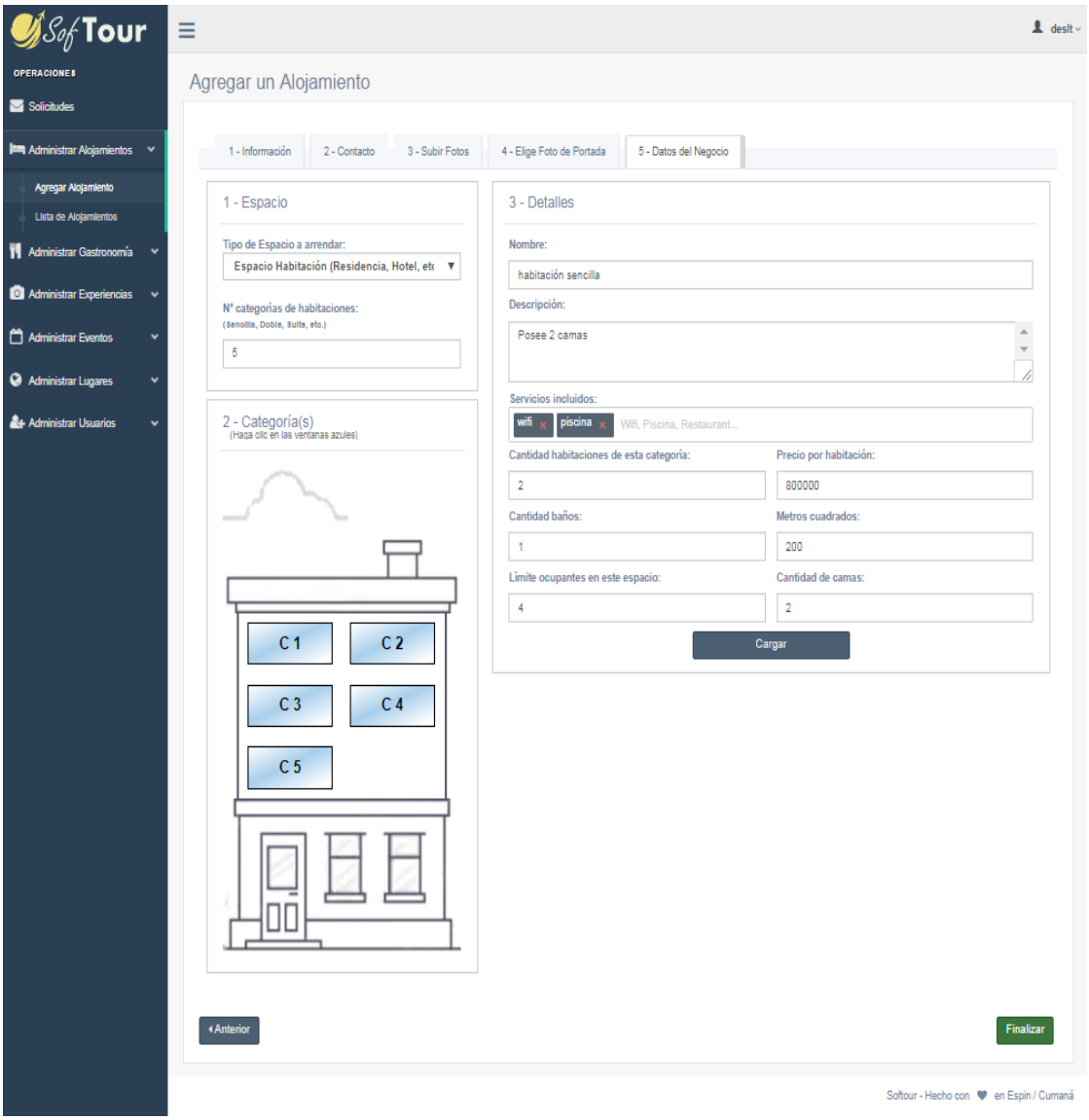


Figura R. 17.Interfaz 5 de proceso de Agregar Alojamiento por usuario Administrador.

Código desarrollado para la funcionalidad Agregar Alojamiento por usuario Administrador.

```
function constructorVentanas(e) { // una ventana es un form de una categoria
    let ventanas = [],
        forms_categ = [];
    $('#contenedor-categorias').empty();

    $('#categoria-habitaciones').empty();
    for (let i = 1; i <= $('input[name="cantidad_categorias"]').val(); i++) {
        ventanas.push('<a href="#detalles" class="ventana-categoria col-md-5 col-sm-5 col-lg-5" id="' + i + '">C ' + i + '</a>');
        forms_categ.push('<form class="form-horizontal clase-formulario-categorias" id="C' + i + '">' + form_espacio_habitacion + '</
            form>');
    };
    $('#categoria-habitaciones').append(ventanas);
    $('#contenedor-categorias').append(forms_categ);
    inicializarEtiquetador(".input-etiquetas", "Wifi, Piscina, Restaurant...");

    // mostrar solo el form q quiero ver
    $('.ventana-categoria').click(function(e) {
        $(".clase-formulario-categorias").hide();
        $(".clase-formulario-categorias").filter("[id=C" + $(this).attr("id") + "]").show();
    });

    // cargar datos al pulsar el btn
    $('.btn-guardar-categoria').click(function(e) {
        e.preventDefault();
        let contenedor = $(this).parent(),
            f = $(contenedor).attr("id").slice(1);
        $(contenedor).parsley({excluded: 'a, input[type=button], input[type=submit], input[type=reset]', inputs: 'input, textarea,
            select, input[type=hidden], :hidden'})
            .whenValidate()
            .done(() => {
                $(contenedor).find("input, textarea, .input-etiquetas").attr("readonly", "readonly");
                let tagsinput = $(contenedor).find("[name='servicios_incluidos']").siblings();

                $(tagsinput).css("background-color", "#eee");
                $(tagsinput).find("div").css("background-color", "#eee");
                $(tagsinput).find(".tag a").hide();
                $(this).attr("disabled", "disabled");

                new PNotify({
                    title: 'Cargado',
                    text: 'Ingrese siguiente categoría',
                    styling: "bootstrap3",
                    delay: 10000,
                    type: "success",
                    sticker: false
                });

                lista_categorias[parseInt(f) - 1] = $(contenedor).serialize() + "&id_alojamiento=" + id_alojamiento;
            });
    });
}

$('select[name="tipo_espacio"]')
    .ready(function(e) {
        lista_categorias = [];
```

Figura R. 18. Código desarrollado para el proceso Ingresar Alojamiento por usuario Administrador.

Prueba agregar alojamiento

Siguiendo con lo estipulado para el desarrollo de pruebas se continuó utilizando

la técnica de caja negra, la función registrar alojamientos tiene como finalidad permitirle al usuario Administrador agregar cabañas, hoteles y posadas a la base de datos del sistema.

A continuación, se muestran los valores y parámetros a los cuales se ajusta la función

Tabla R. 2. Clases de equivalencia Agregar Alojamiento por Administrador.

Condición de entrada	Tipo	Clases de equivalencias válidas	Clases de equivalencias no válidas
Nombre	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Descripción	Alfanumérico	1:Cadena de caracteres, números.	2: Campo en blanco
Total habitaciones	Numérico	3: números	2: Campo en blanco
Tipo alojamiento	Alfabético	4: palabras	2: Campo en blanco
Distinción	Numérico	3: números	2: Campo en blanco
Etiquetas	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Prestador de servicio	Selección		2: Campo en blanco
Correo	Correo	3: símbolos, caracteres, números	2: Campo en blanco 4: formato invalido
País	Alfabético	1: Cadena de caracteres	5: Símbolos 6:números
Estado/provincia	Alfabético	1: Cadena de caracteres	5: Símbolos 6:números
Ciudad	Alfabético	1: Cadena de caracteres	5: Símbolos 6:números
Código postal	Numérico	6: números	1: Cadena de caracteres
Dirección	Alfanumérico	1: Cadena de	2: Campo en

		caracteres, números.	blanco
Teléfono	Numérico	6: números	1: Cadena de caracteres
Página web	Alfanumérico	1: Cadena de caracteres, números.	
Selección mapa	selección		
Subir foto	Imagen	7: JPG, GIF, PNG	2: Campo en blanco
Seleccionar foto	selección		
Tipo espacio a arrendar	selección		
N° de categorías de habitaciones	Numérico	6: números	
Nombre*	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Descripción	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Servicios incluidos	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Cantidad de habitaciones de esta categoría	Numérico	6: números	
Precio de habitación	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Cantidad de baños	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Metros cuadrados	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Límites de ocupantes en este espacio	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco
Cantidad de camas	Alfanumérico	1: Cadena de caracteres, números.	2: Campo en blanco

Tabla R. 3. Casos de prueba Agregar Alojamiento por Administrador.

Condición de entrada	A1	A2	A3
Nombre	Los ángeles	Ciutat	caribe
Descripción	El Hotel dispone de seis piscinas, una cancha de petanca y tres de tenis, cinco bares y tres restaurantes. Asimismo, se conecta directamente con la prestigiosa playa El Agua. Se encuentra igualmente a tan solo 35 kilómetros del aeropuerto Santiago Mariño y a 25 de los principales centros comerciales y restaurantes de la isla.		El Hotel dispone de habitaciones estandar, ejecutivas y suites dónde la comodidad y el buen gusto se unen para dar a los clientes un servicio de calidad. Cuenta con 2 habitaciones adaptadas para discapacitados.
Total habitaciones	4	5	2
Tipo alojamiento	Hotel	Hotel	hotel
Distinción	3	2	1
Etiquetas	Hotel; viajes;	Hotel; viajes;	Hotel; viajes;
Prestador de servicio	selección	selección	selección
Correo	Ang@hotmail.com	hotel@gmail.com	hotel@caribe.com
País	Venezuela	Mexico	Venezuela
Estado/provincia	Sucre	Sucre	Sucre
Ciudad	Cumaná	Cumaná	Cumaná
Código postal	6101	ljghf	2301
Dirección	Av. Perimetral	Cantarrana	

Teléfono	029345124	0293446525	0293445522
Página web	Hoteles.es	Hotelc.com	
Selección mapa	Selección	Selección	
Subir foto	JPG,PNG	PDF	
Seleccionar foto	Selección	Selección	selección
Tipo espacio a arrendar	Hotel	Hotel	hotel
N° de categorías de habitaciones	1	2	2
Nombre*	doble	Sencilla	Doble
Descripción	Cuando te hospedas en un hotel, uno de los elementos a los que tienes que poner especial atención es el tipo de habitación que vas a escoger.		Habitación doble es con dos camas o una cama matrimonial o King size, para dos personas, no más.
Servicios incluidos	Wifi; nevera; T.V.	Jacuzzi	
Cantidad de habitaciones de esta categoría	5	1	1
Precio de habitación	500000 Bs	400 Bs	1000000Bs
Cantidad de baños	1	1	2
Metros cuadrados	40	20	Cincuenta
Límites de ocupantes en este espacio	4	2	4
Cantidad de camas	2	1	2
Clases Equivalencia	1,3,4,5,6,7	1,2,3,4,5	1,2,3,4
Resultado	Registro satisfactorio	Registro no satisfactorio	Registro no satisfactorio

Para el caso de pruebas A1 se procede a llenar el formulario con los datos planteados, para este caso no se encontraron fallos ni inconvenientes, de tal manera que el registro básico se cumplió de forma exitosa como se muestra en las siguientes imágenes secuencia del proceso de prueba realizado:

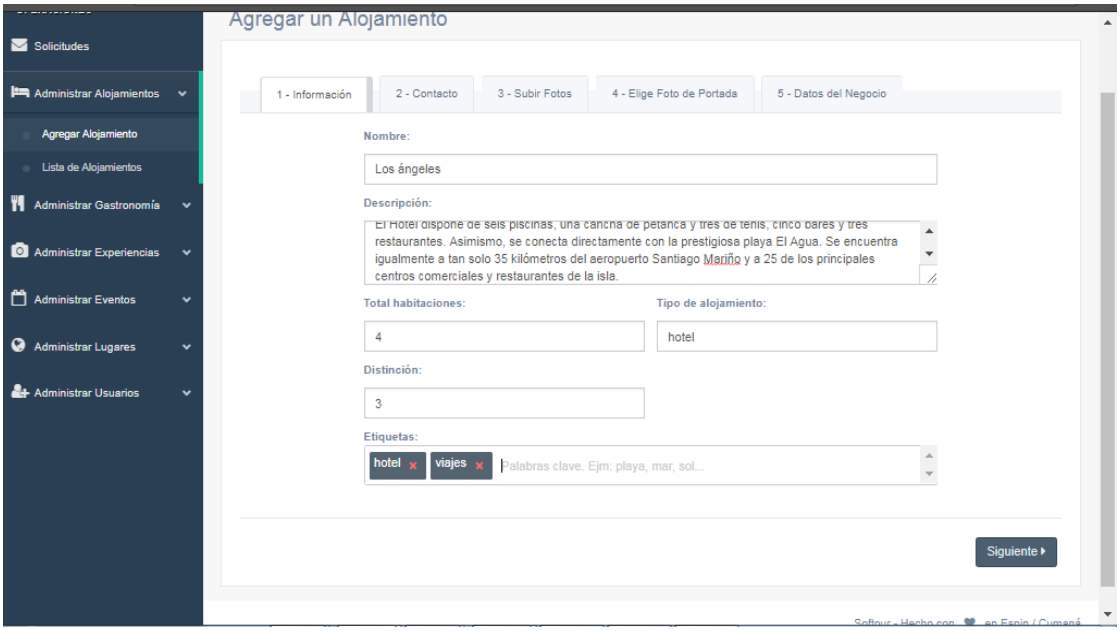


Figura R. 19.Caso de prueba A1 agregar alojamiento por Administrador. Paso 1.

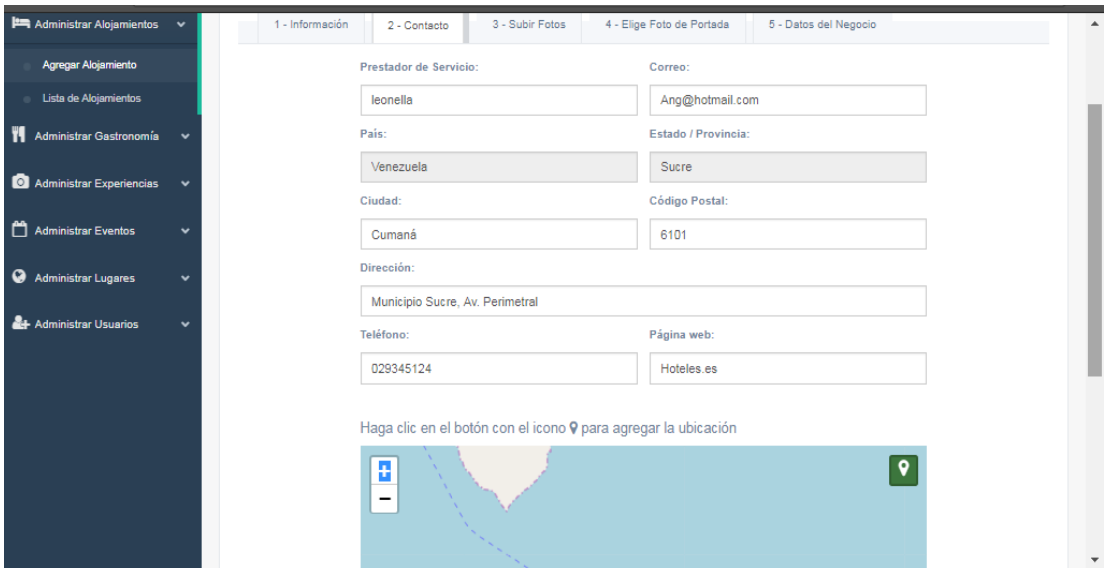


Figura R. 20.Caso de prueba A1 agregar alojamiento por Administrador. Paso 2.

Figura R. 21.Caso de prueba A1 agregar alojamiento por administrador. Paso 3

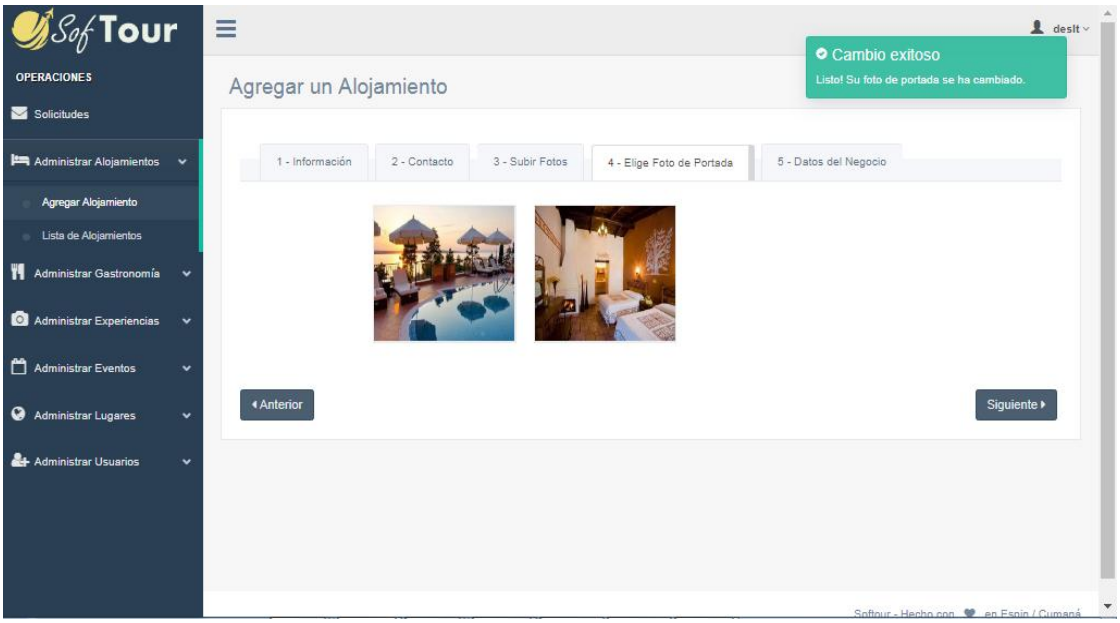


Figura R. 22.Caso de prueba A1 agregar alojamiento por administrador. Paso 4

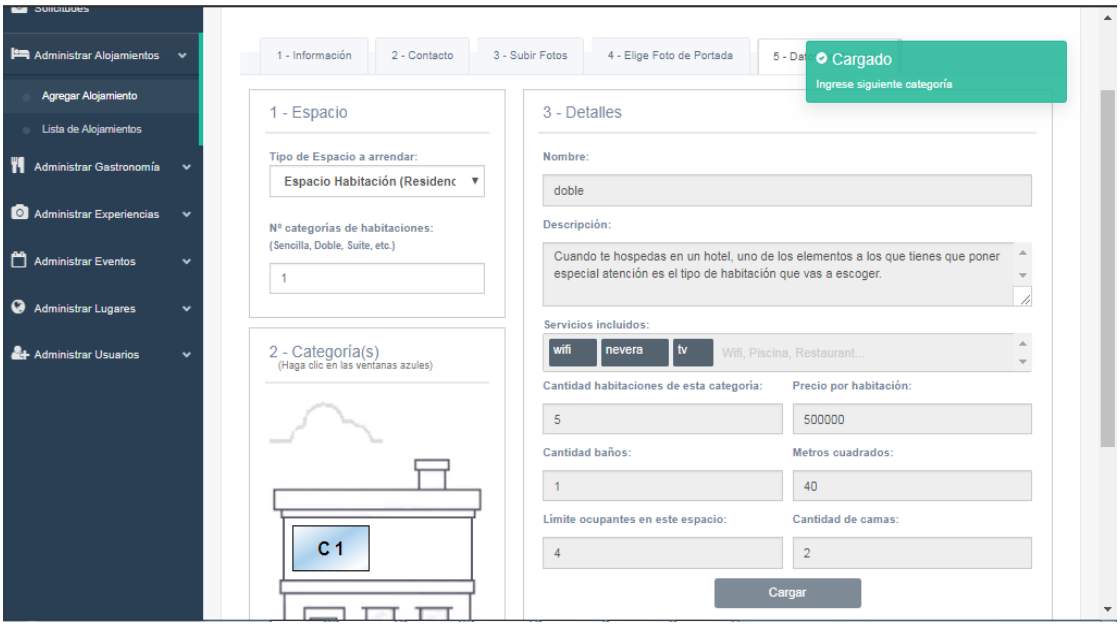


Figura R. 23.Caso de prueba A1 agregar alojamiento por administrador. Paso 5

Para el caso de pruebas A2 de Agregar Alojamiento se planteó establecer otro país, pero el sistema no permite este tipo de cambios.

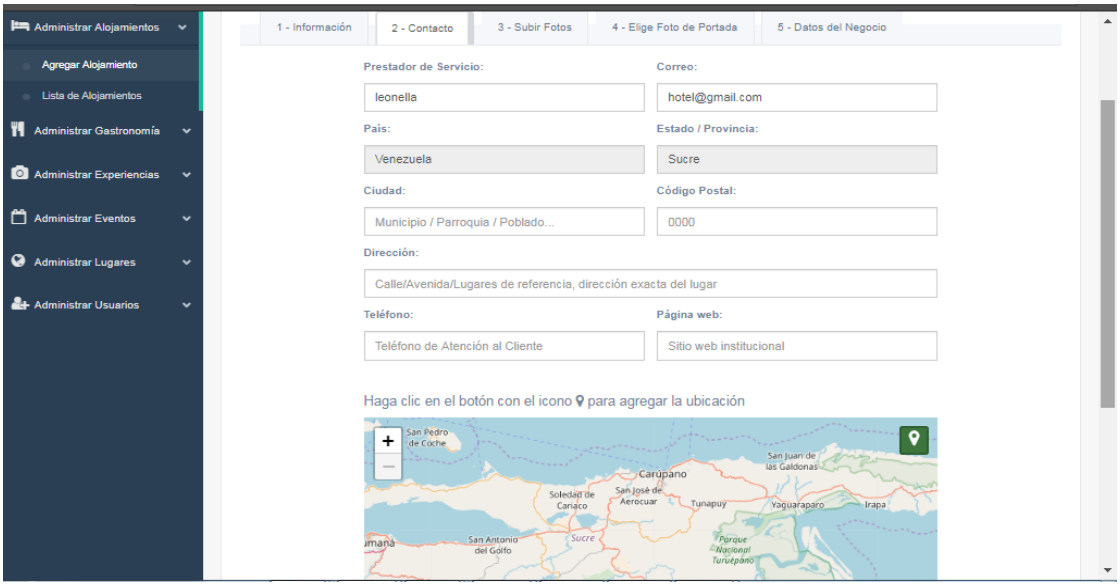


Figura R. 24.Caso de prueba A2 ingresar alojamiento por administrador.

En el caso de pruebas A3 de Agregar Alojamiento el sistema no permitió agregar el sitio porque el valor dirección no puede quedar en blanco.

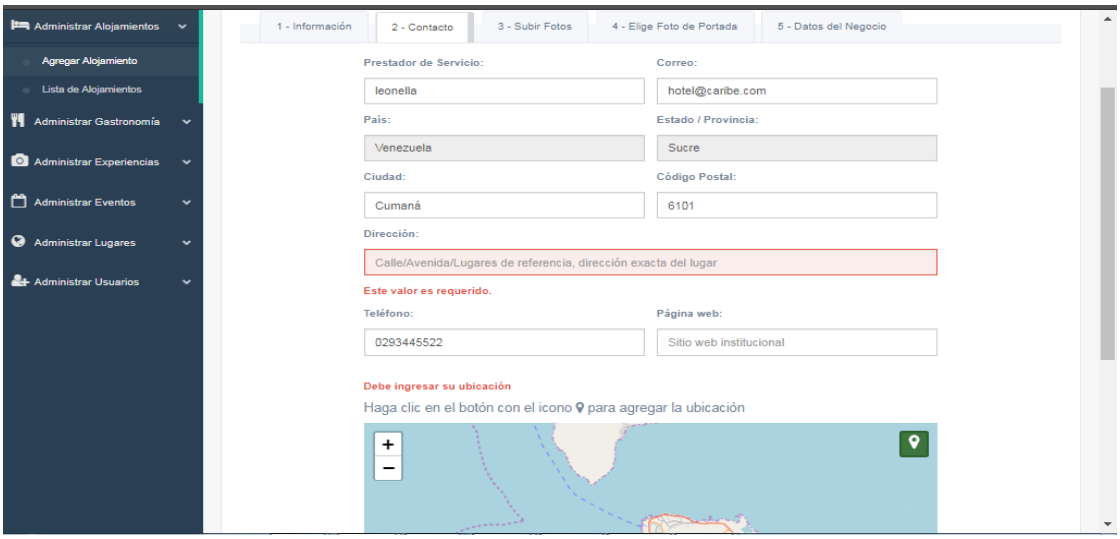


Figura R. 25.Caso de prueba A3 ingresar alojamiento por administrador

Conclusión de prueba

Mediante la prueba se validó que la función cumple su finalidad y que además los campos de ingreso del formulario están validados.

Prueba de interfaz y usabilidad

De acuerdo al análisis realizado a toda la interfaz de Administrador y al uso e implementación de principios de usabilidad de Nielsen, esta interfaz es óptima para su uso.

Apéndice S. Ciclo de Versión Administrador: Iteración Listar Alojamiento.

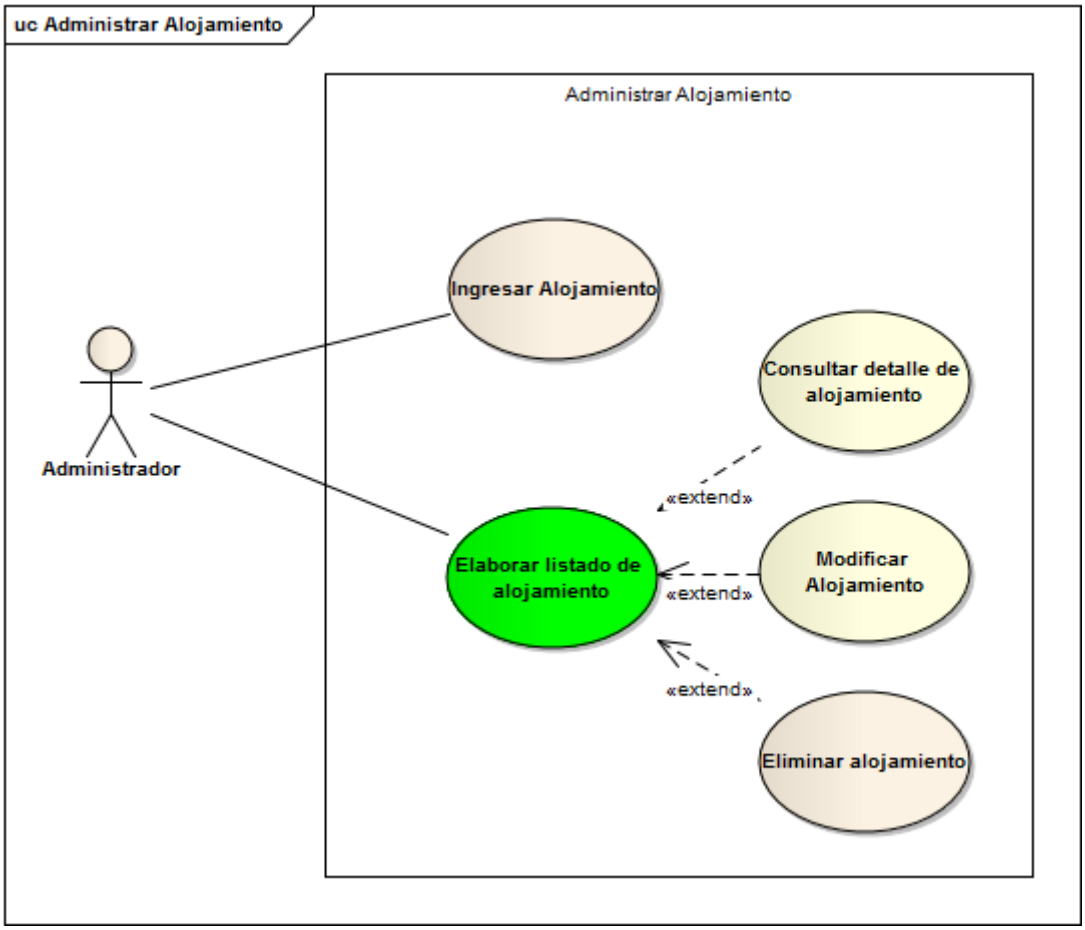


Figura S. 1.Caso de uso: Elaborar Listado Alojamientos por administrador

Tabla S. 1. Descripción Caso de uso Elaborar Listado Alojamientos por usuario administrador

Nombre del Caso de uso	Elaborar listado de alojamientos.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en elaborar un listado de los alojamientos registrados en la de base de datos del sistema, de manera satisfactoria.	
Precondición	Para elaborar el listado de los alojamientos, deben estar registrados en la base de datos del sistema.	
Secuencia Normal de Eventos	Paso	Acción
	1	El Administrador selecciona la opción “Lista de alojamientos”.
	2	El sistema muestra el listado de los alojamientos almacenados en la base de datos del sistema, de acuerdo a un orden predeterminado, brindando las siguientes opciones de extensión: a. “Consultar alojamiento”. b. “Modificar alojamiento” c. “Eliminar alojamiento”
	3	El Administrador visualiza el listado de los alojamientos y selecciona alguna de las opciones de extensión propuestas.
	4	El sistema extiende <<extend>> hacia el caso de uso que representa la opción seleccionada.
	5	El sistema finaliza el caso de uso.
Postcondición	Se elabora el listado de alojamientos existentes en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de	Paso	Acción

Eventos	3.1.	El Administrador decide no seleccionar ninguna opción de extensión, se cancela el caso de uso, seleccionando otra opción en el sistema.
	3.2.	El sistema finaliza el caso de uso.
Reglas del negocio	1. El listado de alojamientos debe ser elaborado de acuerdo a un orden preestablecido. 2. Se podrá seleccionar, desde el listado general de alojamientos, un alojamiento en específico para realizar las operaciones de consultar, modificar o eliminar.	
Inclusiones	Ninguna.	
Extensiones	1. “Consultar alojamiento”. 2. “Modificar alojamiento”. 3. “Eliminar alojamiento”.	

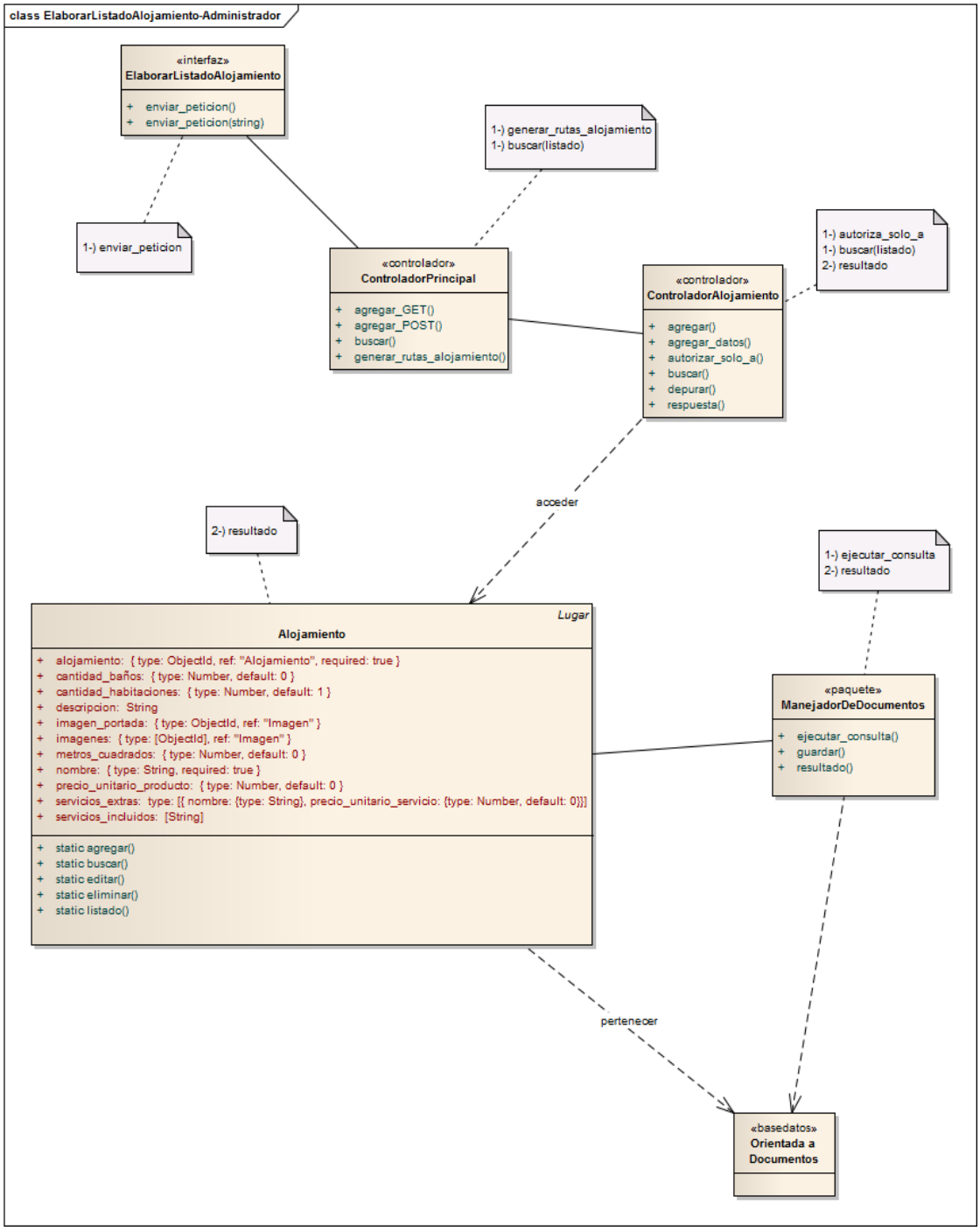


Figura S. 2.Diagrama de Clases: Elaborar Listado Alojamientos por usuario Administrador

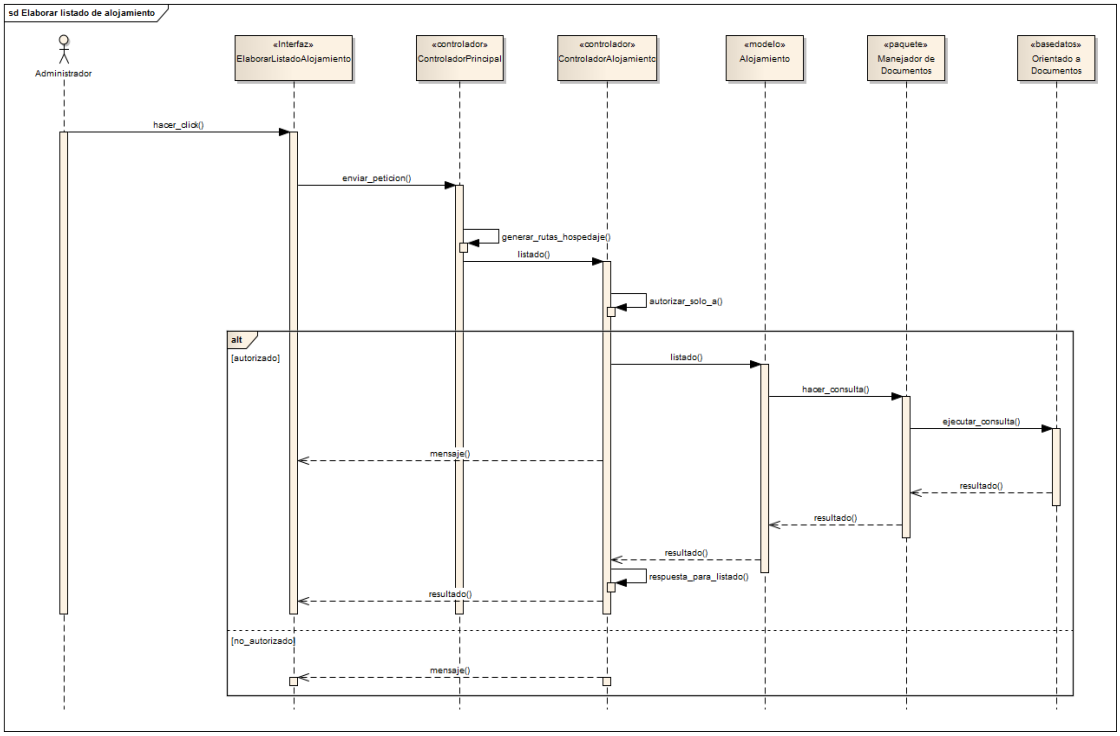


Figura S. 3.Diagrama de secuencia: Elaborar Listado Alojamientos por usuario Administrador

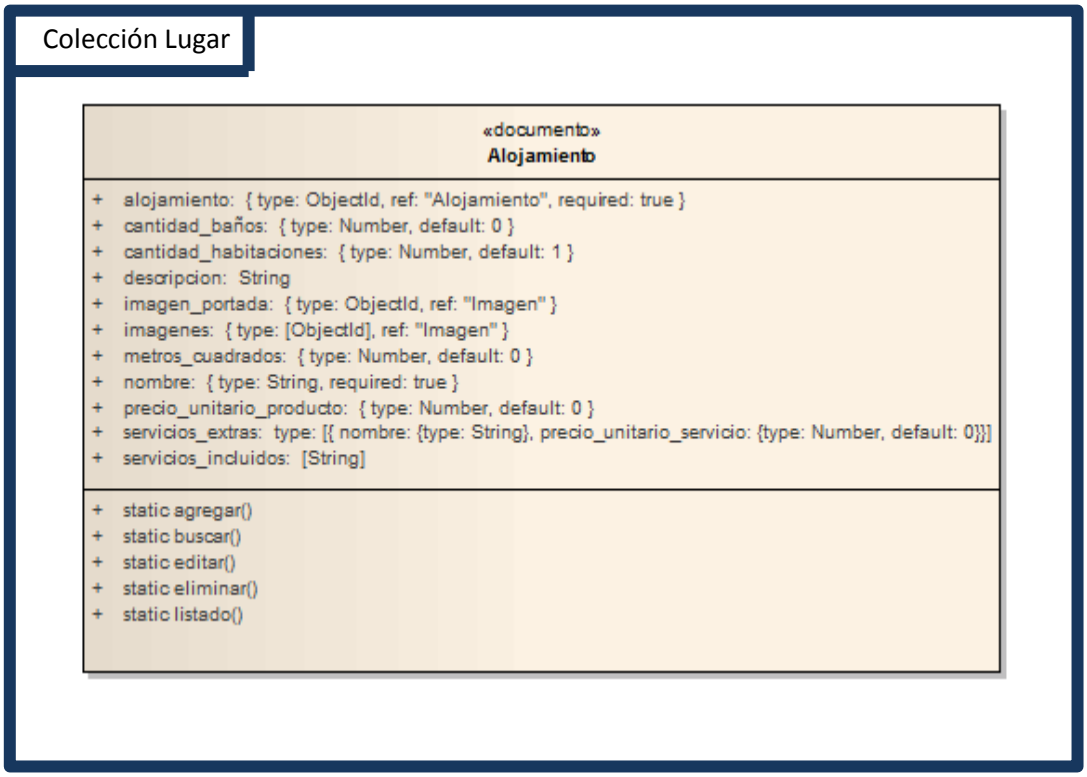


Figura S. 4.Colección de Alojamiento: Listado de alojamiento por usuario Administrador.

A continuación, se muestran el modelo de plantilla para el proceso de listado de alojamiento.

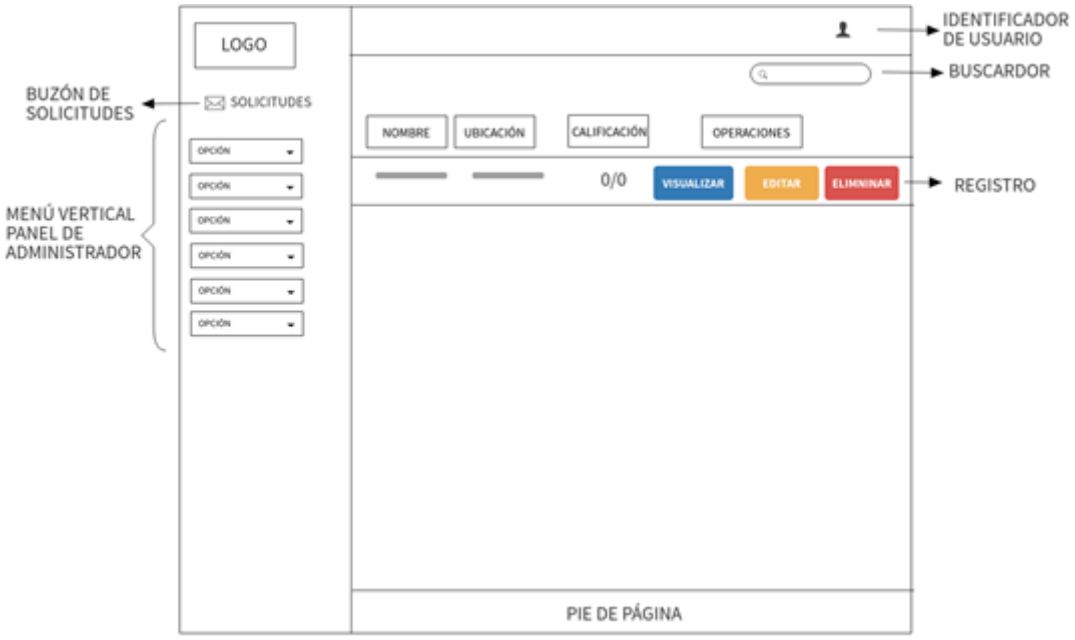


Figura S. 5. Plantilla de interfaz: Listado de Alojamiento por usuario administrador

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Lista de Alojamiento y el sistema responde mostrando la interfaz correspondiente.

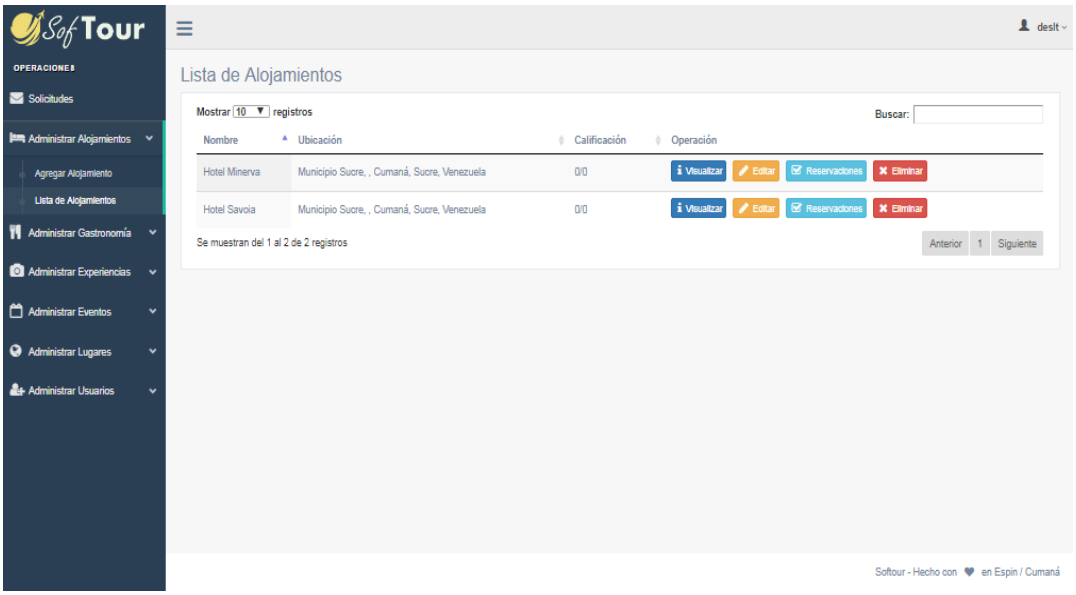


Figura S. 6. Interfaz Listado de Alojamientos por usuario Administrador

Parte del código desarrollado para la funcionalidad Listado de Alojamiento por usuario Administrador

```
listado(req, res, next) {
    let query = {};

    if(req.session.passport.user.tipo_usr == "PrestadorServicio" && req.params.tipo_entidad != "lugar")
        query.prestador_servicio = req.session.passport.user.id;

    if(req.params.tipo_entidad == "lugar")
        query.__t = null;
    else{
        query.__t = req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1);
        query.aprobado = true;
    }

    this._Lugar.listado({"consulta": query}, (error, lista) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_listado(req, res);
        }

        res.locals.datos = {"value": lista};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_listado(req, res);
    })
}
```

Figura S. 7. Código desarrollado para el proceso Listar Alojamiento por usuario Administrador.

Apéndice T. Ciclo de Versión Administrador: Iteración Consultar Alojamiento.

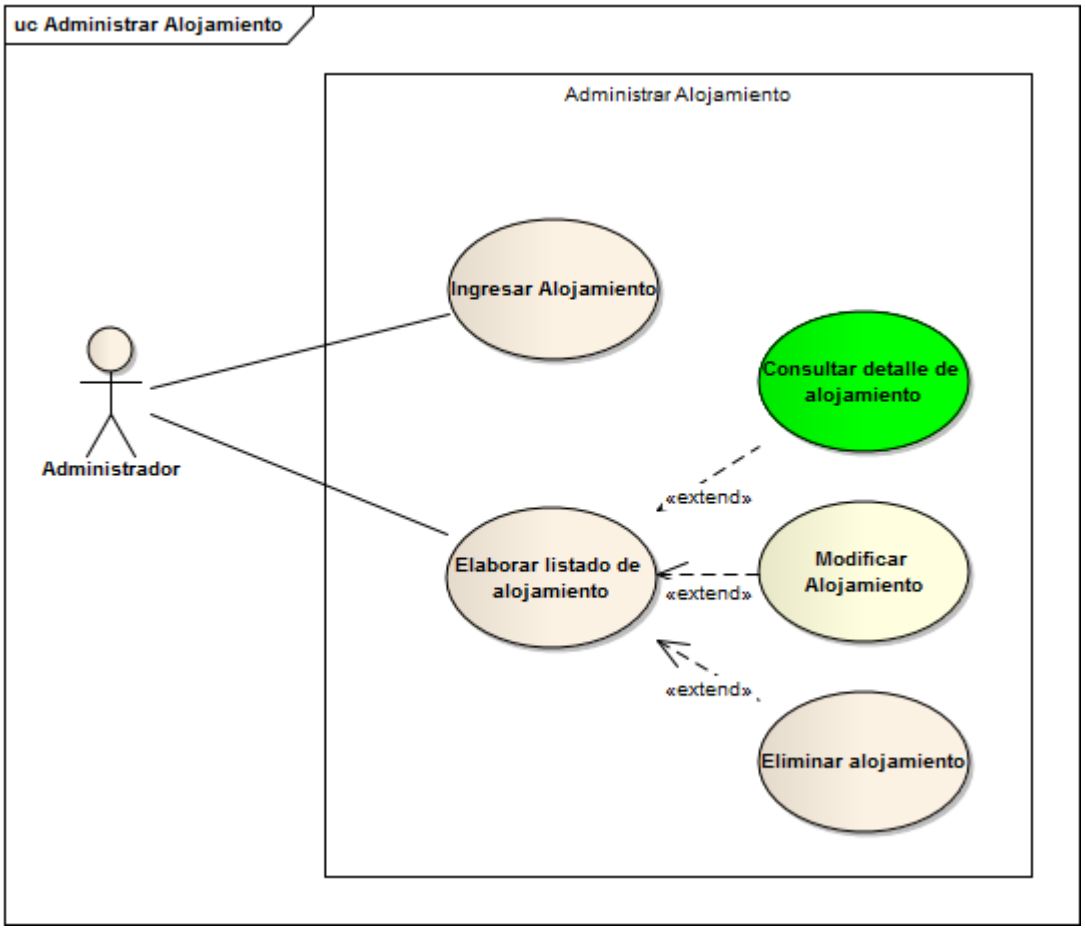


Figura T. 1.Caso de uso: Consultar Detalle de Alojamiento por administrador

Tabla T.1.Descripción Caso de uso Consultar Detalle de Alojamiento por usuario administrador

Nombre del Caso de uso	Consultar detalle de alojamiento.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en consultar los atributos del alojamiento de manera satisfactoria.	
Precondición	Para consultar un alojamiento, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Consultar alojamiento".
	2	El sistema ejecuta la búsqueda del alojamiento, mostrando los datos del alojamiento a consultar.
	3	El Administrador visualiza la consulta de los atributos del alojamiento.
	4	El sistema finaliza el caso de uso.
Postcondición	Se consultan los atributos del alojamiento de la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
Reglas del negocio	1. El alojamiento debe estar registrado para que pueda ser consultado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

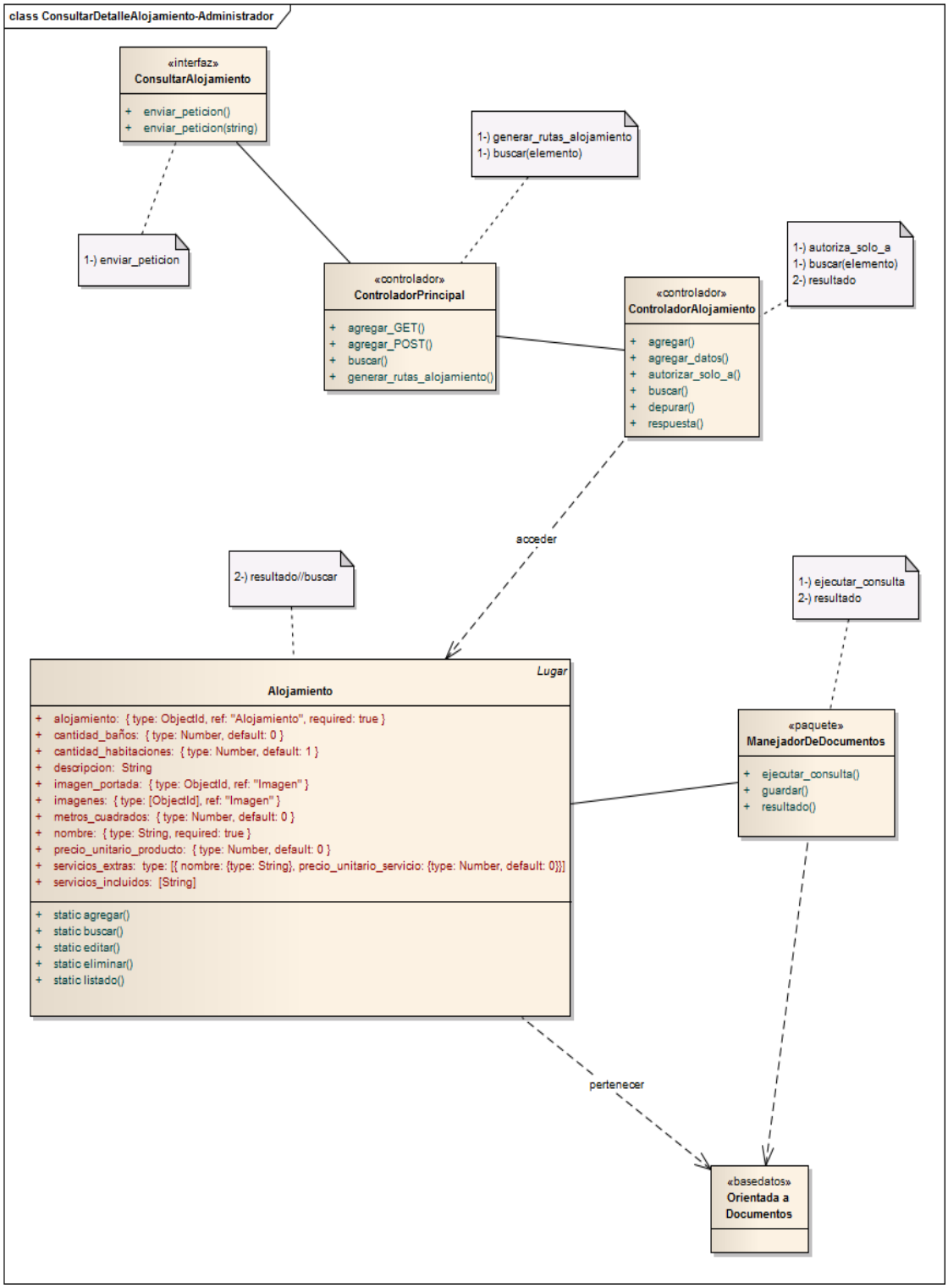


Figura T. 2.Diagrama de Clases: Consultar Detalle Alojamiento por usuario Administrador

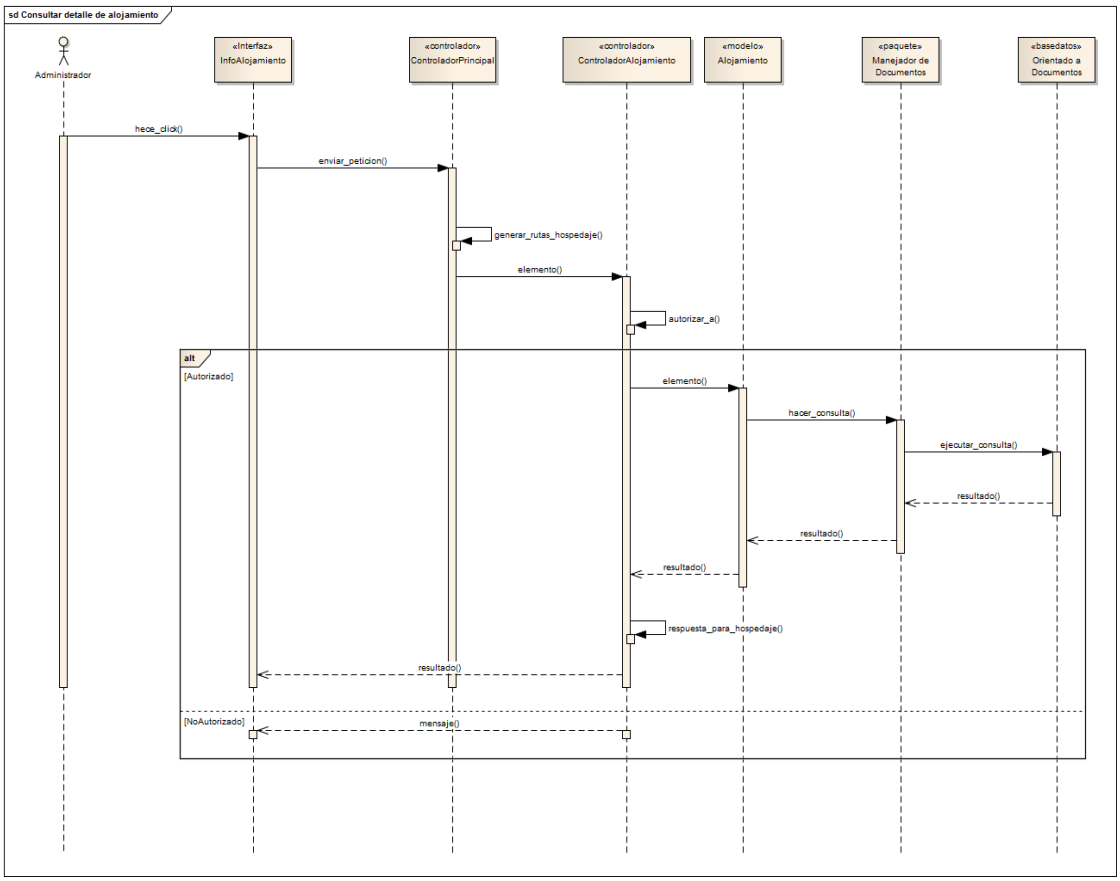


Figura T. 3.Diagrama de secuencia: Consultar Detalle Alojamiento por usuario Administrador

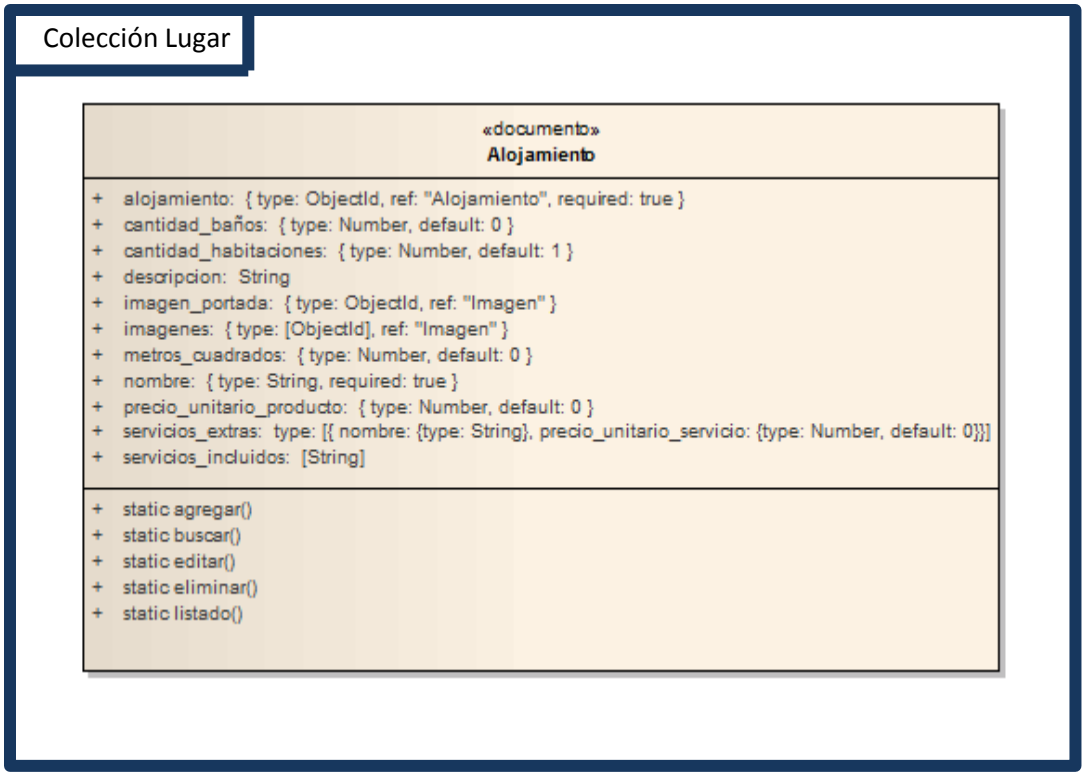


Figura T. 4.Colección de Alojamiento: Consultar alojamiento por usuario administrador.

A continuación, se muestran el modelo de plantilla para el proceso de consultar alojamiento.

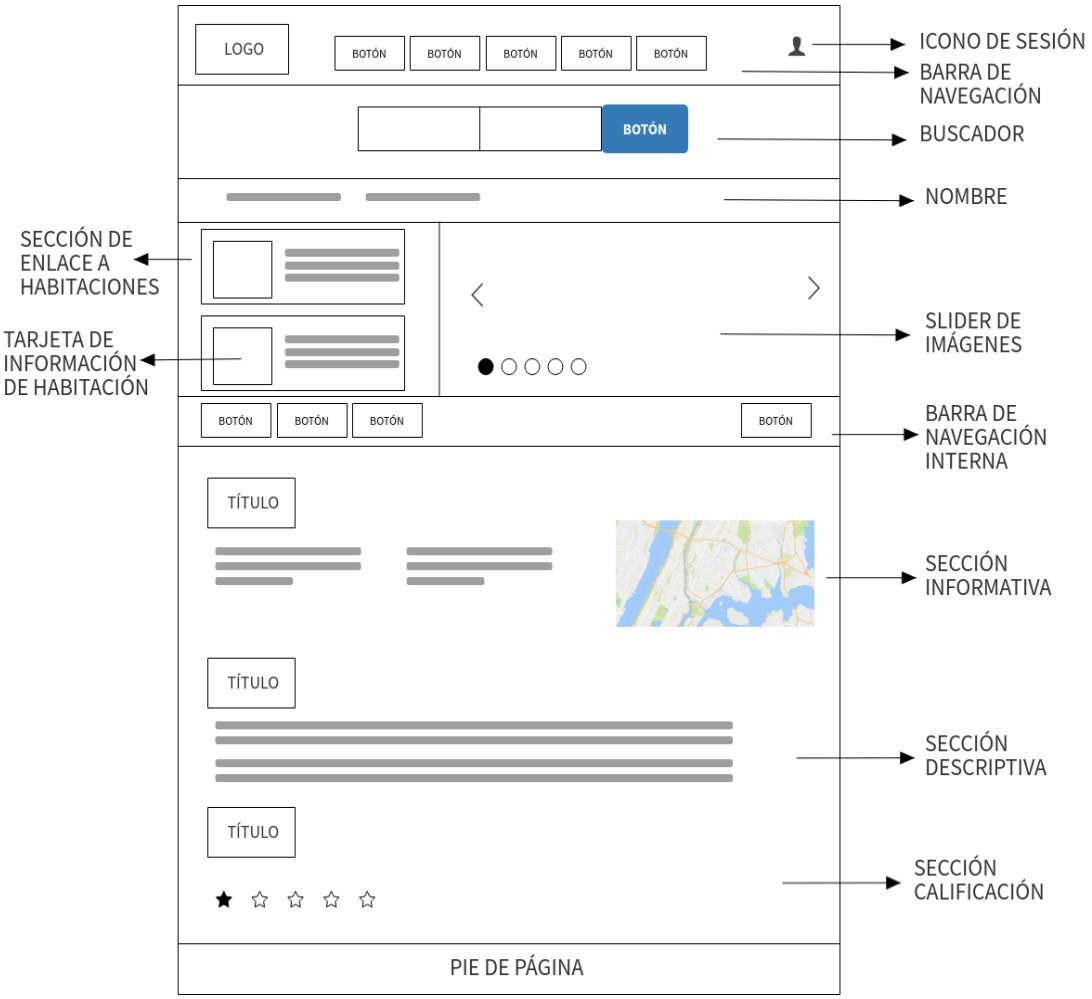


Figura T. 5.Plantilla de interfaz: Consultar Alojamiento por usuario administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Consultar Alojamiento y el sistema responde mostrando la interfaz con las especificaciones del alojamiento.

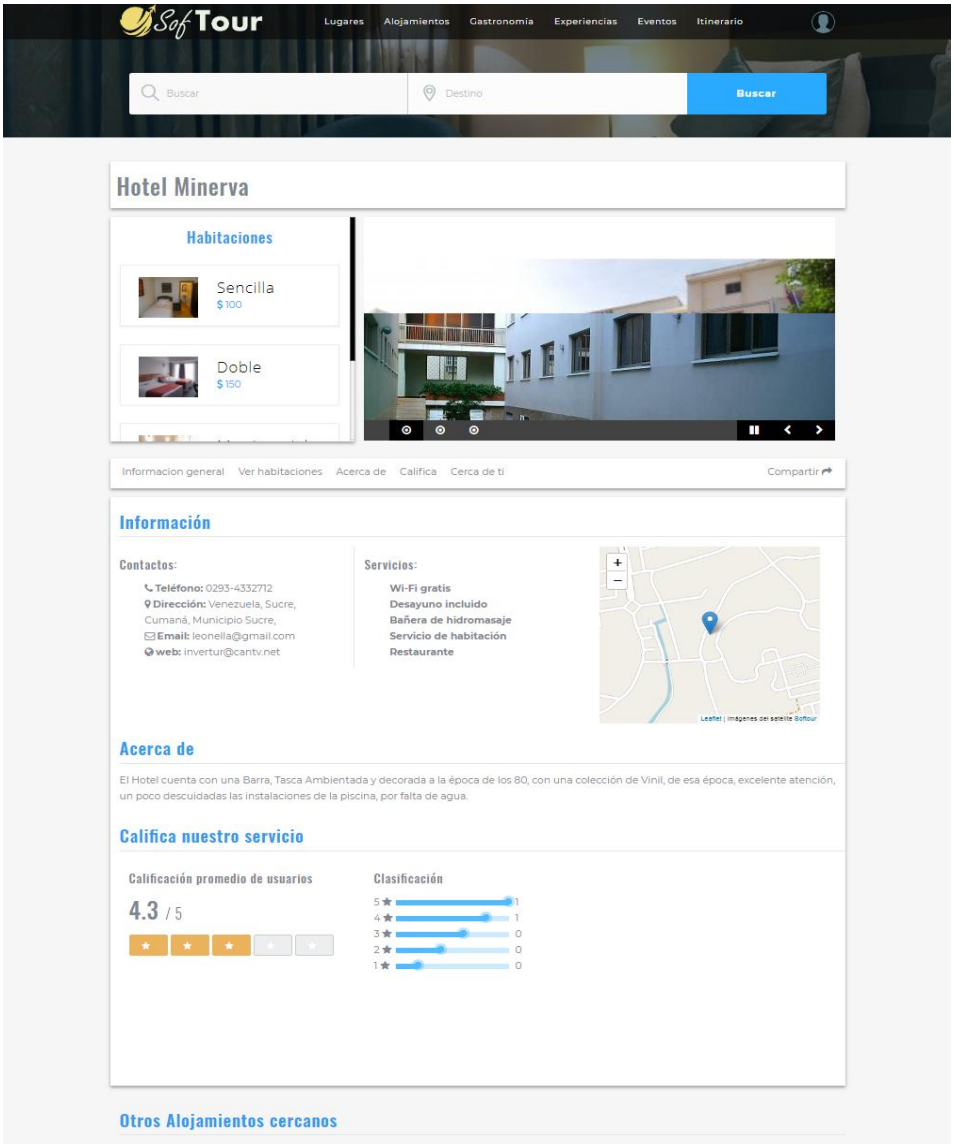


Figura T. 6.Interfaz de proceso de Consultar Detalle Alojamiento por usuario Administrador

Parte del código desarrollado para la funcionalidad Consultar Detalle Alojamiento por usuario Administrador

```
consultar(req, res, next) {
    let parametros = {
        "__t": req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1),
        "url": req.params.url.toLowerCase()
    };
    if(parametros.__t)
        parametros.aprobado = true;

    this._Lugar.buscar_consulta(parametros, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        if(buscado.__t){
            this.CategoriaAlojamiento.find({"alojamiento": buscado._id}, (error, lista_categorias) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    if (req.isAuthenticated())
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_consultar_producto(req, res);
                }

                res.locals.datos = {
                    "value": buscado,
                    "lista_categorias": lista_categorias
                };

                if (req.isAuthenticated())
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                this._respuesta.para_consultar(req, res);
            });
        }
        else {
            res.locals.datos = { "value": buscado };

            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
    })
}
```

Figura T. 7. Código desarrollado para el proceso Listar Alojamiento por usuario Administrador.

Prueba Consultar Detalle alojamiento

Consultar detalles de alojamientos consiste en revisar que los datos registrados en la función Ingresar alojamiento sean guardados satisfactoriamente. Cabe señalar que para la validación de esta prueba es necesario tener Alojamientos previamente registrados en base datos del sistema, para cumplir con eso se utilizaran los datos registrados en la prueba anterior.

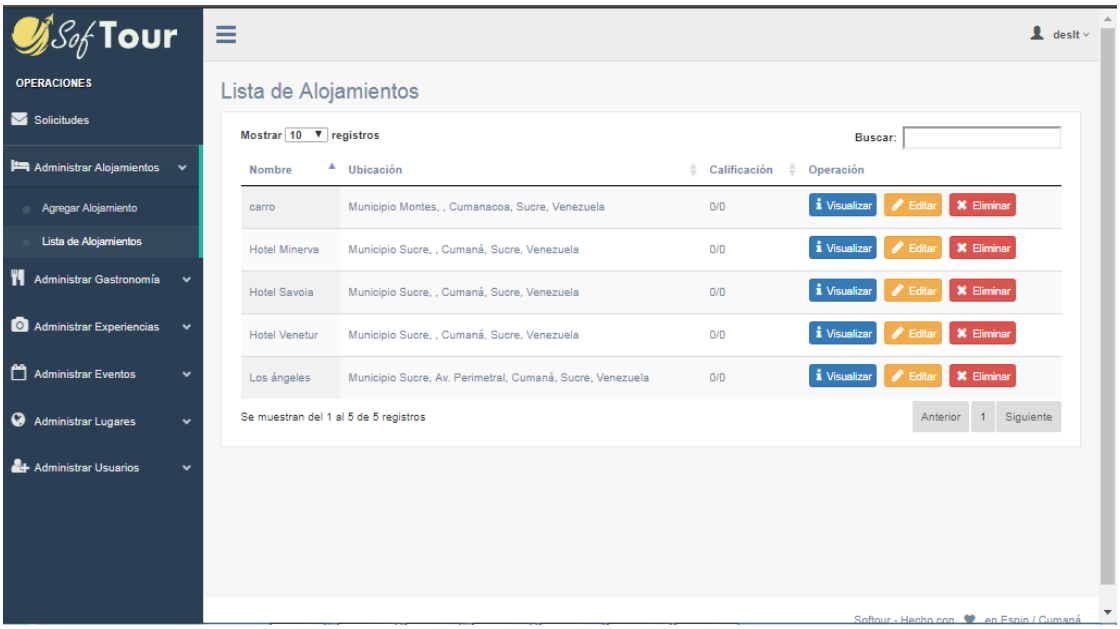


Figura T. 8.Prueba visualizar alojamiento por Administrador. Imagen listados alojamiento

A continuación, el sistema muestra la información del alojamiento, validando que la función se ejecuta de forma correcta.

Figura T. 9.Prueba visualizar alojamiento por administrador

Conclusión de prueba

La función se ejecutó de forma correcta.

Pruebas de interfaz y usabilidad

La interfaz forma parte del conjunto de interfaces de usuario Administrador que fueron diseñadas bajos estándares de usabilidad y cuenta con dichos principios.

Apéndice U. Ciclo de Versión Administrador: Iteración Modificar Alojamiento.

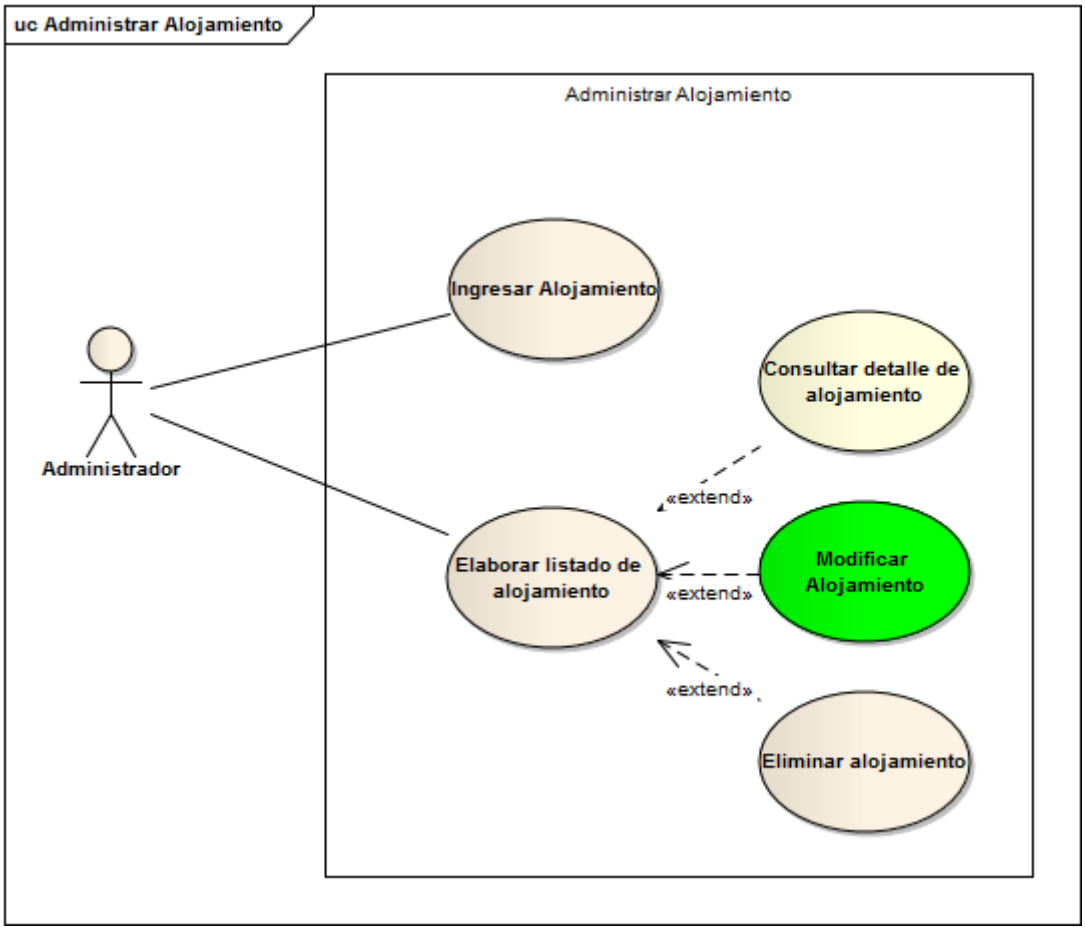


Figura U. 1.Caso de uso: Modificar Alojamiento por administrador

Tabla U. 1. Descripción Caso de uso Modificar Alojamiento por usuario administrador

Nombre del Caso de uso	Modificar alojamiento.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en modificar los atributos del alojamiento de manera satisfactoria.	
Precondición	Para modificar un alojamiento, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Modificar”.
	2	El sistema ejecuta la búsqueda del alojamiento, mostrando un formulario con la consulta en modo edición.
	3	El Administrador visualiza la consulta de los atributos del alojamiento que desea modificar, los modifica y selecciona aceptar.
	4	El sistema realiza las modificaciones en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se modifican los atributos del alojamiento en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no modificar el alojamiento y selecciona cancelar el caso de uso, seleccionando otra opción del sistema.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. El alojamiento debe estar registrado para que pueda ser modificado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

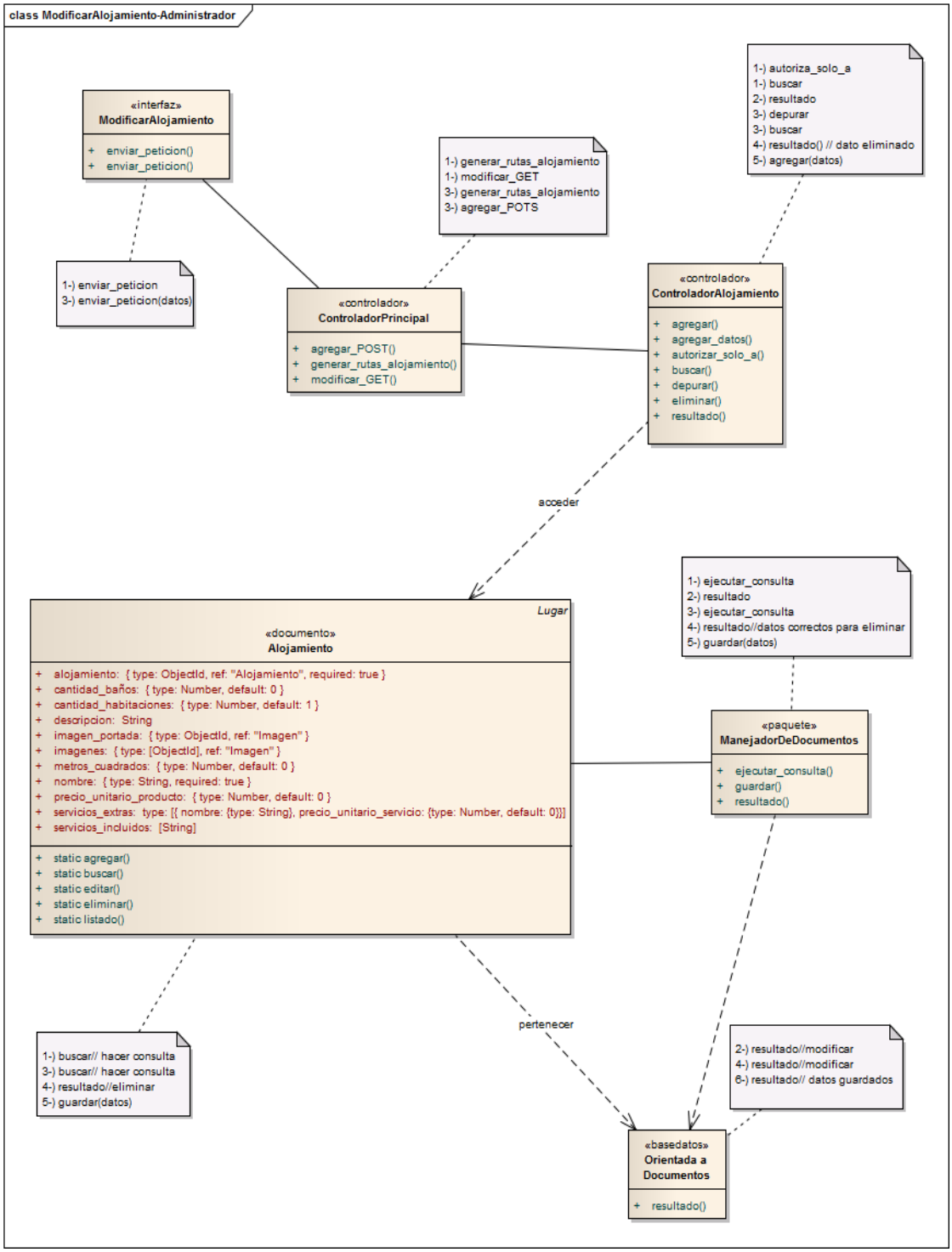


Figura U. 2.Diagrama de Clases: Modificar Alojamiento por usuario Administrador

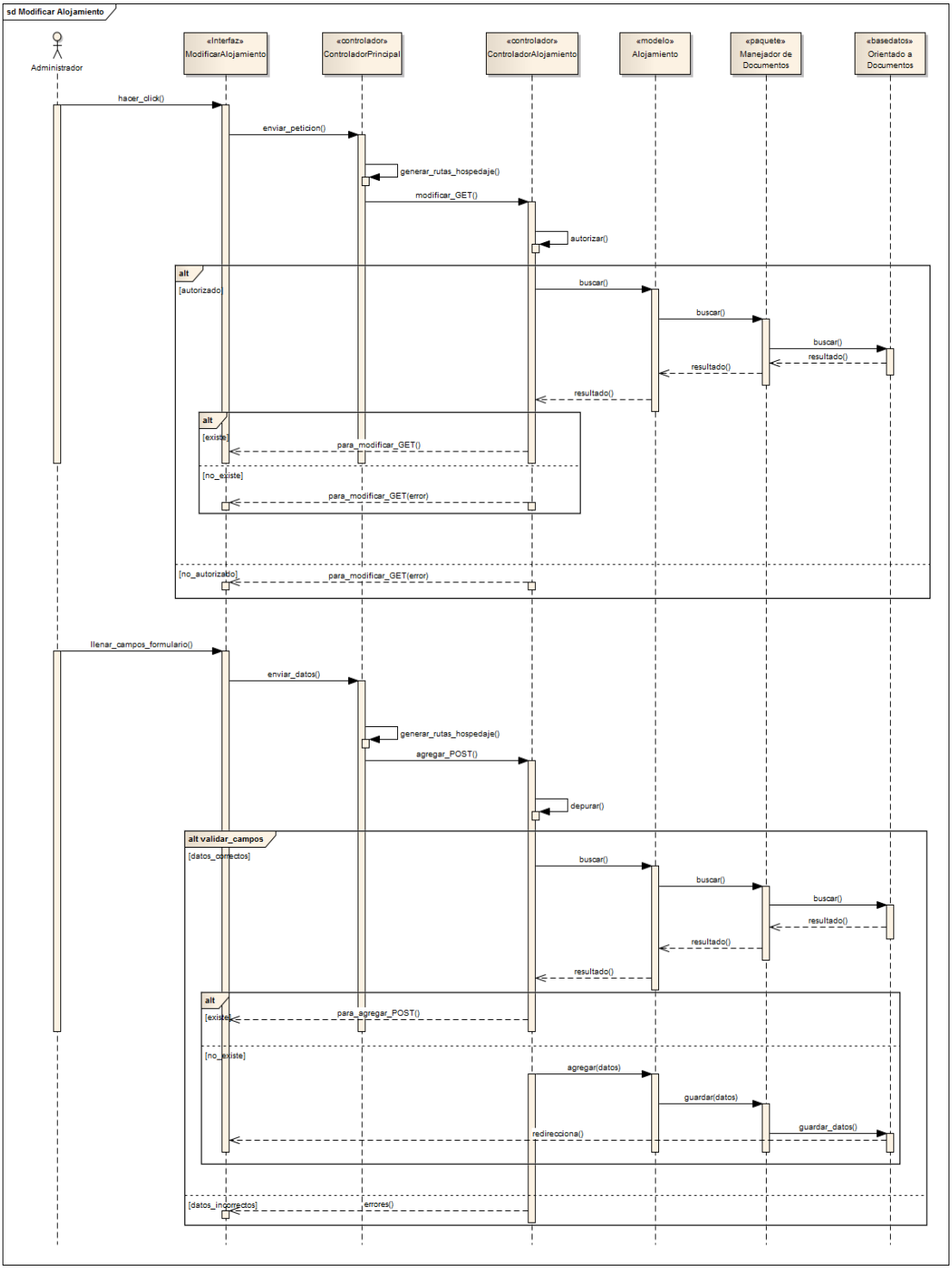


Figura U. 3.Diagrama de secuencia: Modificar Alojamiento por usuario administrador

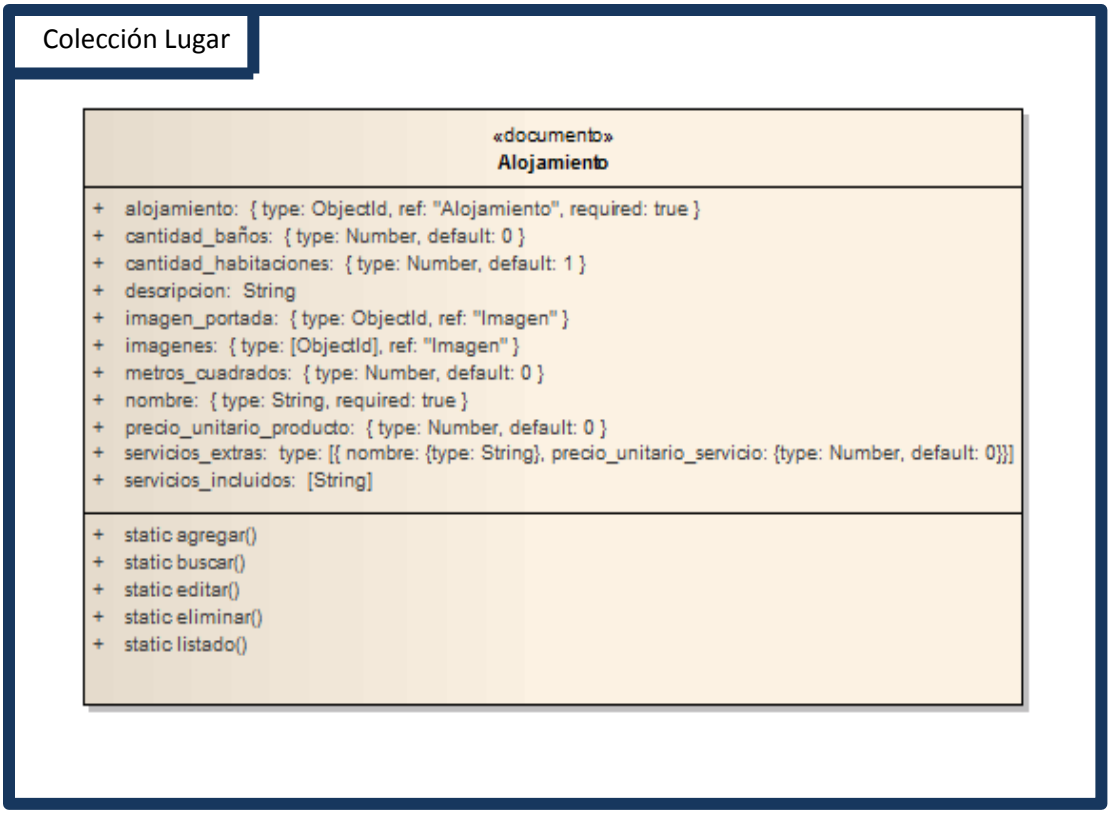


Figura U. 4. Colección de Alojamiento: Modificar alojamiento por usuario Administrador

A continuación, se muestran los modelos de plantillas para el proceso de modificar alojamiento.

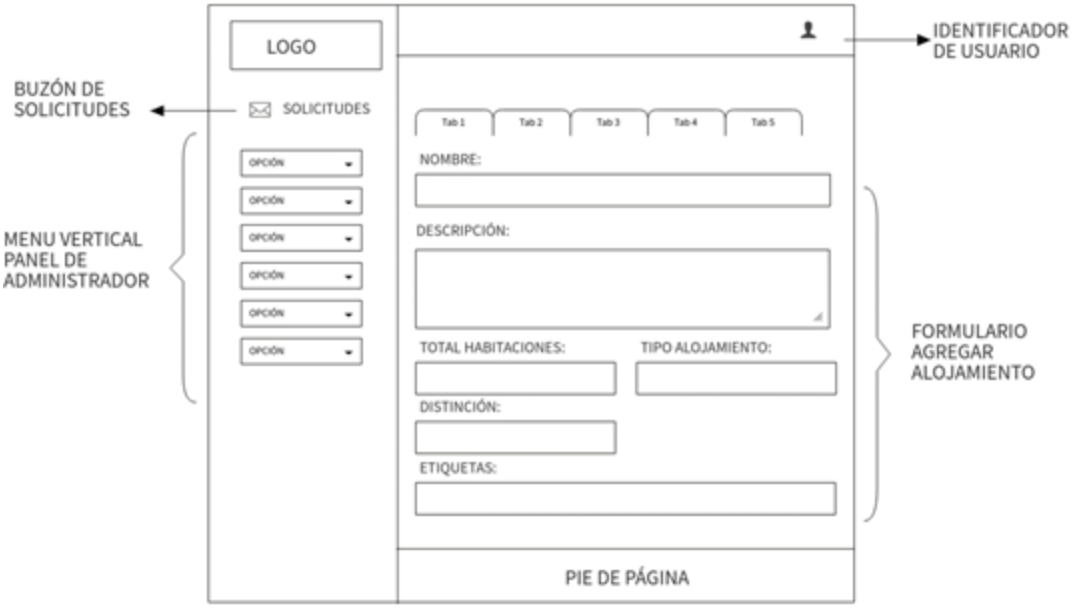


Figura U. 5.Plantilla de interfaz 1: Modificar Alojamiento por usuario administrador.

Figura U. 6.Plantilla de interfaz 2: Modificar Alojamiento por usuario administrador.

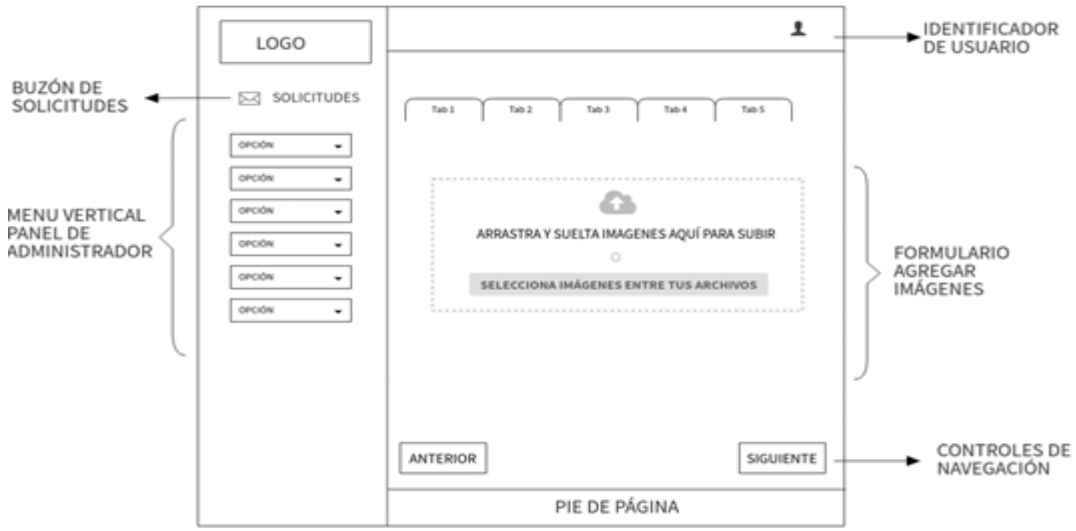


Figura U. 7. Plantilla de interfaz 3: Modificar Alojamiento por usuario administrador.

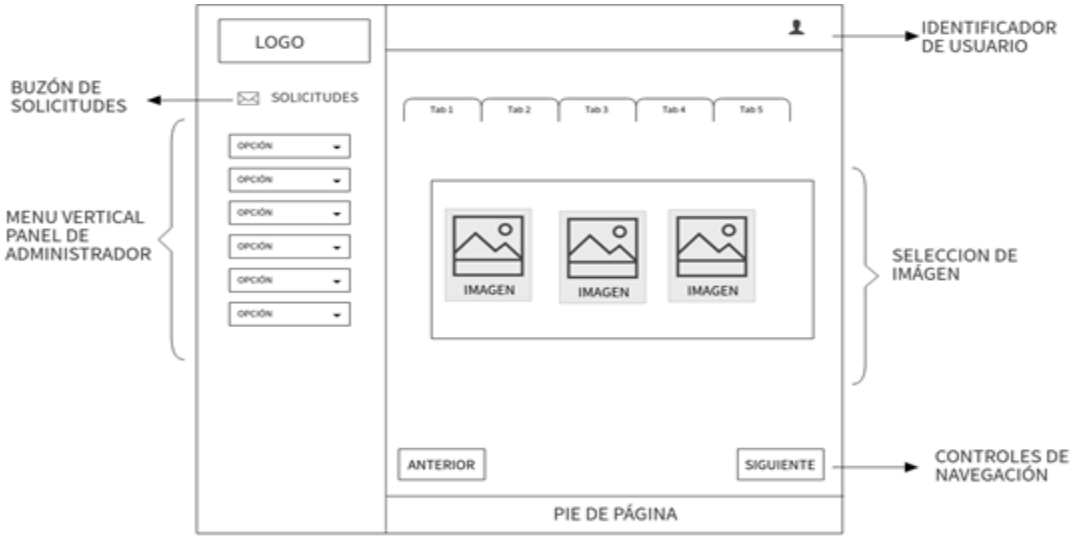


Figura U. 8. Plantilla de interfaz 4: Modificar Alojamiento por usuario administrador.

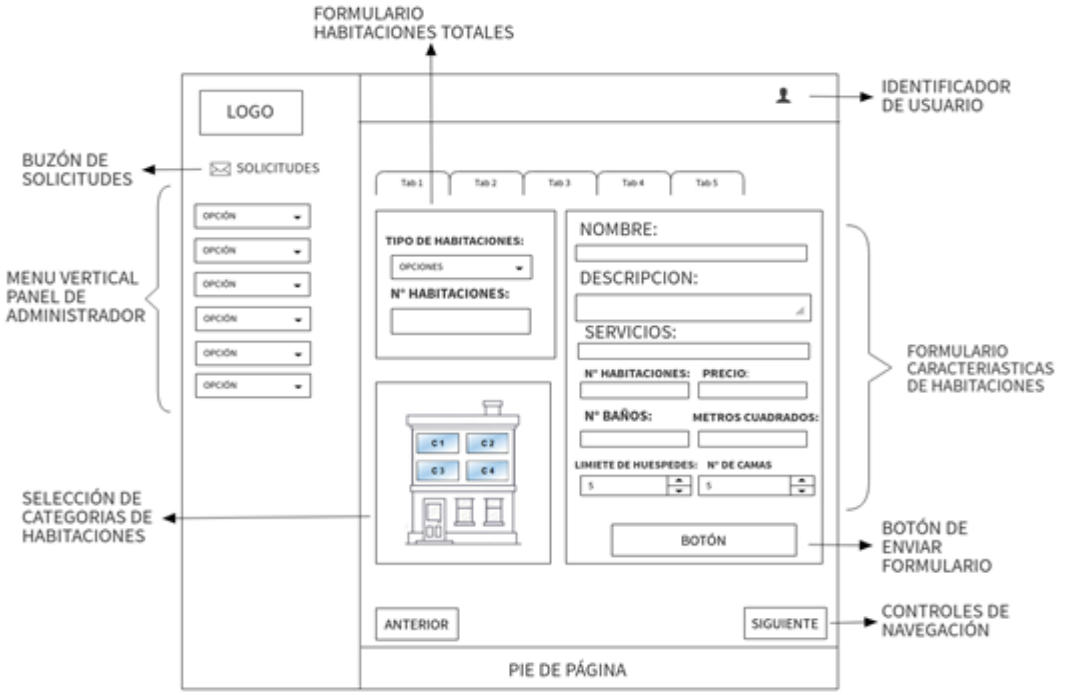


Figura U. 9. Plantilla de interfaz 5: Modificar Alojamiento por usuario administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Modificar Alojamiento y el sistema responde mostrando el formulario correspondiente.

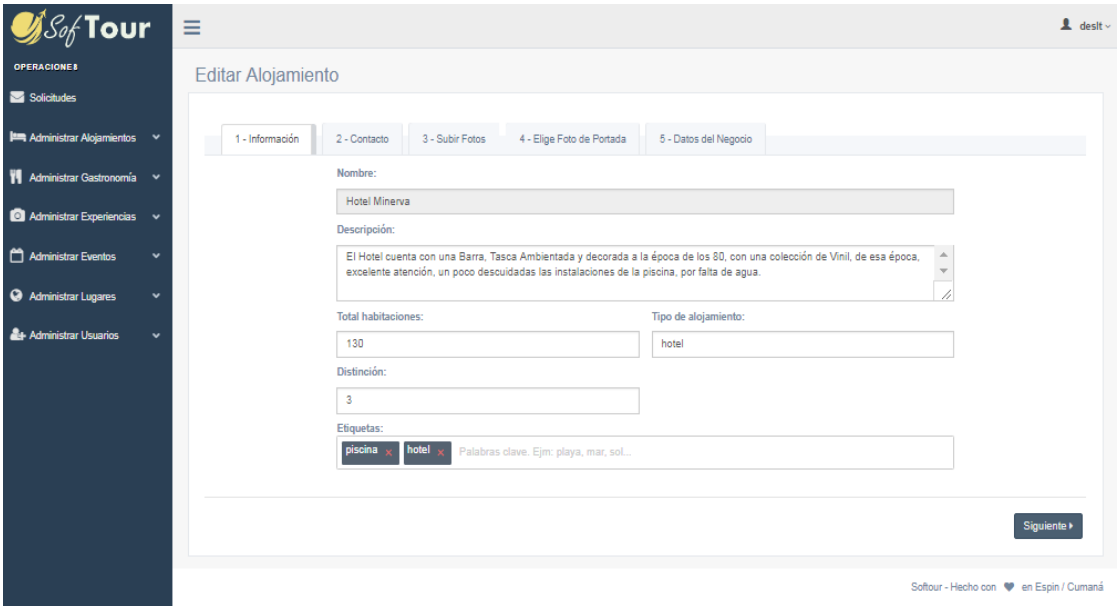


Figura U. 10. Interfaz 1 de proceso de Modificar Alojamiento por usuario Administrador

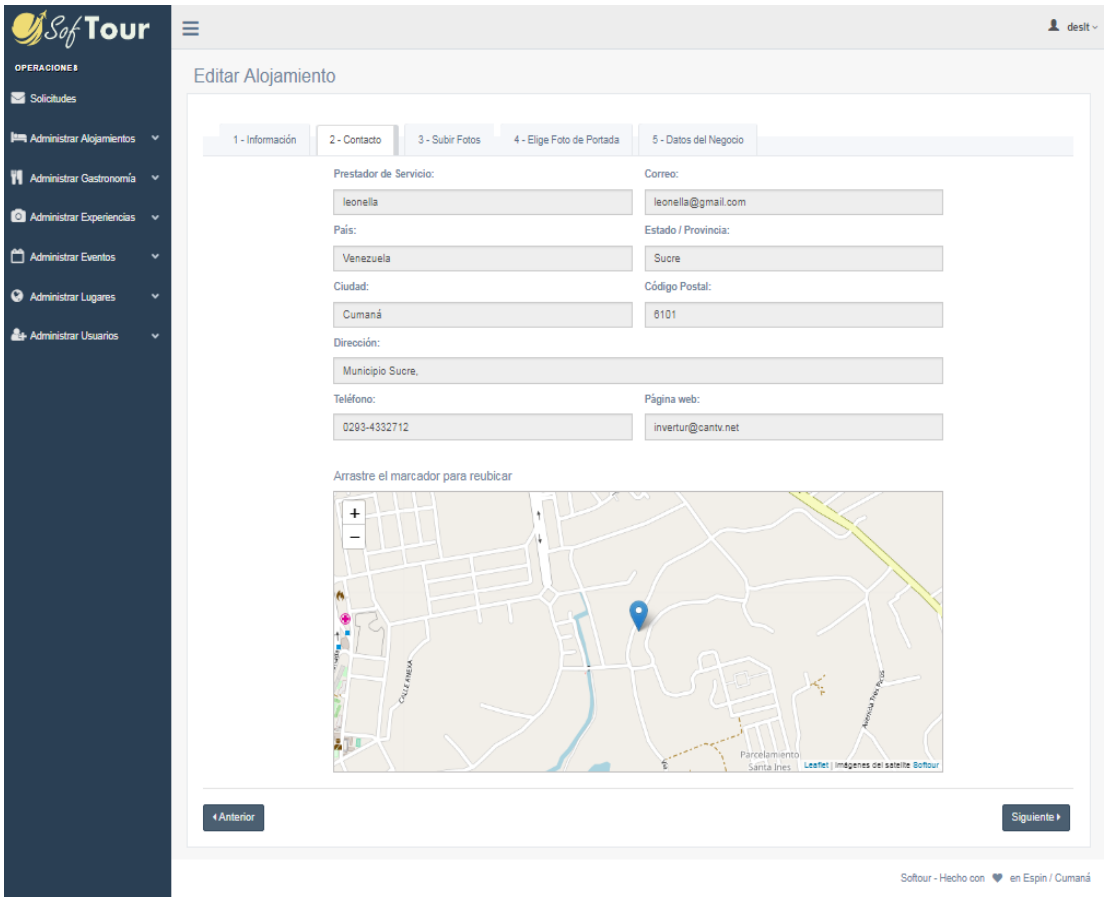


Figura U. 11. Interfaz 2 de proceso de Modificar Alojamiento por usuario Administrador

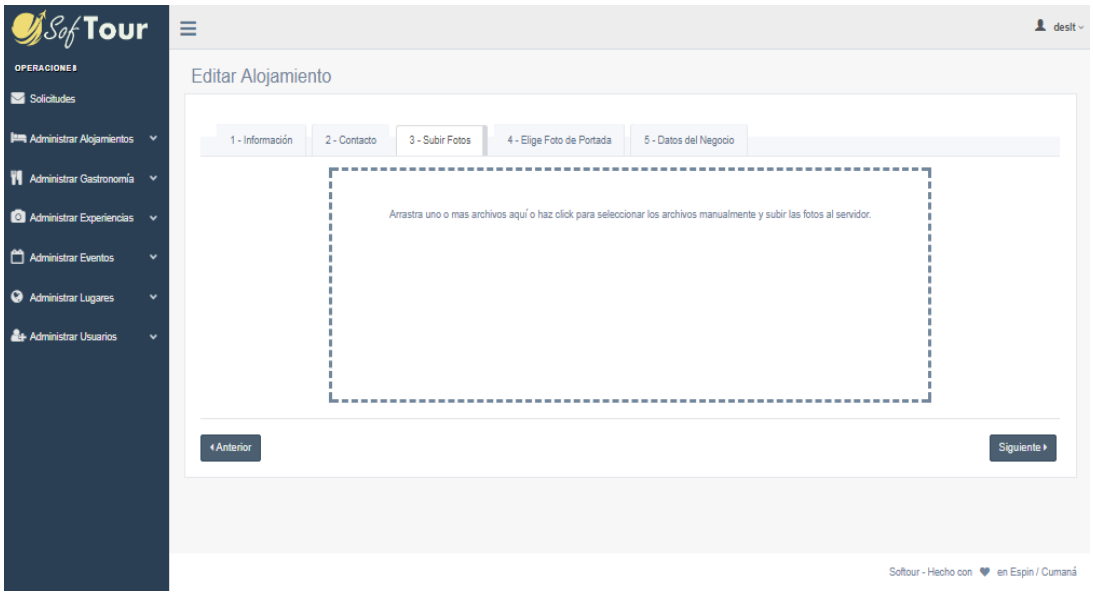


Figura U. 12. Interfaz 3 de proceso de Modificar Alojamiento por usuario Administrador

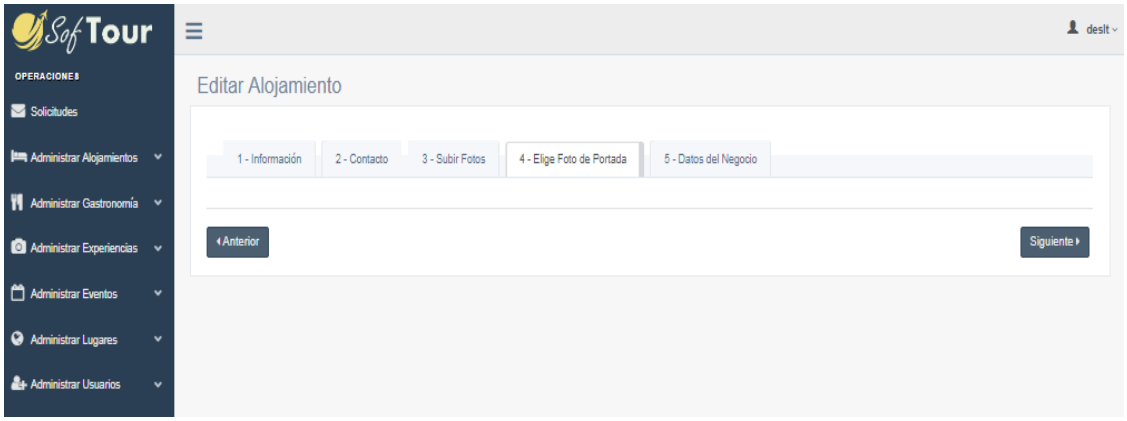


Figura U. 13. Interfaz 4 de proceso de Modificar Alojamiento por usuario Administrador

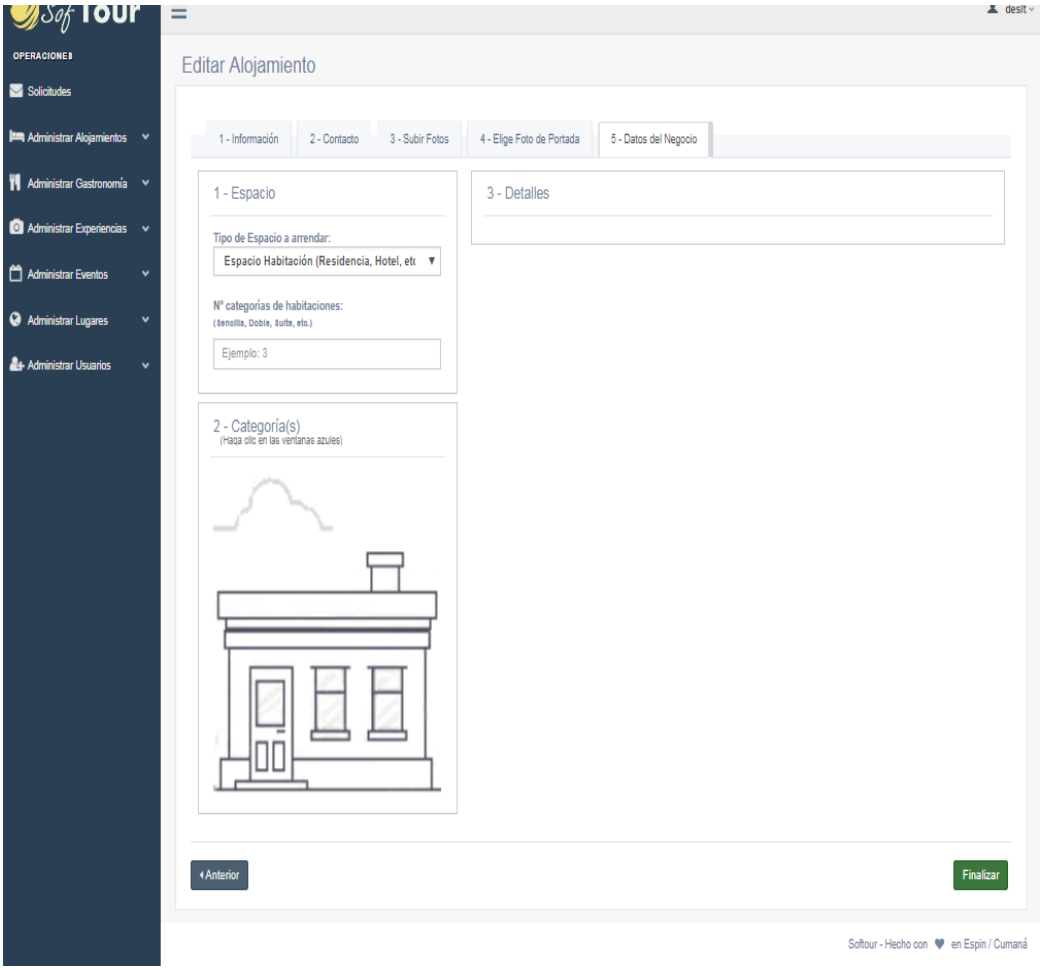


Figura U. 14. Interfaz 5 de proceso de Modificar Alojamiento por usuario Administrador

Parte del código desarrollado para la funcionalidad Modificar Alojamiento por usuario Administrador

```
function constructorVentanas(e) { // una ventana es un form de una categoria
    let ventanas = [],
        forms_categ = [];
    $('#contenedor-categorias').empty();

    $('#categoria-habitaciones').empty();
    for (let i = 1; i <= $('input[name="cantidad_categorias"]').val(); i++) {
        ventanas.push('<a href="#detalles" class="ventana-categoria col-md-5 col-sm-5 col-lg-5" id="' + i + '">C ' + i + '</a>');
        forms_categ.push('<form class="form-horizontal clase-formulario-categorias" id="C' + i + '">' + form_espacio_habitacion + '</
            form>');
    };
    $('#categoria-habitaciones').append(ventanas);
    $('#contenedor-categorias').append(forms_categ);
    inicializarEtiquetador(".input-etiquetas", "Wifi, Piscina, Restaurant...");

    // mostrar solo el form q quiero ver
    $('.ventana-categoria').click(function(e) {
        $(".clase-formulario-categorias").hide();
        $(".clase-formulario-categorias").filter("[id=C" + $(this).attr("id") + "]").show();
    });

    // cargar datos al pulsar el btn
    $('.btn-guardar-categoria').click(function(e) {
        e.preventDefault();
        let contenedor = $(this).parent(),
            f = $(contenedor).attr("id").slice(1);
        $(contenedor).parsley({excluded: 'a, input[type=button], input[type=submit], input[type=reset]', inputs: 'input, textarea,
            select, input[type=hidden], :hidden'})
            .whenValidate()
            .done(() => {
                $(contenedor).find("input, textarea, .input-etiquetas").attr("readonly", "readonly");
                let tagsinput = $(contenedor).find("[name='servicios_incluidos']").siblings();

                $(tagsinput).css("background-color", "#eee");
                $(tagsinput).find("div").css("background-color", "#eee");
                $(tagsinput).find(".tag a").hide();
                $(this).attr("disabled", "disabled");

                new PNotify({
                    title: 'Cargado',
                    text: 'Ingrese siguiente categoría',
                    styling: "bootstrap3",
                    delay: 10000,
                    type: "success",
                    sticker: false
                });

                lista_categorias[parseInt(f) - 1] = $(contenedor).serialize() + "&id_alojamiento=" + id_alojamiento;
            });
    });
}

$('select[name="tipo_espacio"]')
    .ready(function(e) {
        lista_categorias = [];
```

Figura U. 15. Código desarrollado para el proceso Modificar Alojamiento por usuario Administrador.

Prueba modificar alojamiento

Para validar y verificar esta función se utilizarán los datos registrados con

anterioridad, pero como técnica de prueba se utilizó caja negra. La finalidad de esta función es editar los datos guardados previamente en la información guardada en la base de datos del sistema.

En la prueba que se presenta a continuación se modificaran los siguientes datos:

Distinción: 4

Página web: angeleshotel.es

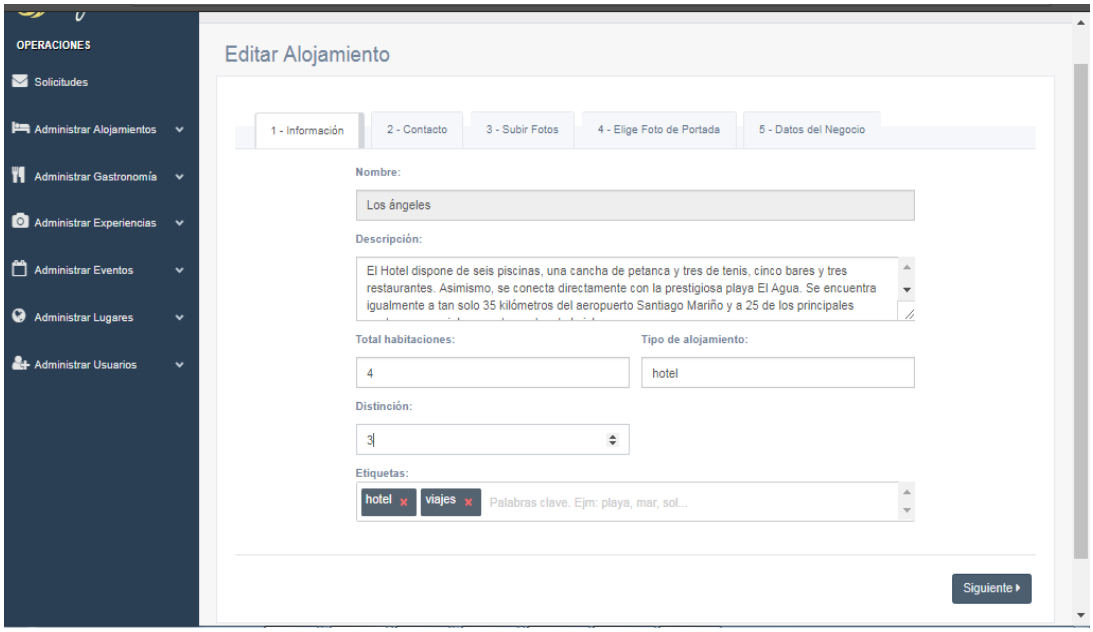


Figura U. 16.Prueba modificar alojamiento por administrador. Paso 1.

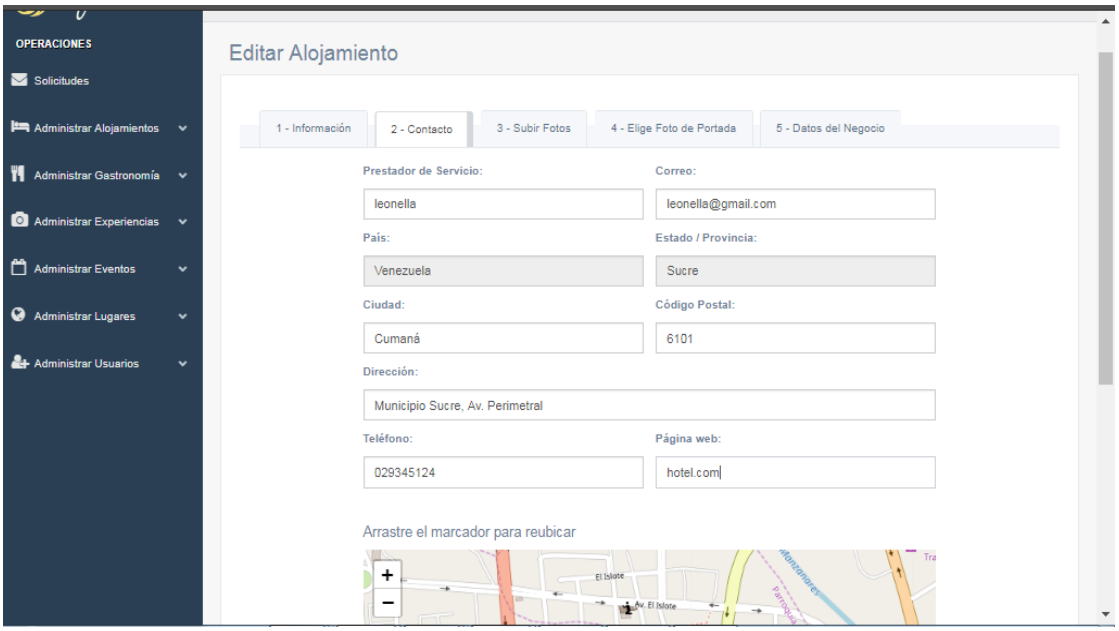


Figura U. 17. Prueba modificar alojamiento por administrador. Paso 2.

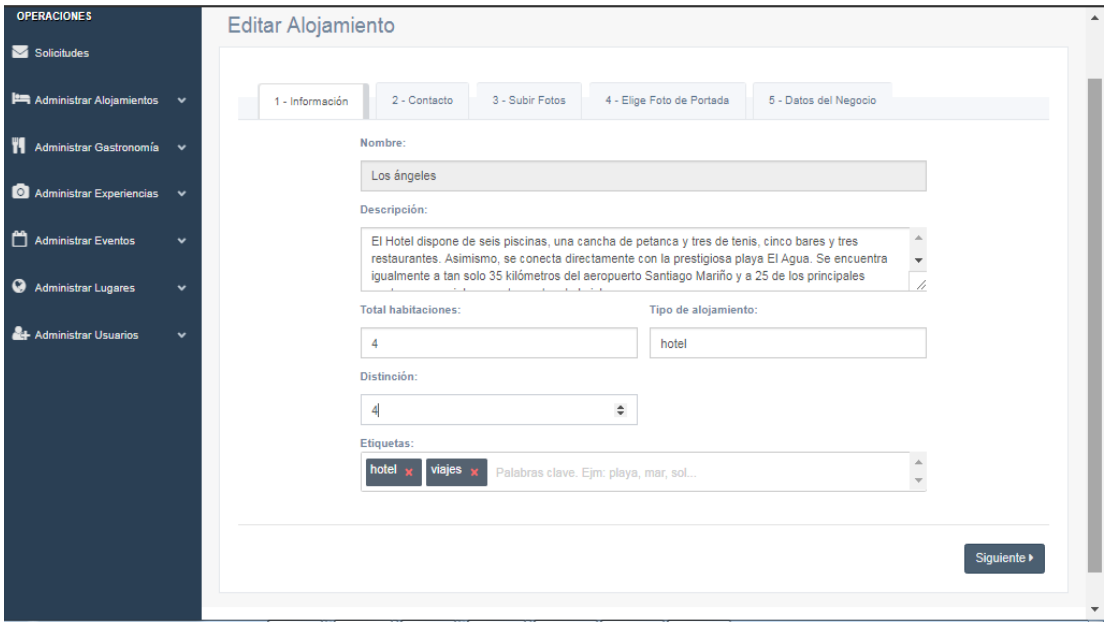


Figura U. 18.Prueba modificar alojamiento por administrador. Paso 3.

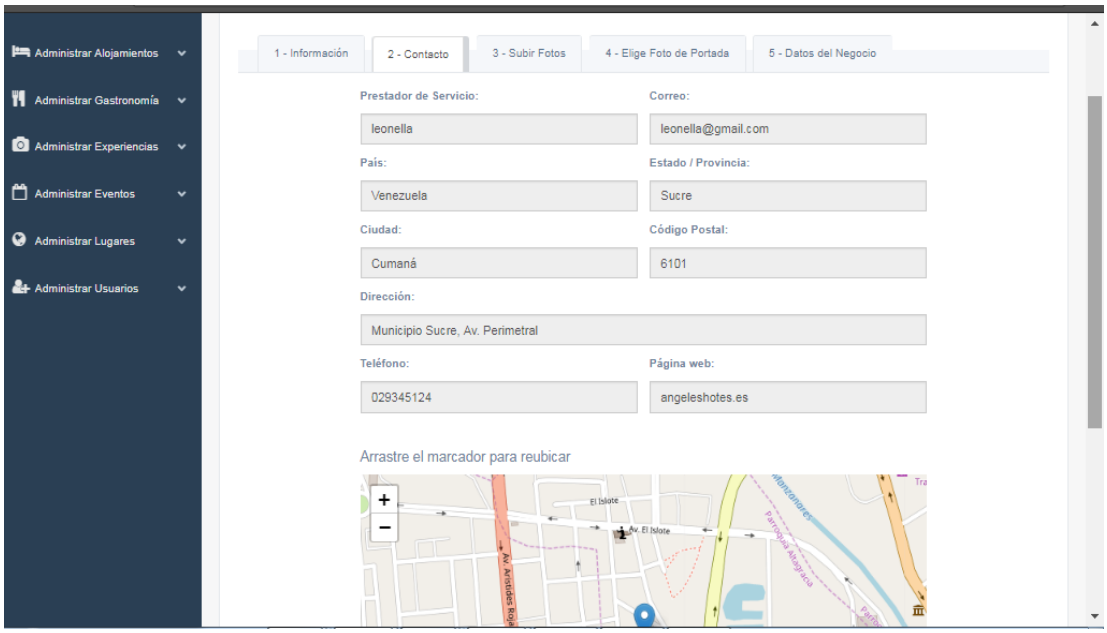


Figura U. 19. Prueba modificar alojamiento por administrador. Paso 4.

Conclusión de prueba

La función cumple con su objetivo.

Prueba de interfaz y usabilidad

La interfaz cumple con estándares de usabilidad.

Apéndice V. Ciclo de Versión Administrador: Iteración Eliminar Alojamiento.

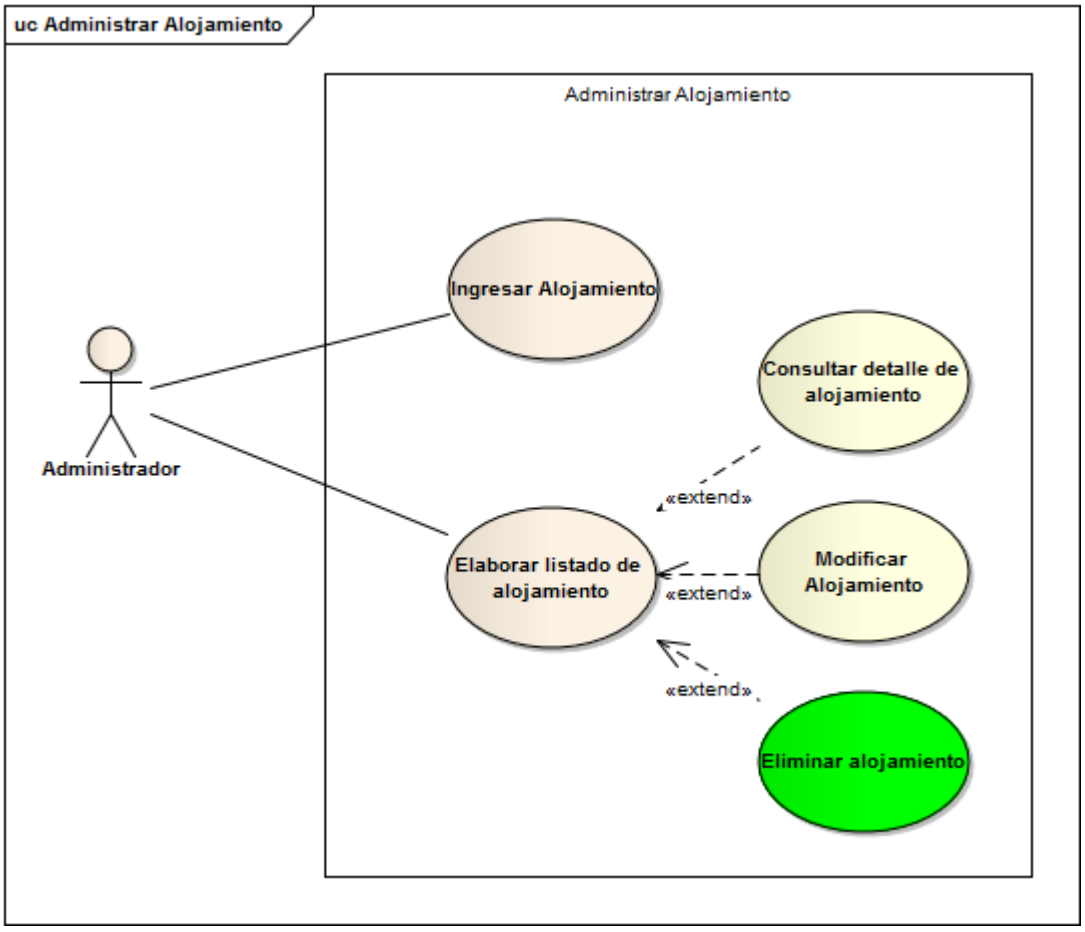


Figura V. 1.Caso de uso: Eliminar Alojamiento por administrador

Tabla V. 1. Descripción Caso de uso Eliminar Alojamiento por usuario administrador

Nombre del Caso de uso	Eliminar alojamiento.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en eliminar un alojamiento de manera satisfactoria.	
Precondición	Para eliminar un alojamiento, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Eliminar alojamiento".
	2	El sistema ejecuta la búsqueda del alojamiento.
	3	El Administrador visualiza el alojamiento que desea eliminar y selecciona eliminar.
	4	El sistema muestra un mensaje "¿Esta seguro que desea eliminar este alojamiento?"
	5	El administrador selecciona la opción "Si, estoy seguro"
	6	El sistema realiza la eliminación del alojamiento en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se elimina el alojamiento de la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	5.1.	El Administrador decide no continuar con el caso de uso y selecciona cancelar el caso de uso, seleccionando el botón cancelar.
	5.2.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. El alojamiento debe estar registrado para que pueda ser eliminado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

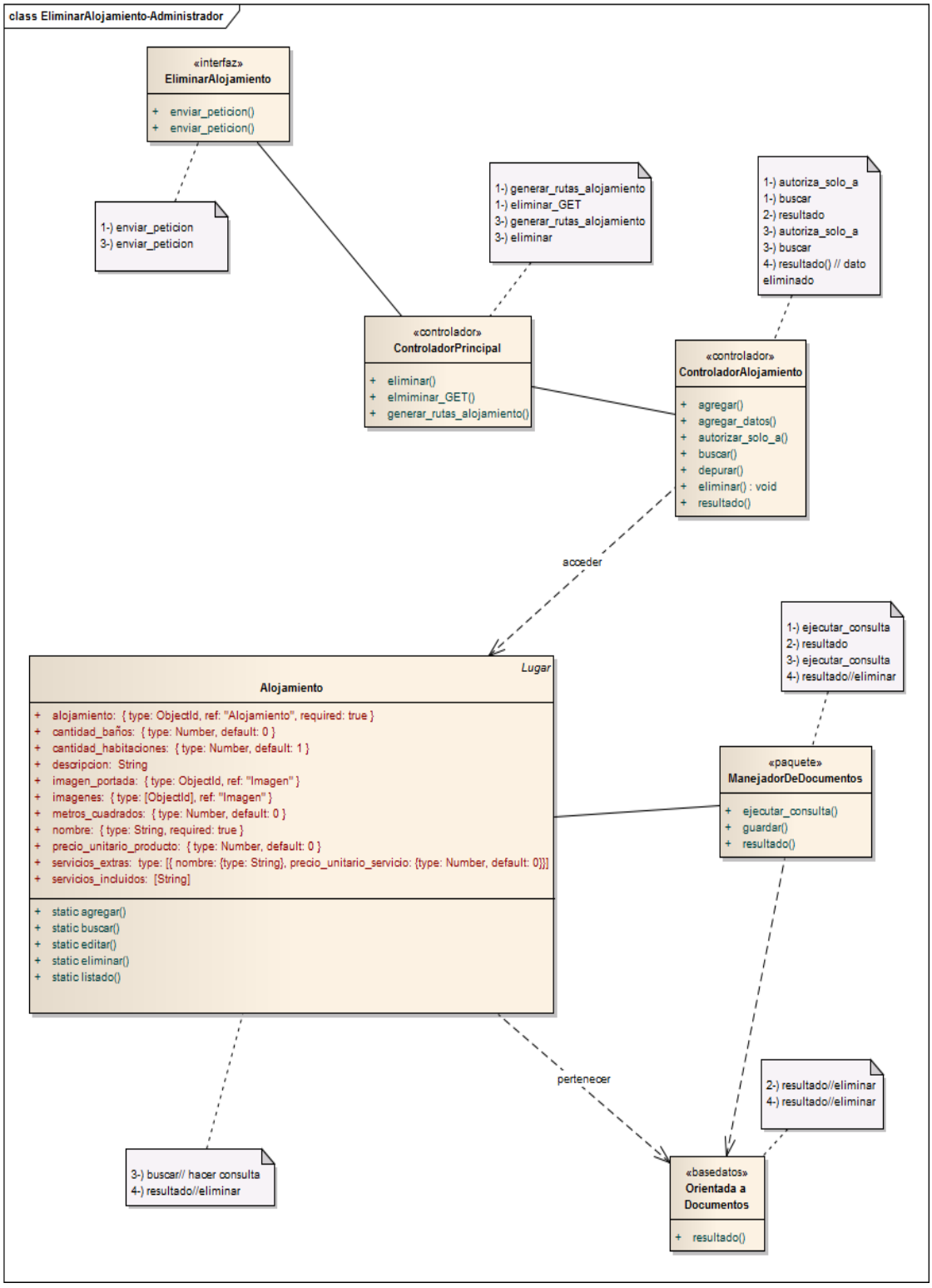


Figura V. 2.Diagrama de Clases: Eliminar Alojamiento por usuario Administrador

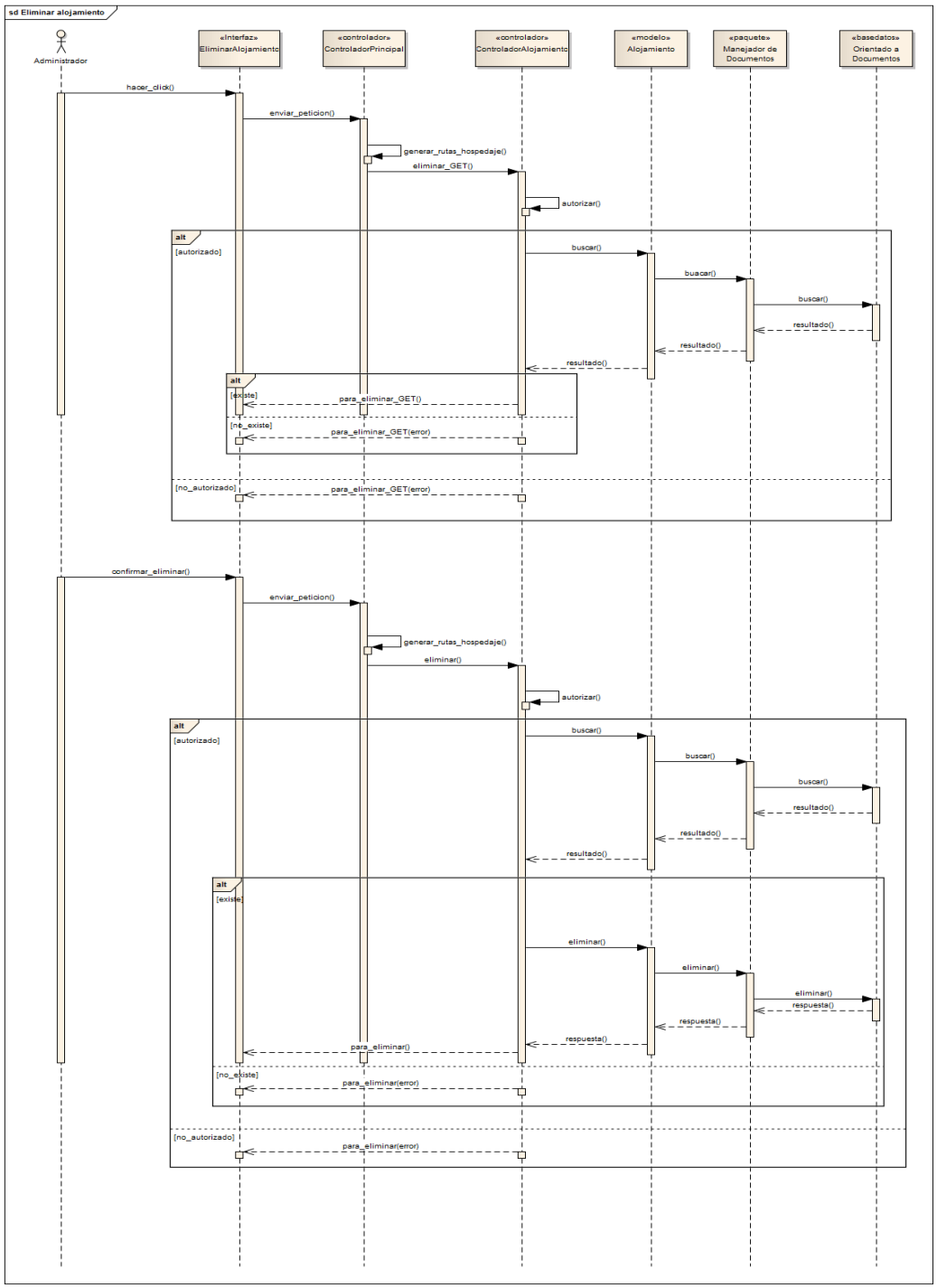


Figura V. 3.Diagrama de secuencia: Eliminar alojamiento por usuario Administrador

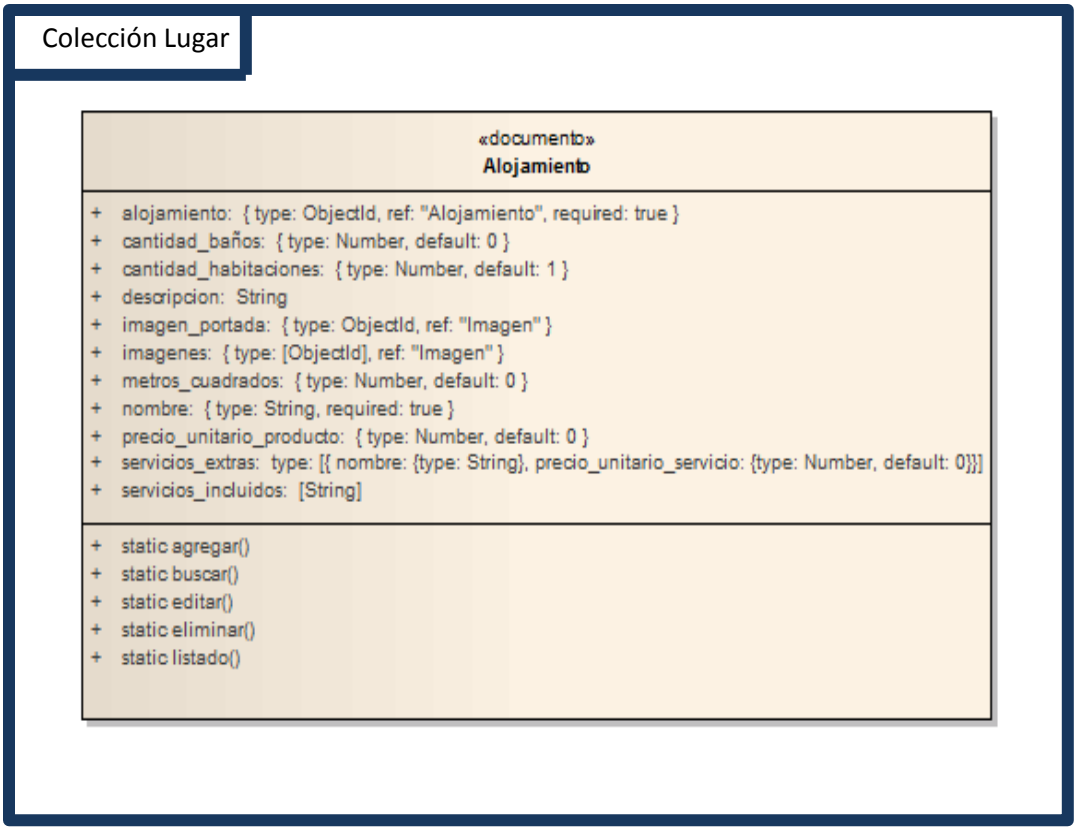


Figura V. 4. Colección de Alojamiento: Eliminar alojamiento por usuario Administrador.

A continuación, se muestran el modelo de plantilla para el proceso de eliminar alojamiento.

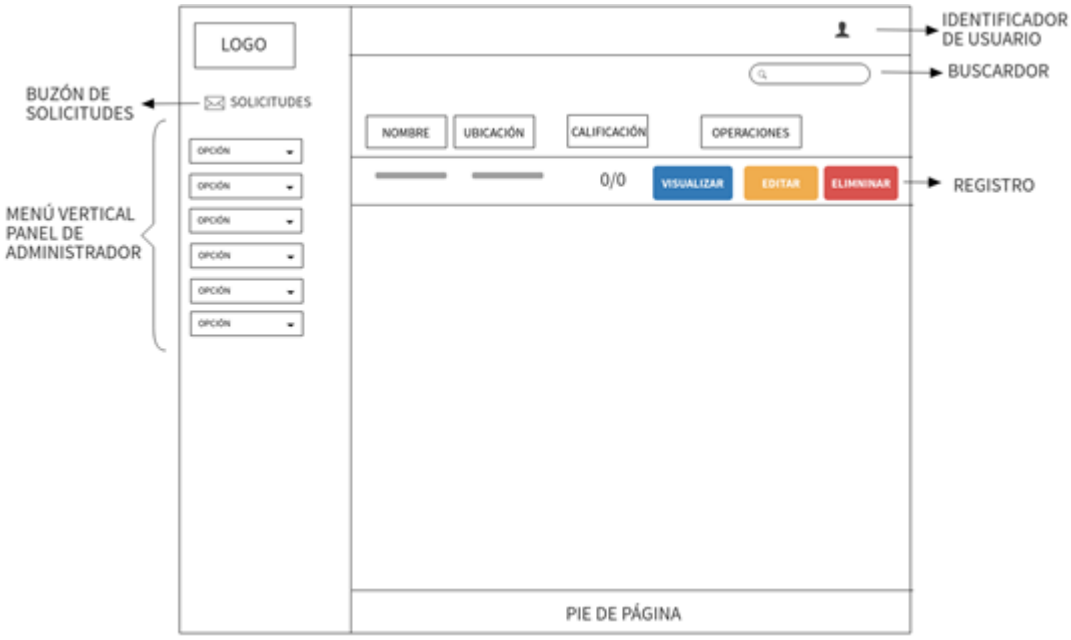


Figura V. 5.Plantilla de interfaz: Eliminar Alojamiento por usuario administrador.

Figura V. 6.Plantilla de interfaz: Eliminar Alojamiento por usuario administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Eliminar Alojamiento y el sistema responde mostrando la interfaz de confirmación de acción.

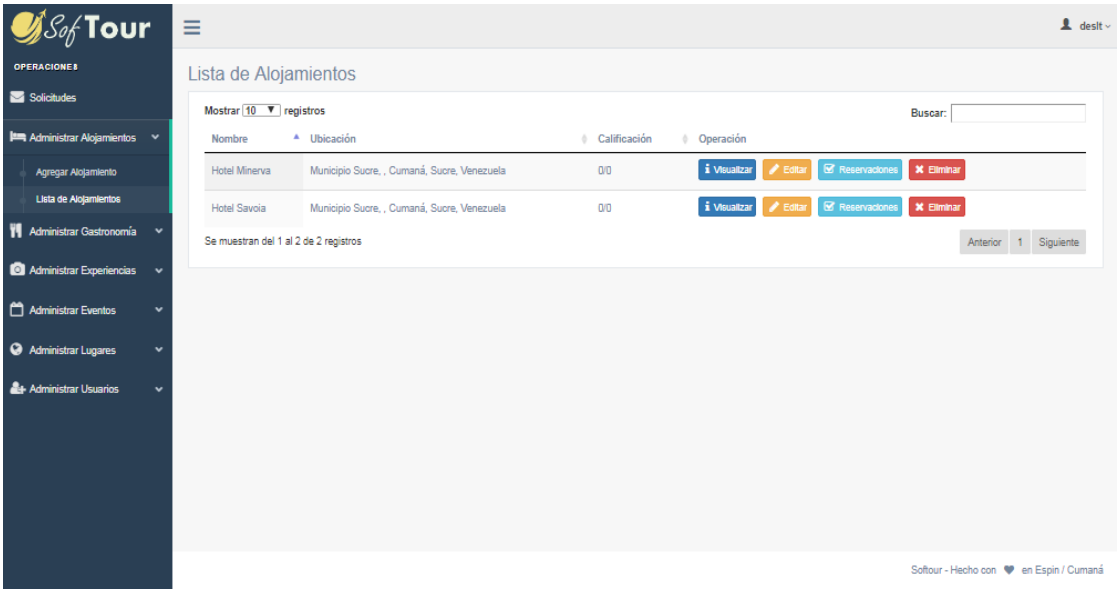


Figura V. 7.Interfaz 1 de proceso de Eliminar Alojamiento por usuario Administrador

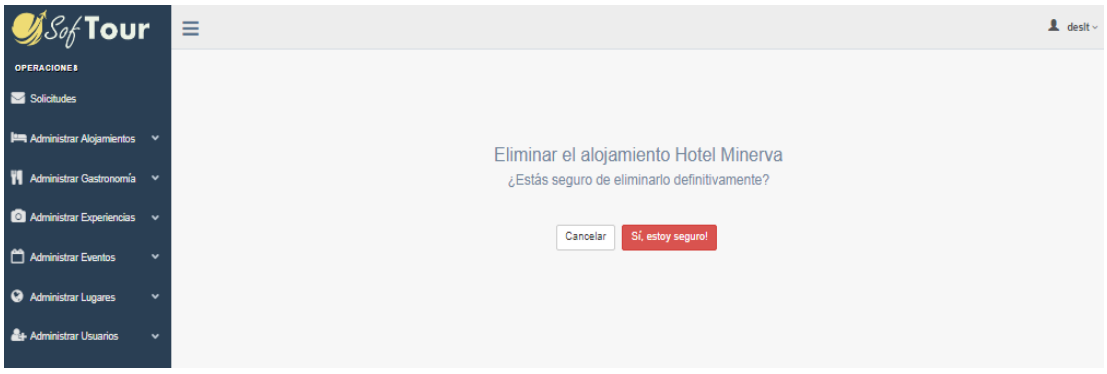


Figura V. 8.Interfaz 1 de proceso de Eliminar Alojamiento por usuario Administrador

Parte del código desarrollado para la funcionalidad Eliminar Alojamiento por usuario Administrador

```
eliminar(req, res, next) {
    let tipo_entidad = req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.
        substring(1);

    this._Lugar.eliminar({"url": req.params.url.toLowerCase(), "__t": tipo_entidad}, (error, eliminado) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_eliminar(req, res);
        }
        if(!eliminado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": {"url": "listado"}};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_eliminar(req, res);
    });
}
```

Figura V. 9. Código desarrollado para el proceso Eliminar Alojamiento por usuario Administrador.

Apéndice W. Ciclo de Versión Administrador: Iteración Ingresar Gastronomía.

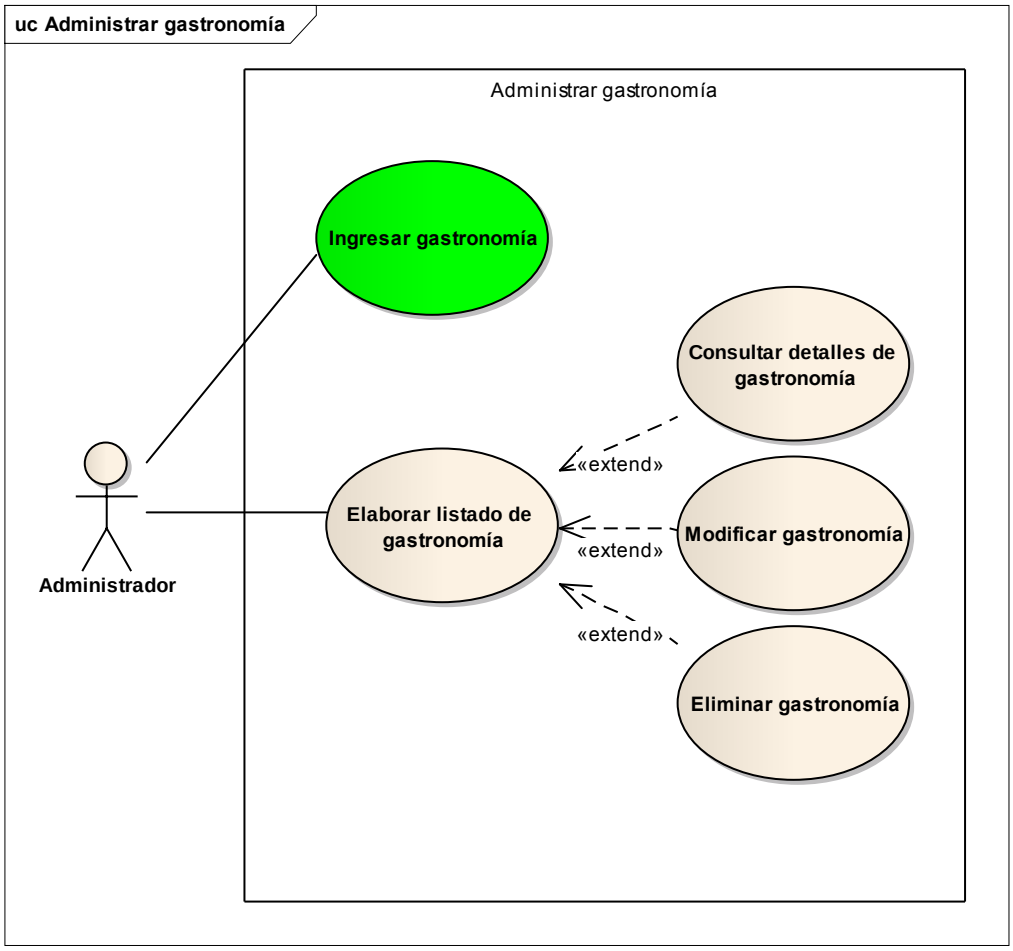


Figura W. 1.Caso de uso: Ingresar Gastronomía por Administrador

Tabla W. 1. Descripción Caso de uso Ingresar Gastronomía por usuario Administrador

Nombre del Caso de uso	Ingresar gastronomía.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en ingresar los atributos de la gastronomía de manera satisfactoria.	
Precondición	Para ingresar una gastronomía, no debe estar registrada en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Ingresar gastronomía".
	2	El sistema muestra el formulario correspondiente.
	3	El Administrador ingresar los atributos de la gastronomía.
	4	El sistema almacena los atributos de la gastronomía y finaliza el caso de uso.
Postcondición	Se ingresa los atributos de la gastronomía en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no ingresar la gastronomía y selecciona cancelar el caso de uso, ya sea seleccionando el botón cerrar o el botón cancelar.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. Para que una gastronomía pueda ser ingresada, no debe estar registrada en la base de datos del sistema.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

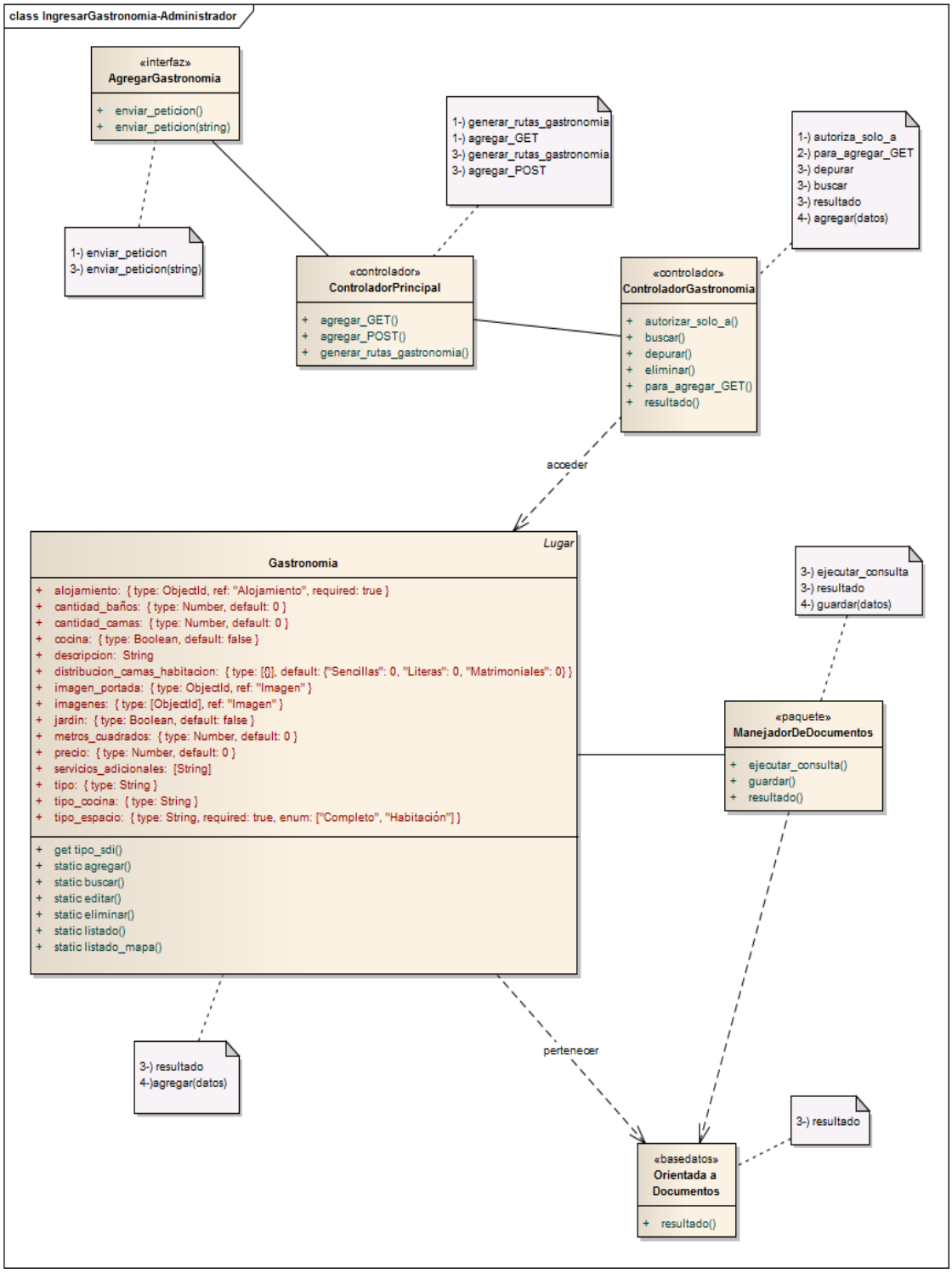


Figura W. 2. Diagrama de Clases: Ingresar Gastronomía por usuario Administrador

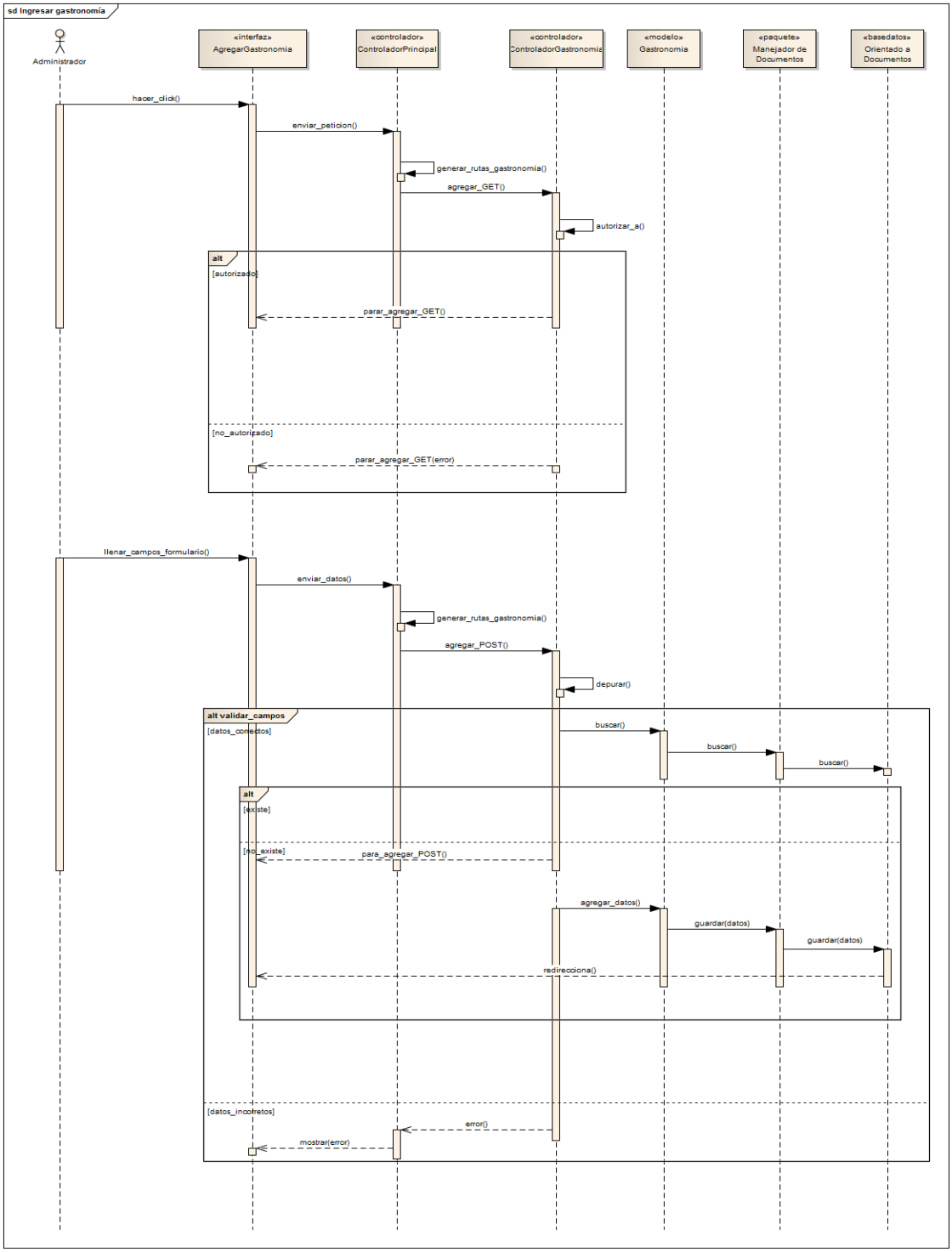


Figura W. 3. Diagrama de secuencia: Ingresar Gastronomía por usuario Administrador

Figura W. 4. Colección Gastronomía 1: Ingresar Gastronomía por usuario Administrador.

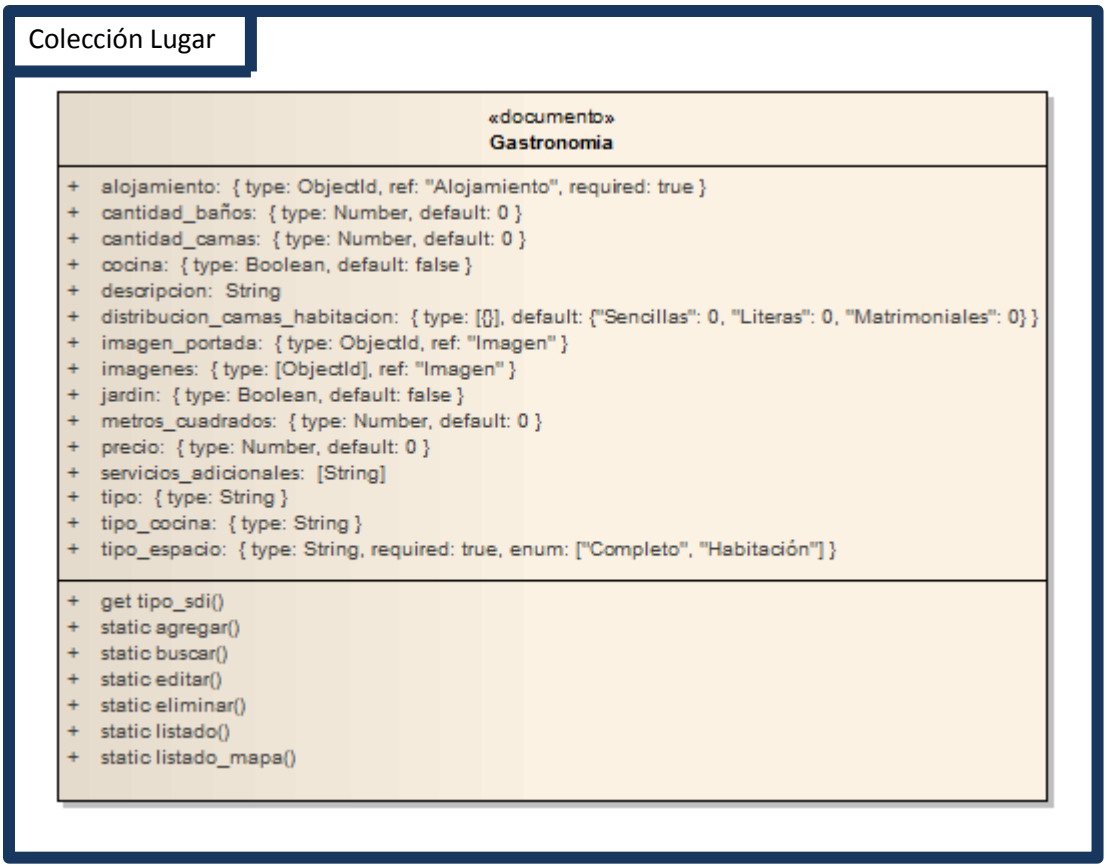


Figura W. 5. Colección Gastronomía 2: Ingresar Gastronomía por usuario Administrador.

A continuación, se muestran las plantillas utilizadas para realizar la interfaz de esta función.

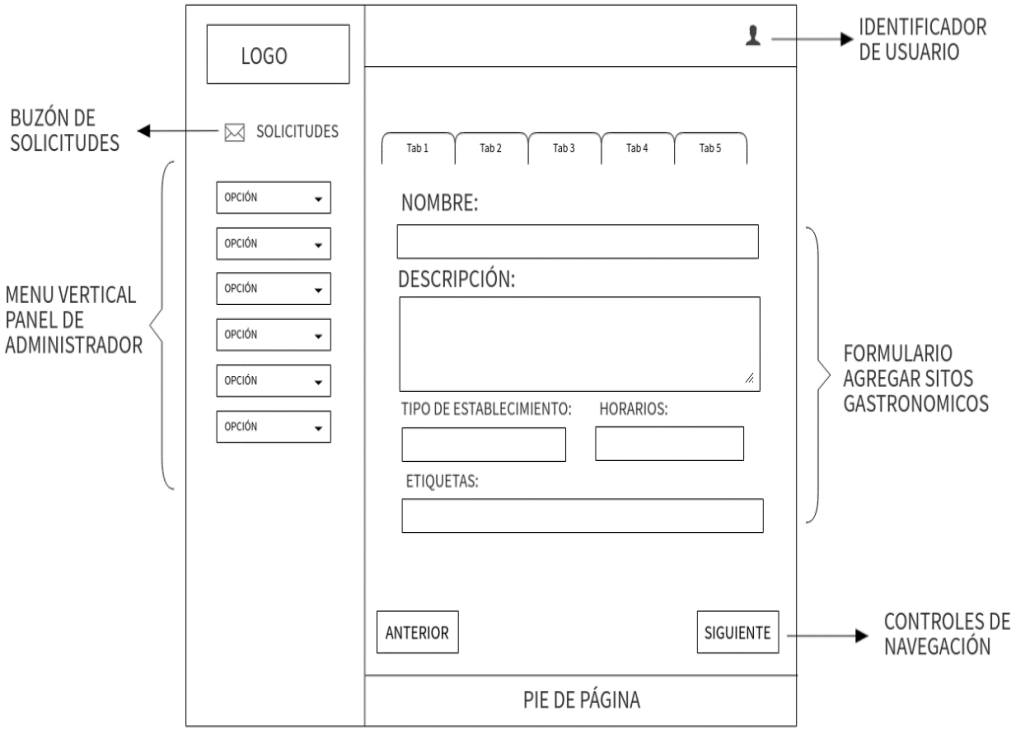


Figura W. 6. Plantilla de interfaz 1: Ingresar Gastronomía por usuario administrador.

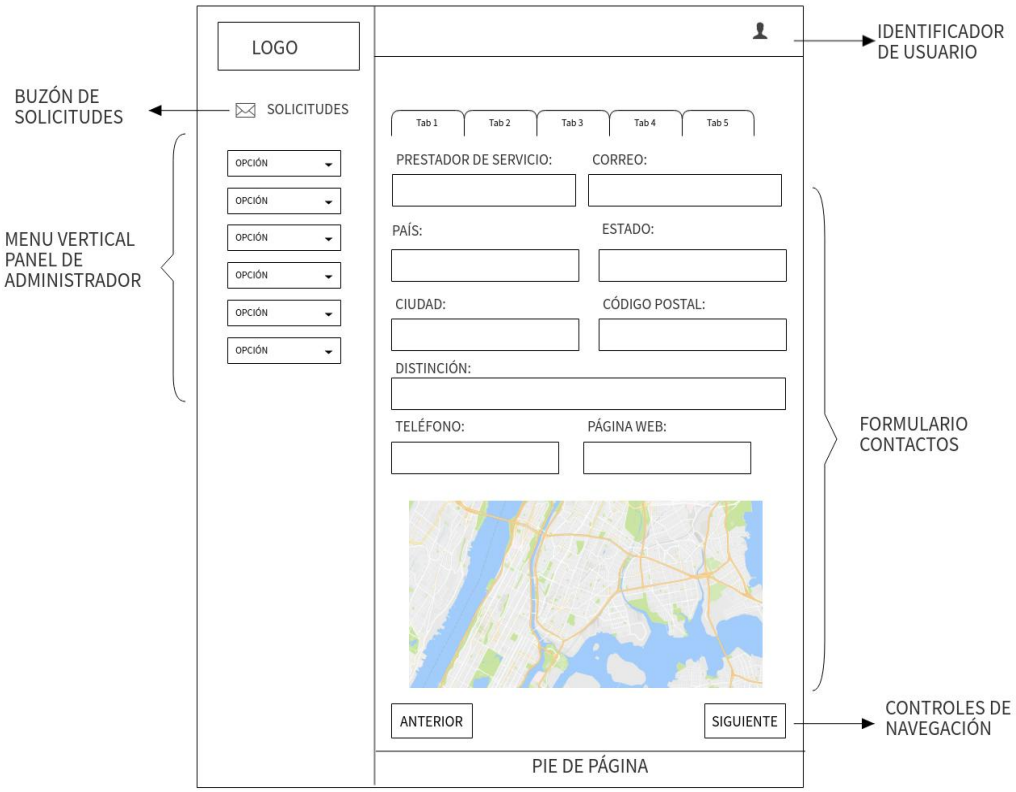


Figura W. 7. Plantilla de interfaz 2: Ingresar Gastronomía por usuario administrador.

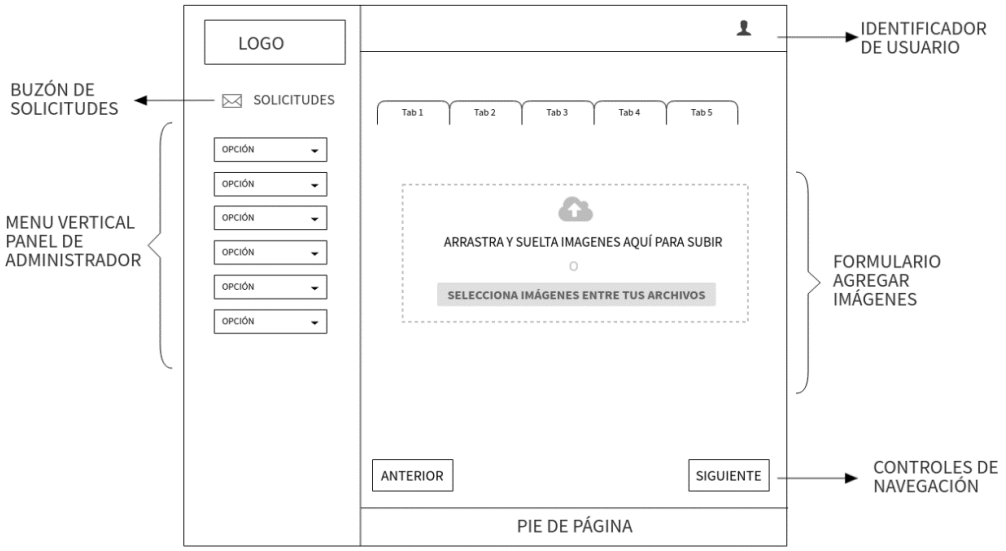


Figura W. 8. Plantilla de interfaz 3: Ingresar Gastronomía por usuario administrador.

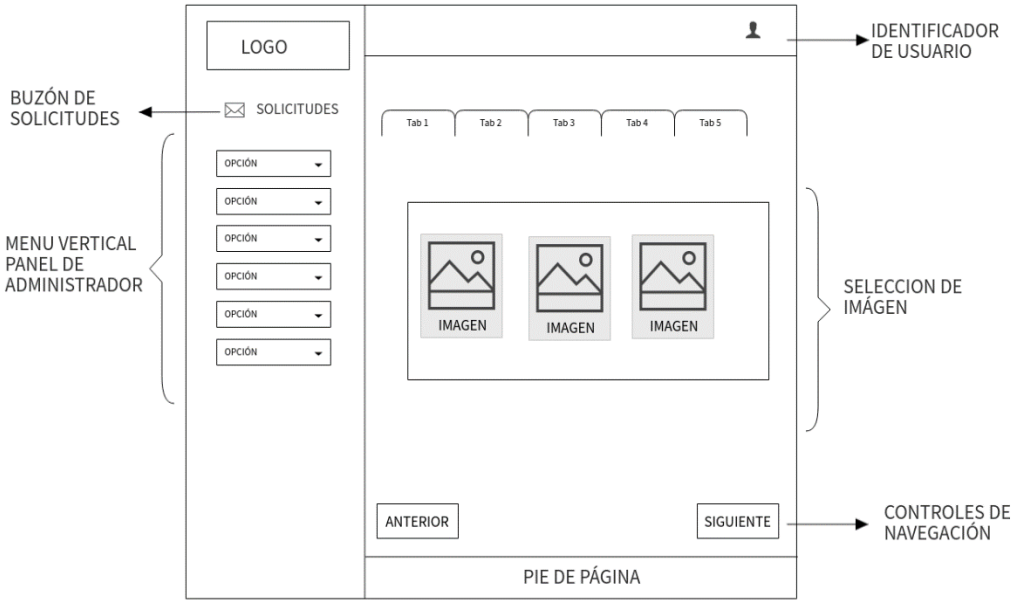


Figura W. 9. Plantilla de interfaz 4: Ingresar Gastronomía por usuario administrador.

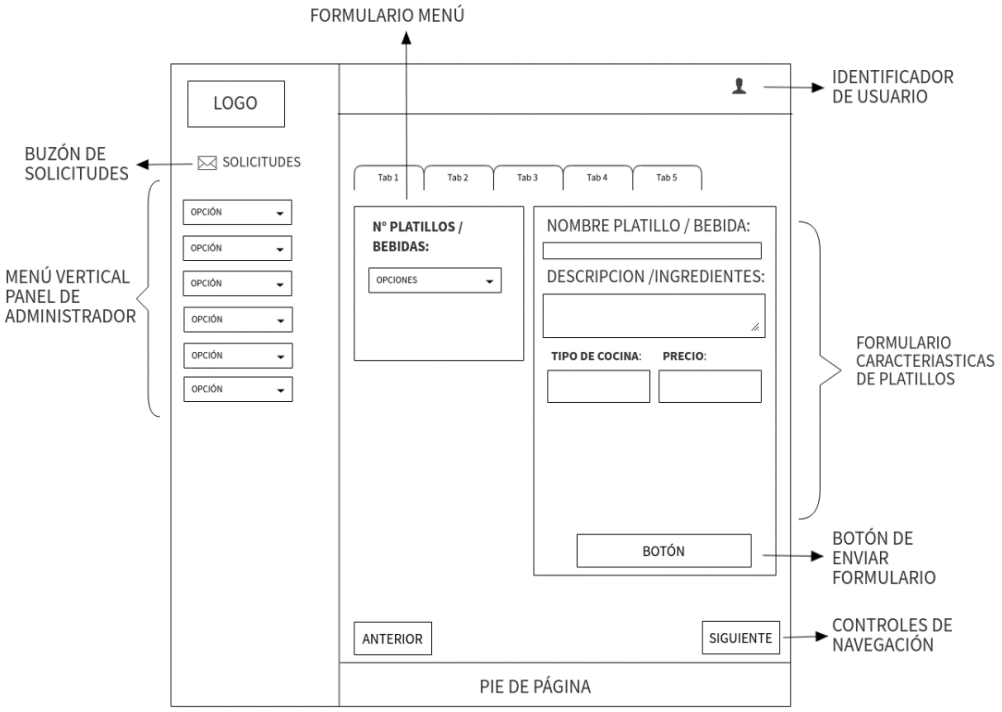


Figura W. 10. Plantilla de interfaz 5: Ingresar Gastronomía por usuario administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Agregar Gastronomía y el sistema responde mostrando el formulario correspondiente.

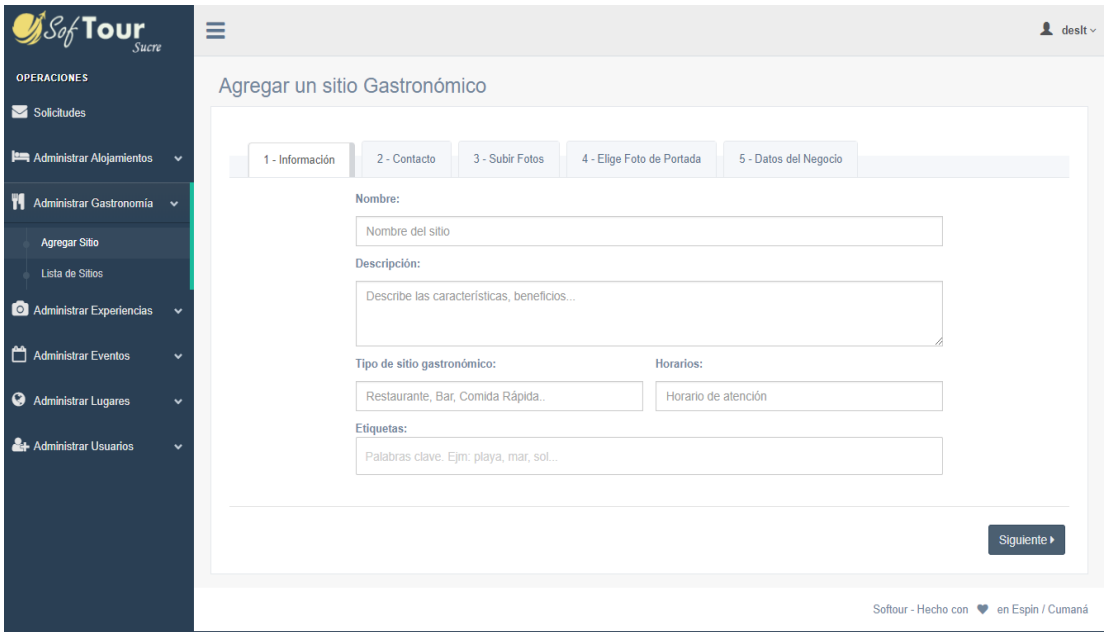


Figura W. 11. Interfaz 1 de proceso de Agregar Gastronomía por usuario Administrador

El administrador ingresa los atributos característicos, lo cual se visualiza en diferentes pestañas: Información, Contacto, Subir fotos, Elige foto de portada y Datos del negocio.

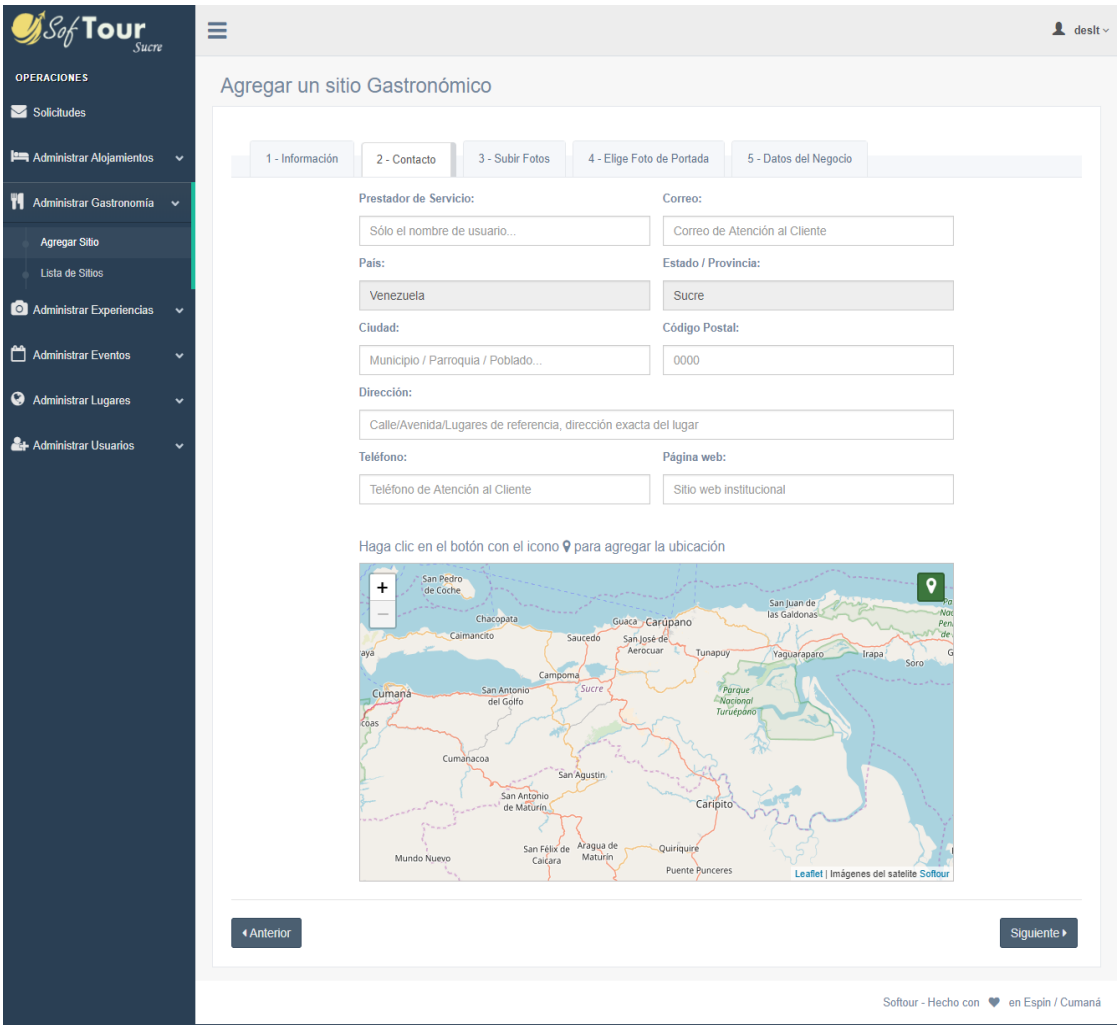


Figura W. 12. Interfaz 2 de proceso de Agregar Gastronomía por usuario Administrador

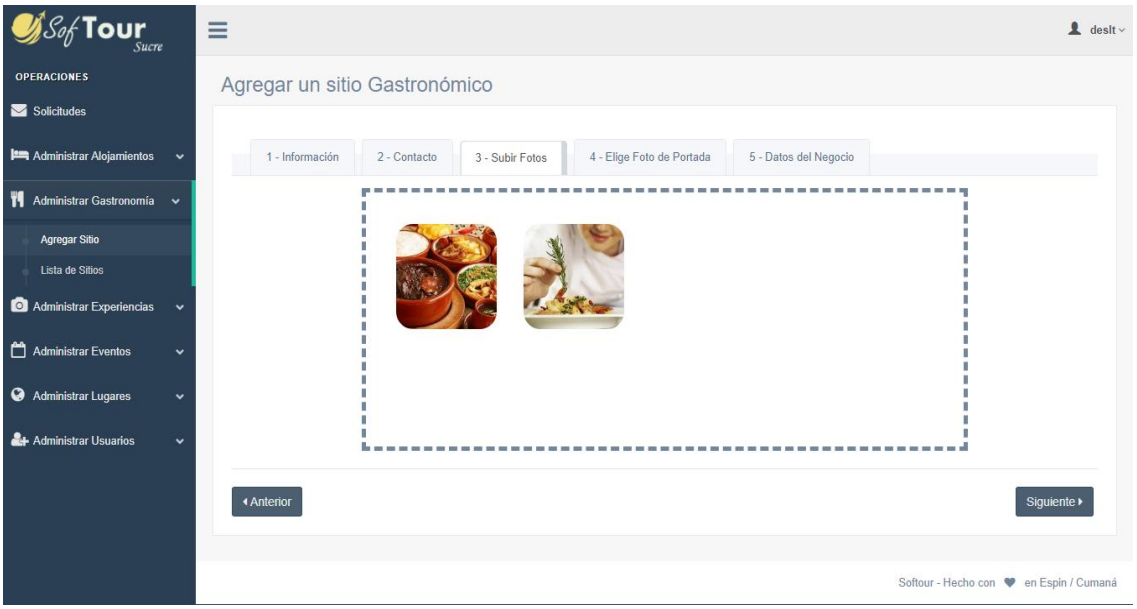


Figura W. 13. Interfaz 3 de proceso de Agregar Gastronomía por usuario Administrador

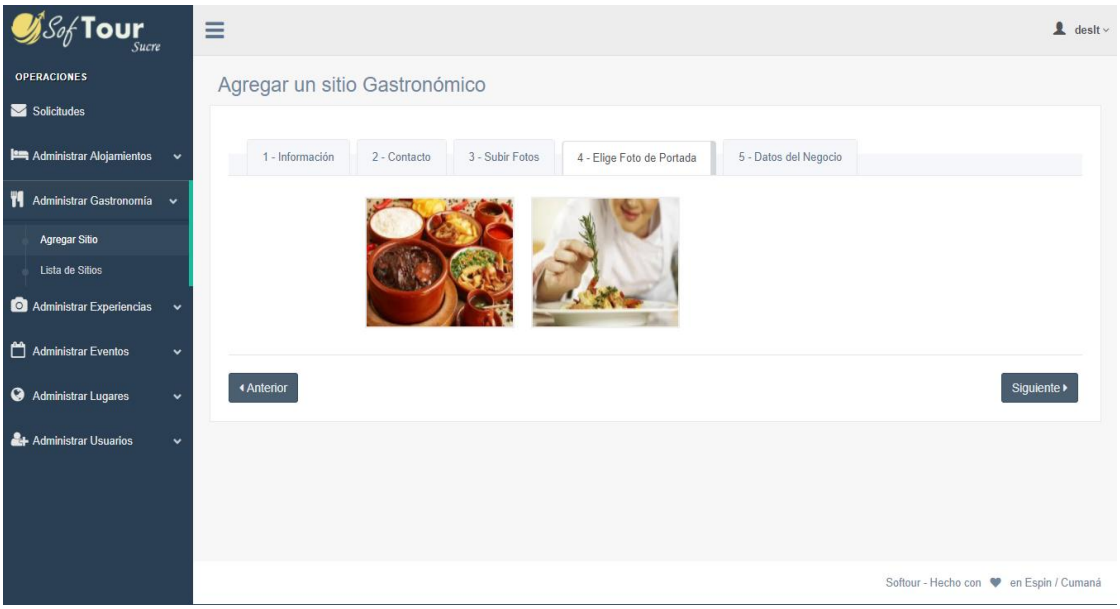


Figura W. 14. Interfaz 4 de proceso de Agregar Gastronomía por usuario Administrador

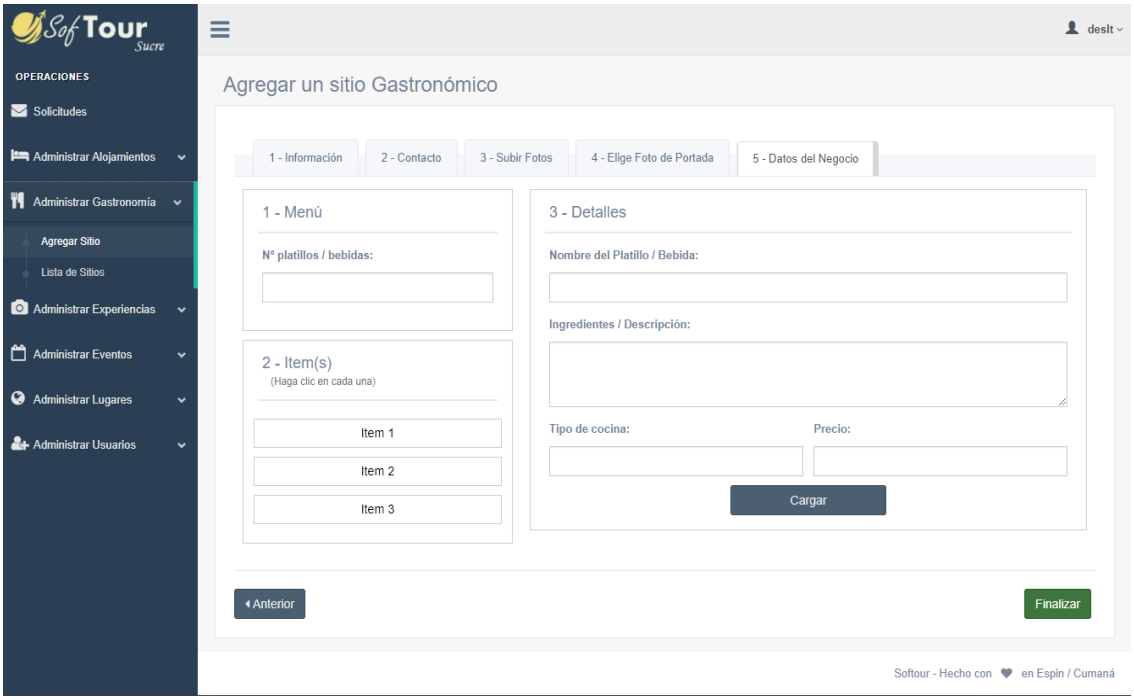


Figura W. 15. Interfaz 5 de proceso de Agregar Gastronomía por usuario Administrador

Código desarrollado para la funcionalidad Agregar Gastronomía por usuario Administrador

```
function constructorVentanas(e) { // una ventana es un form de una categoria
    let ventanas = [],
        forms_categ = [];
    $('#contenedor-categorias').empty();

    $('#categoria-habitaciones').empty();
    for (let i = 1; i <= $('input[name="cantidad_categorias"]').val(); i++) {
        ventanas.push('<a href="#detalles" class="ventana-categoria col-md-5 col-sm-5 col-lg-5" id="' + i + '">C ' + i + '</a>');
        forms_categ.push('<form class="form-horizontal clase-formulario-categorias" id="C' + i + '">' + form_espacio_habitacion + '</
            form>');
    };
    $('#categoria-habitaciones').append(ventanas);
    $('#contenedor-categorias').append(forms_categ);
    inicializarEtiquetador(".input-etiquetas", "Wifi, Piscina, Restaurant...");

    // mostrar solo el form q quiero ver
    $('.ventana-categoria').click(function(e) {
        $(".clase-formulario-categorias").hide();
        $(".clase-formulario-categorias").filter("[id=C" + $(this).attr("id") + "]").show();
    });

    // cargar datos al pulsar el btn
    $('.btn-guardar-categoria').click(function(e) {
        e.preventDefault();
        let contenedor = $(this).parent(),
            f = $(contenedor).attr("id").slice(1);
        $(contenedor).parsley({excluded: 'a, input[type=button], input[type=submit], input[type=reset]', inputs: 'input, textarea,
            select, input[type=hidden], :hidden'})
            .whenValidate()
            .done(() => {
                $(contenedor).find("input, textarea, .input-etiquetas").attr("readonly", "readonly");
                let tagsinput = $(contenedor).find("[name='servicios_incluidos']").siblings();

                $(tagsinput).css("background-color", "#eee");
                $(tagsinput).find("div").css("background-color", "#eee");
                $(tagsinput).find(".tag a").hide();
                $(this).attr("disabled", "disabled");

                new PNotify({
                    title: 'Cargado',
                    text: 'Ingrese siguiente categoría',
                    styling: "bootstrap3",
                    delay: 10000,
                    type: "success",
                    sticker: false
                });

                lista_categorias[parseInt(f) - 1] = $(contenedor).serialize() + "&id_alojamiento=" + id_alojamiento;
            });
    });
}

$('select[name="tipo_espacio"]')
    .ready(function(e) {
        lista_categorias = [];
```

Figura W. 16. Código desarrollado para el proceso Ingresar Gastronomía por usuario Administrador.

Apéndice X. Ciclo de Versión Administrador: Iteración Modificar Gastronomía.

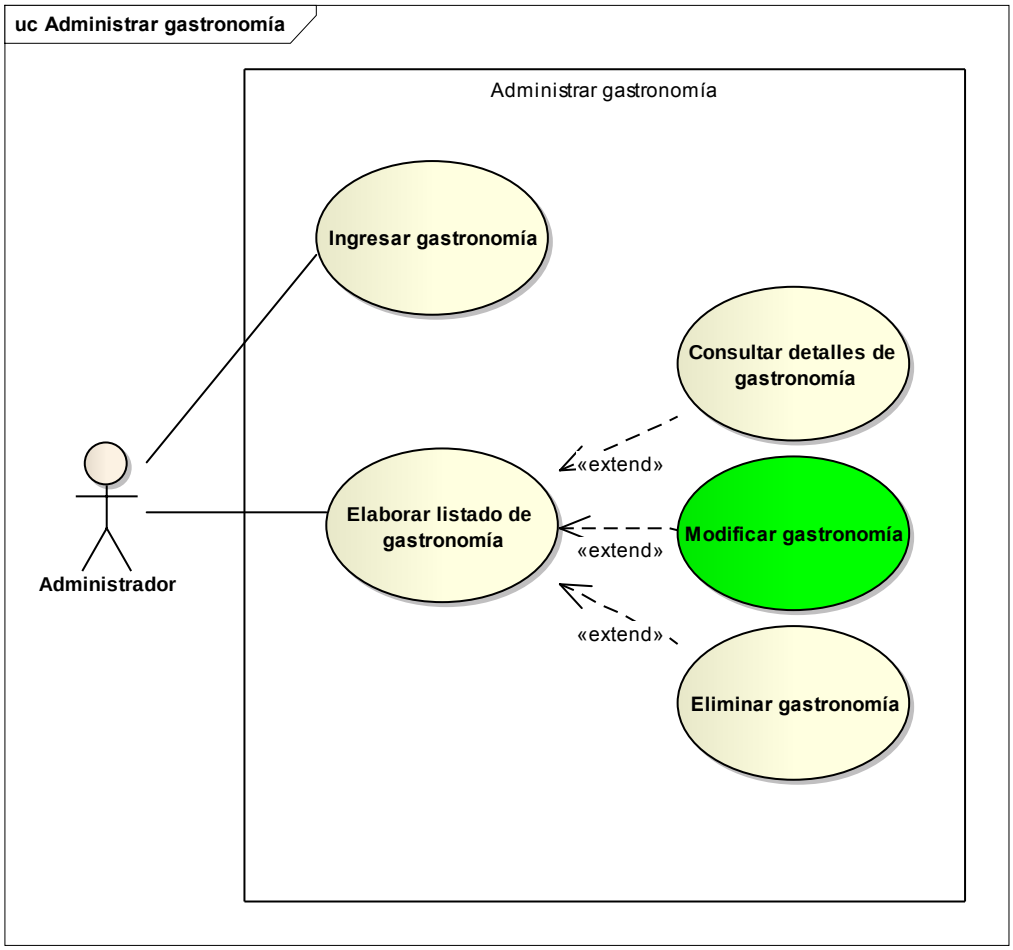


Figura X. 1.Caso de uso: Modificar Gastronomía por administrador

Tabla X. 1. Descripción Caso de uso Modificar Gastronomía por usuario administrador

Nombre del Caso de uso	Modificar gastronomía.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en modificar los atributos de la gastronomía de manera satisfactoria.	
Precondición	Para modificar una gastronomía, debe estar registrada en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Modificar gastronomía”.
	2	El sistema ejecuta la búsqueda de la gastronomía, mostrando un formulario con la consulta en modo edición.
	3	El Administrador visualiza la consulta de los atributos de la gastronomía que desea modificar, los modifica y selecciona aceptar.
	4	El sistema realiza las modificaciones en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se modifican los atributos de la gastronomía en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no modificar la gastronomía y selecciona cancelar el caso de uso, ya sea seleccionando el botón cerrar o el botón cancelar.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. La gastronomía debe estar registrada para que pueda ser modificada.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

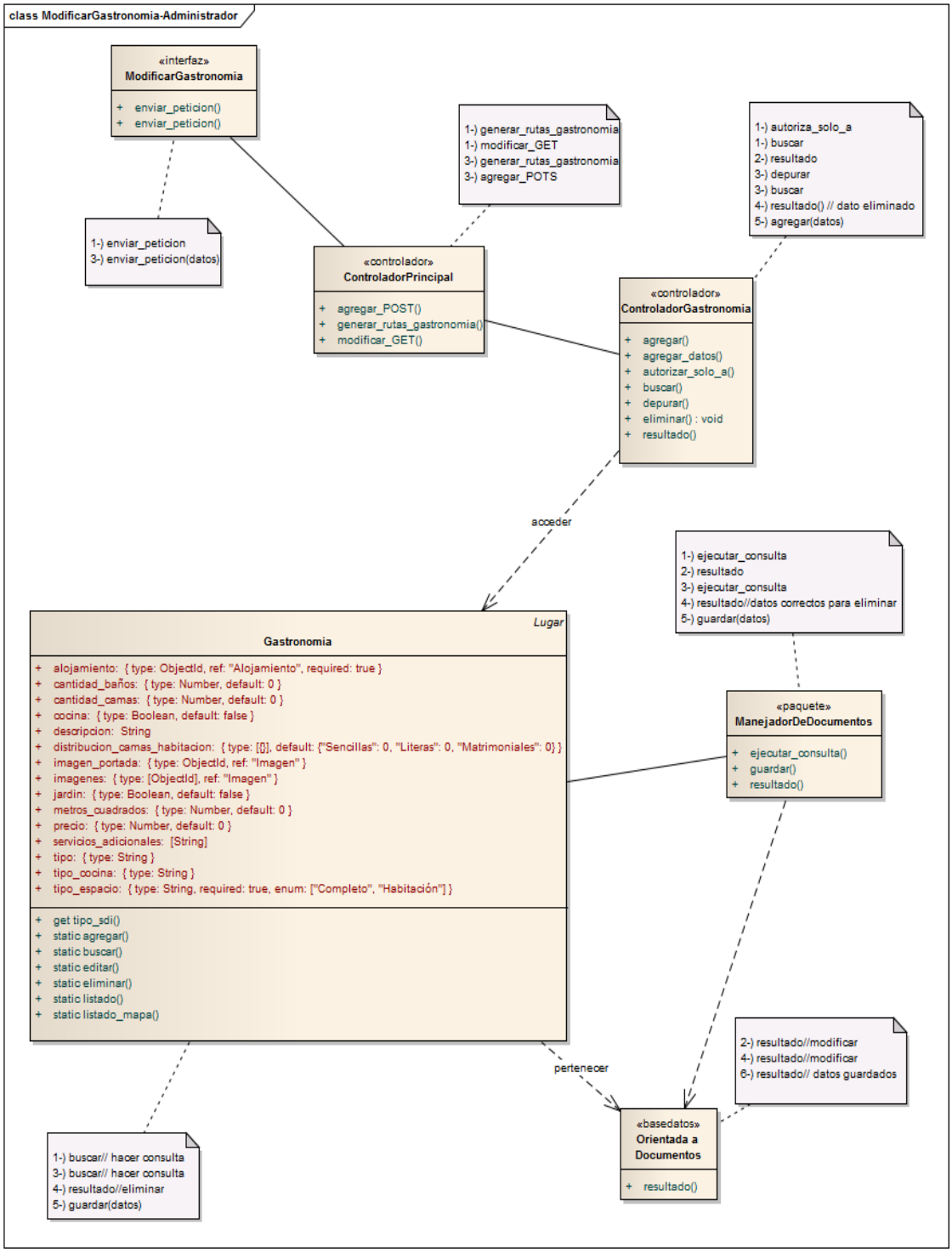


Figura X. 2. Diagrama de Clases: Modificar Gastronomía por usuario Administrador

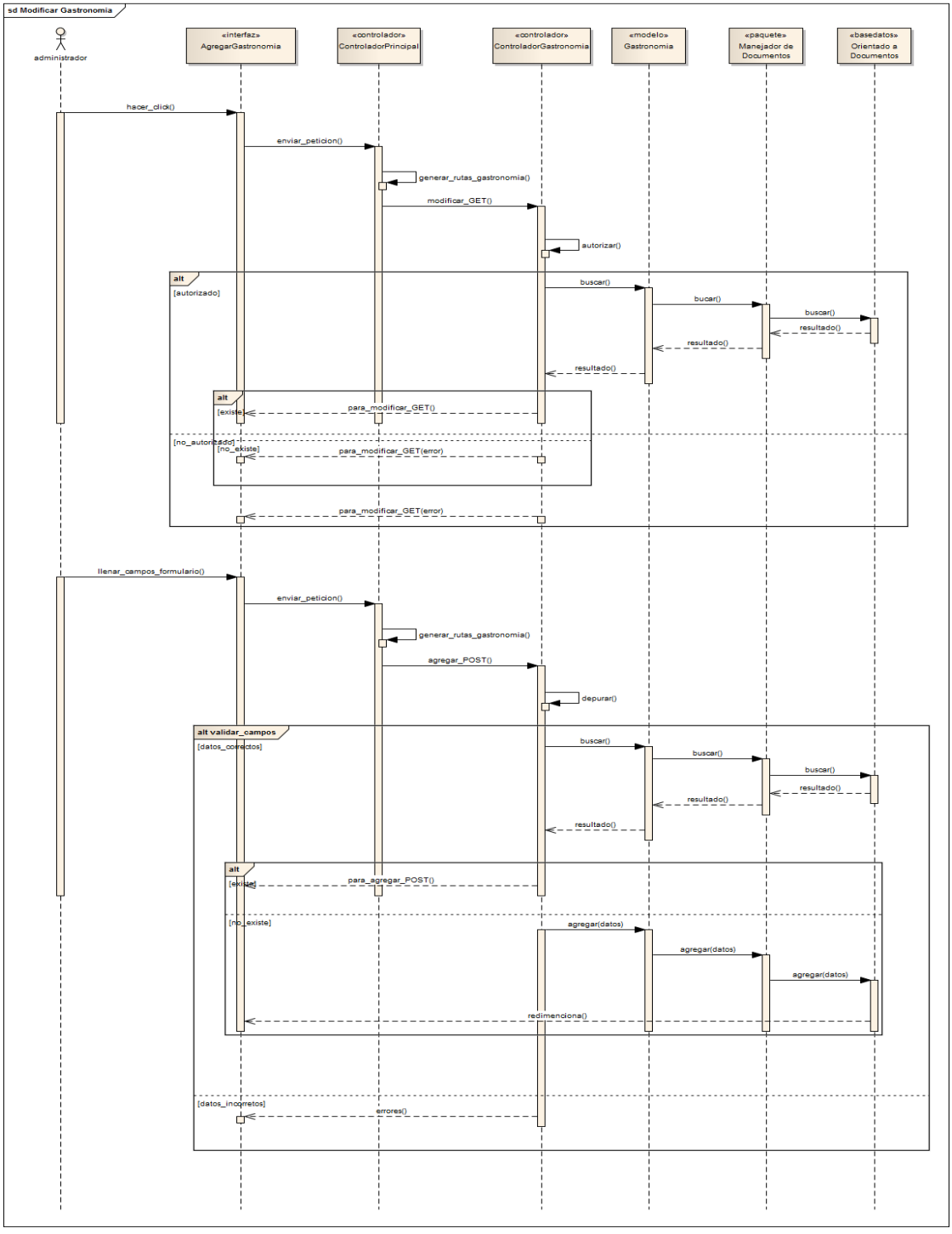


Figura X. 3. Diagrama de secuencia: Modificar Gastronomía por usuario administrador

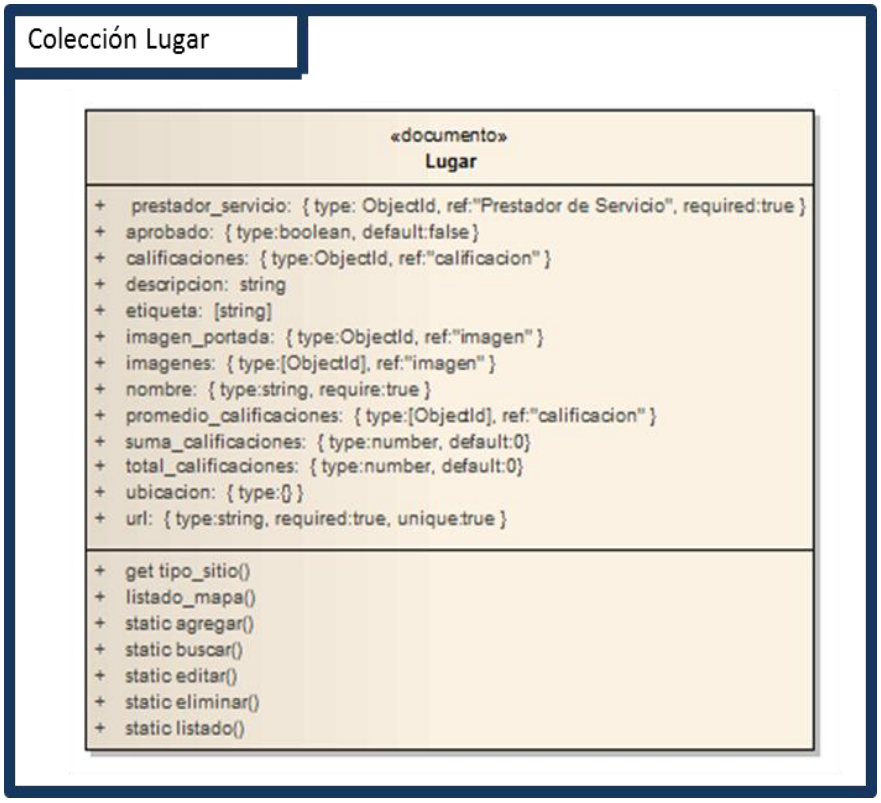


Figura X. 4. Colección Gastronomía 1: Modificar Gastronomía por usuario Administrador.

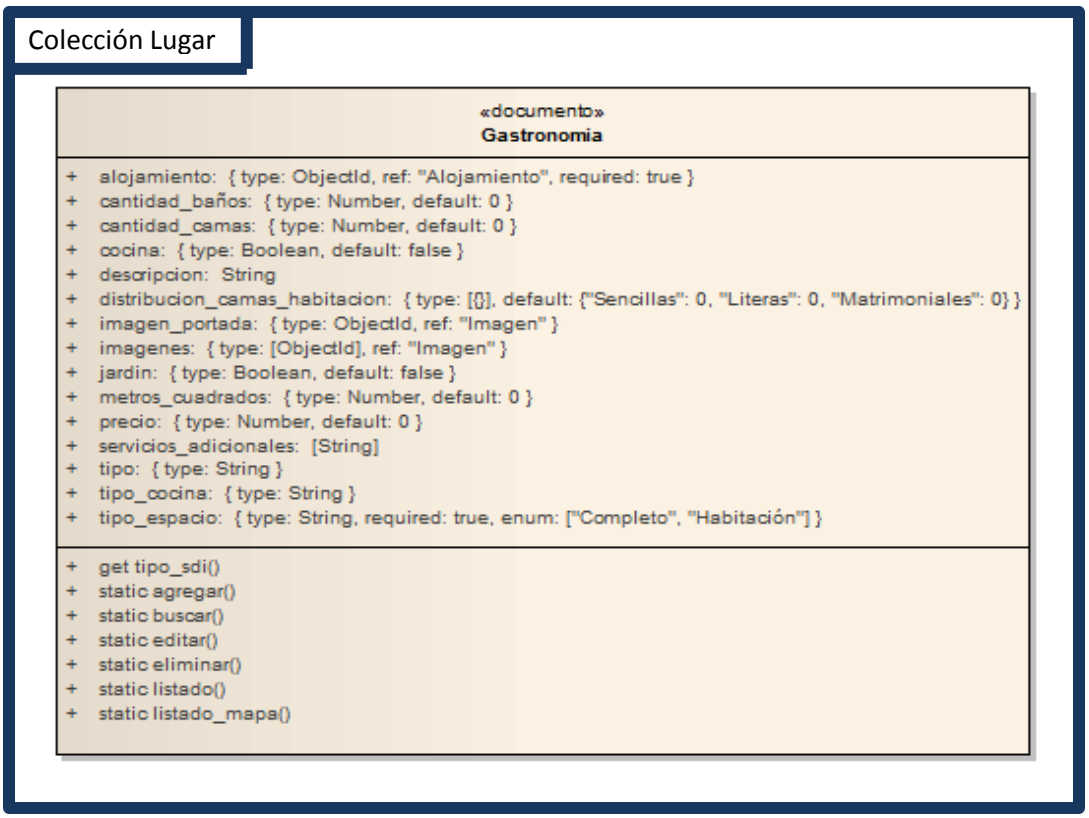


Figura X. 5. Colección Gastronomía 2: Modificar Gastronomía por usuario Administrador

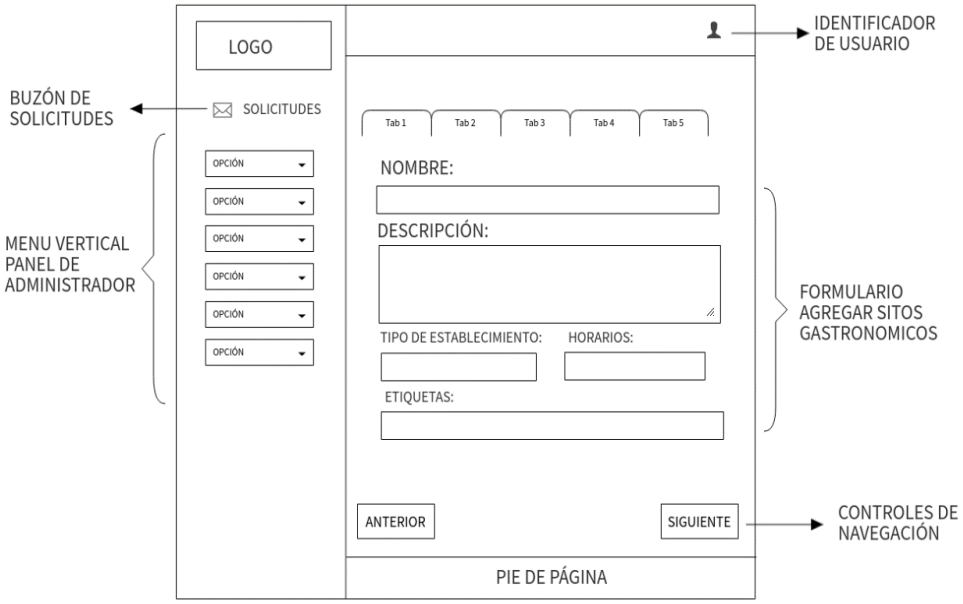


Figura X. 6. Plantilla de interfaz 1: Modificar Gastronomía por usuario administrador.

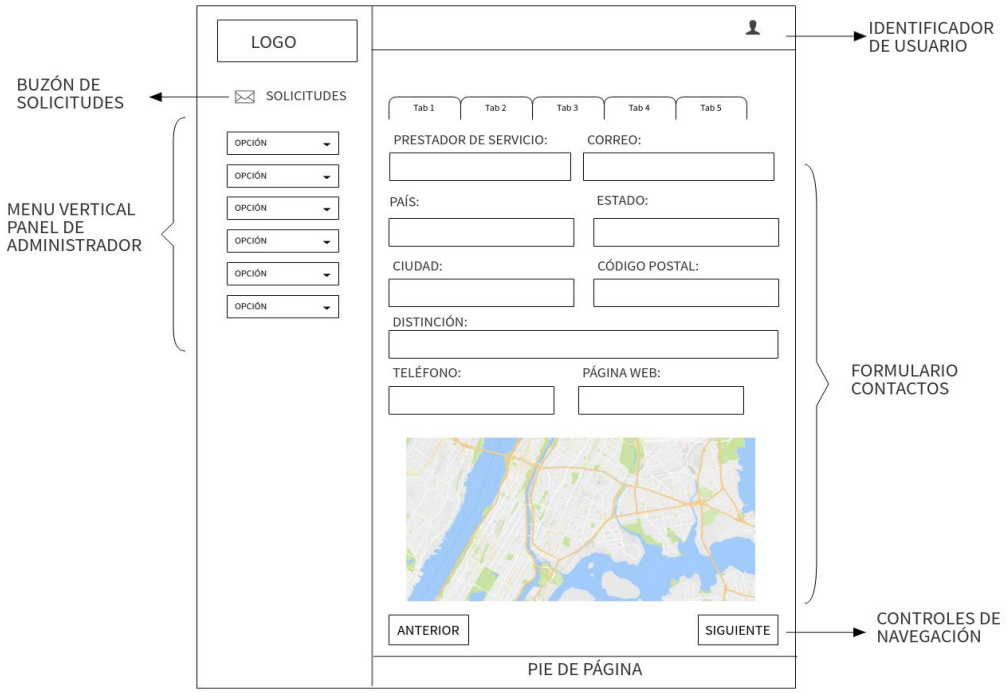


Figura X. 7. Plantilla de interfaz 2: Modificar Gastronomía por usuario administrador.

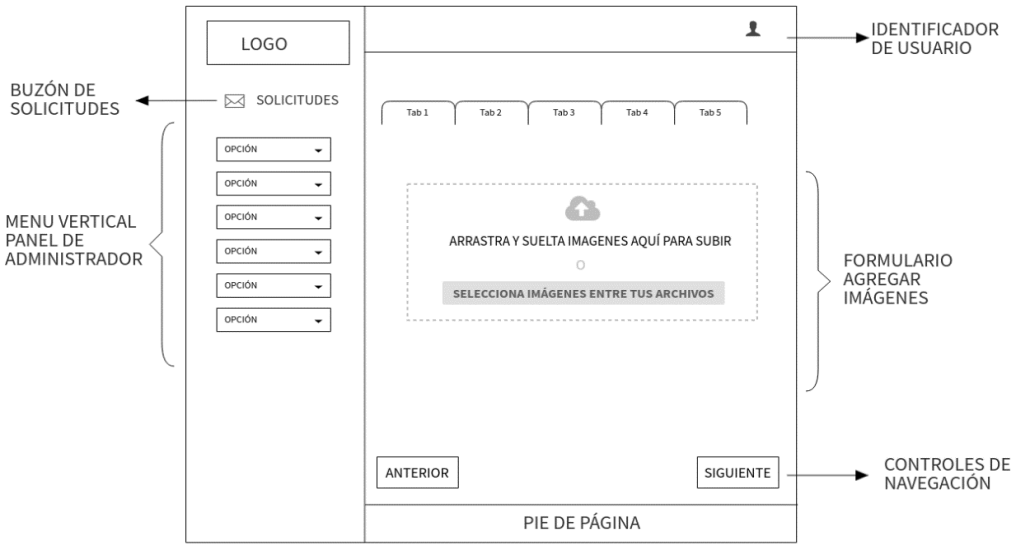


Figura X. 8. Plantilla de interfaz 3: Modificar Gastronomía por usuario administrador.

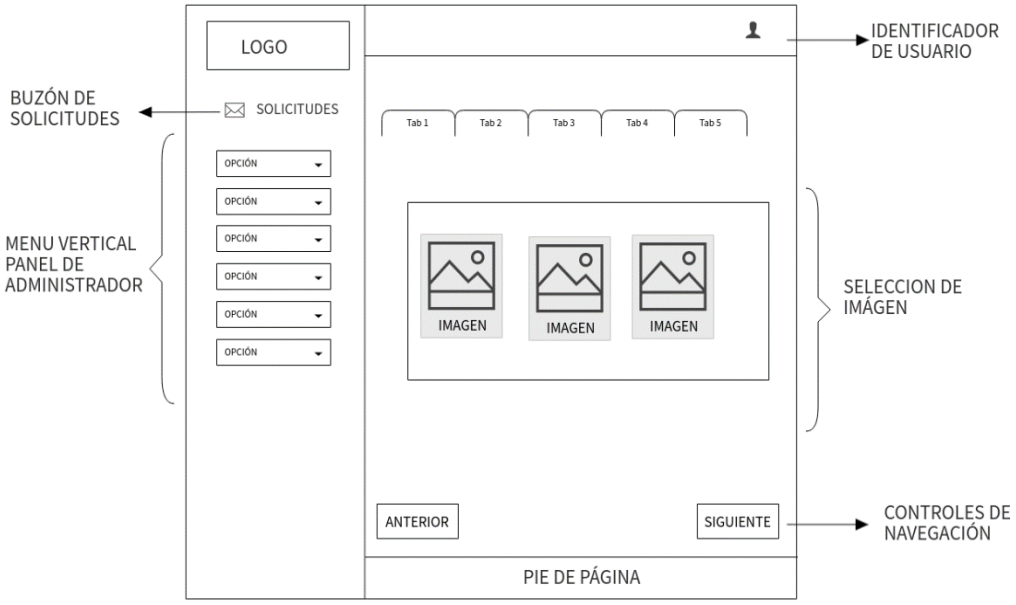


Figura X. 9. Plantilla de interfaz 4: Modificar Gastronomía por usuario administrador.

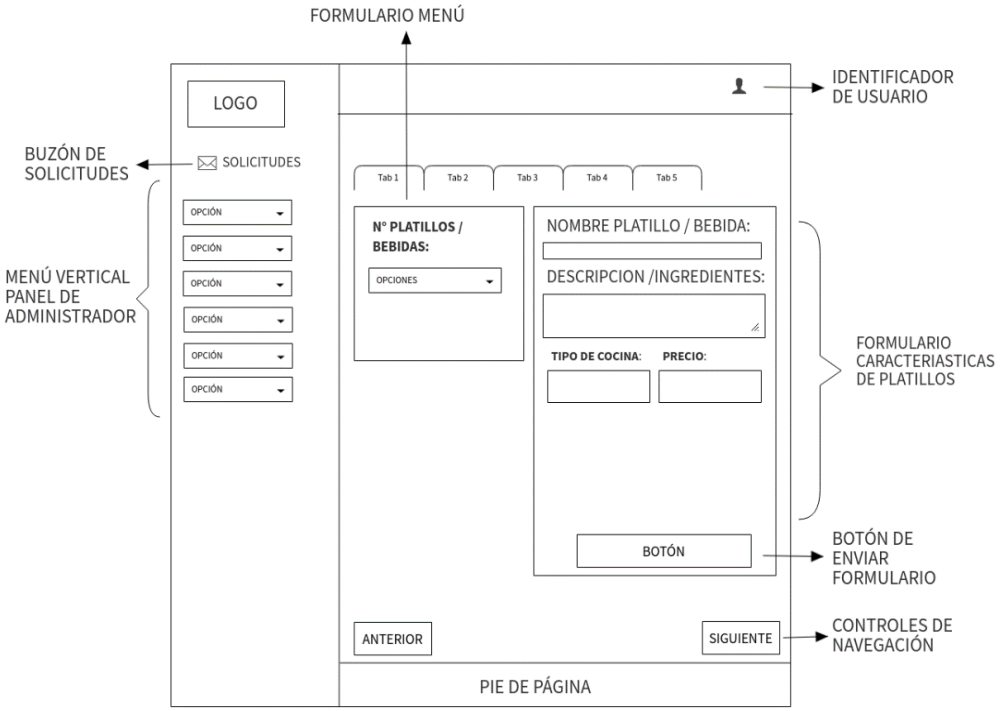


Figura X. 10. Plantilla de interfaz 5: Modificar Gastronomía por usuario administrador.

Flujo a seguir para la funcionalidad

El Administrador selecciona la opción Modificar Gastronomía y el sistema responde mostrando el formulario correspondiente.

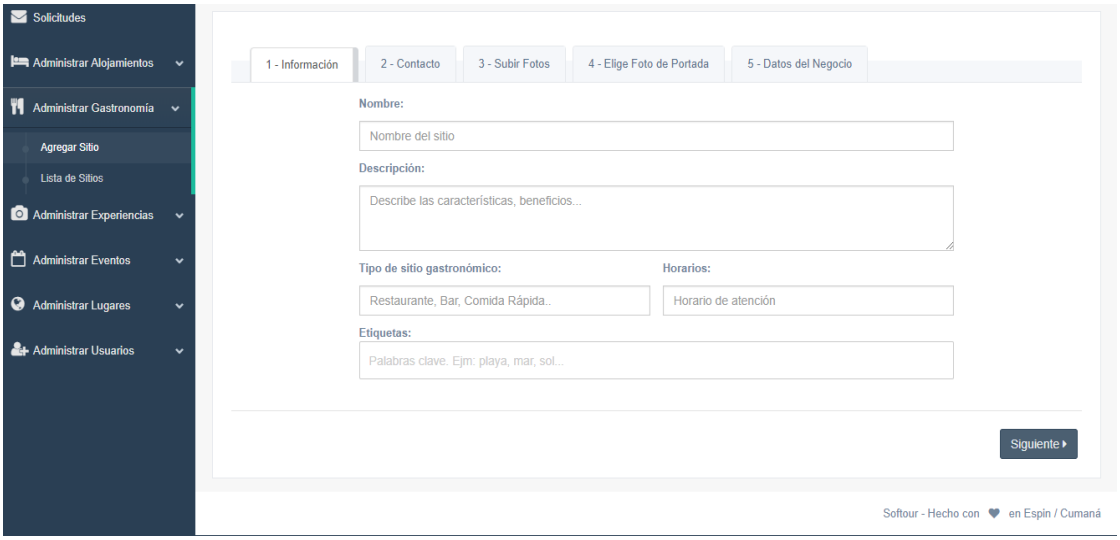


Figura X. 11.Interfaz 1 de proceso de Modificar Gastronomía por usuario Administrador

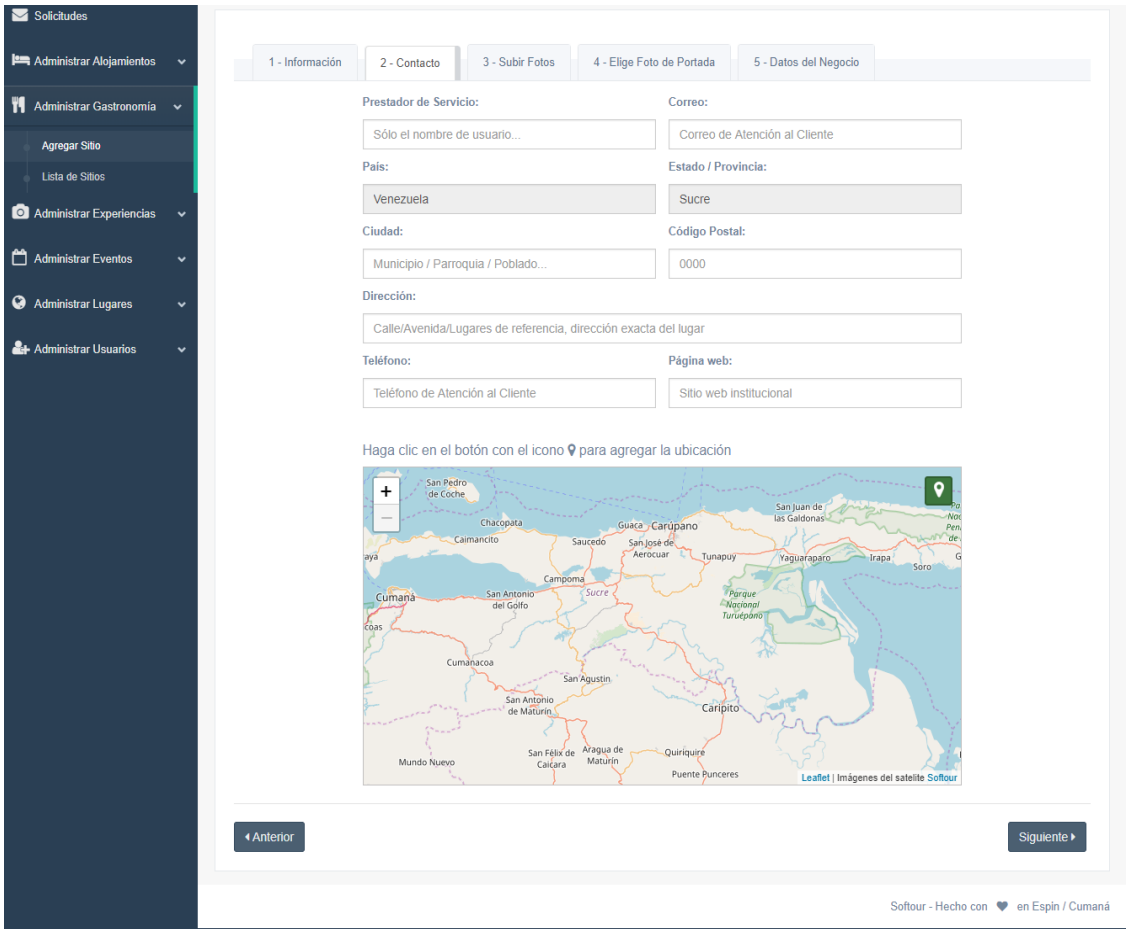


Figura X. 12.Interfaz 2 de proceso de Modificar Gastronomía por usuario Administrador

Figura X. 13.Interfaz 3 de proceso de Modificar Gastronomía por usuario Administrador

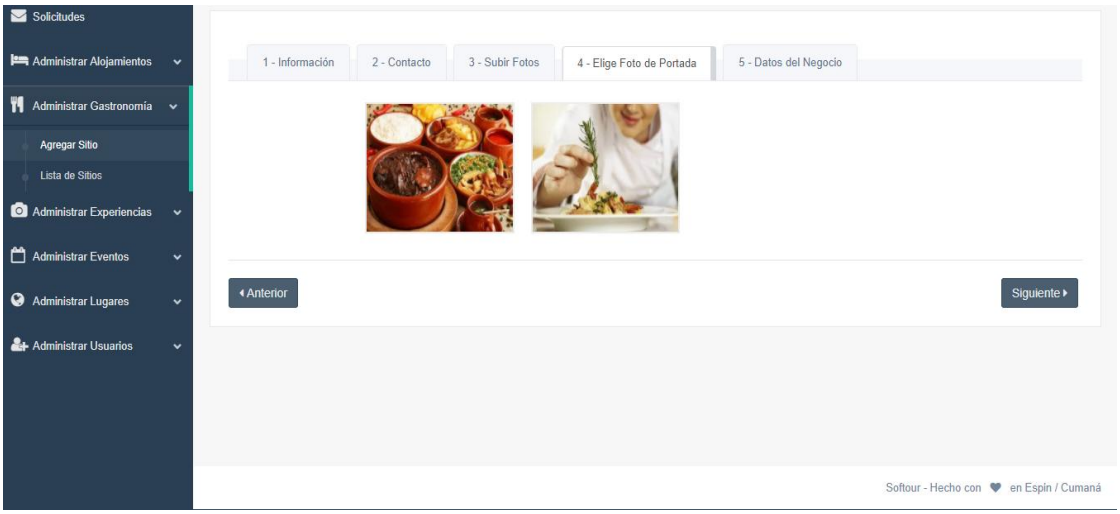


Figura X. 14.Interfaz 4 de proceso de Modificar Gastronomía por usuario Administrador

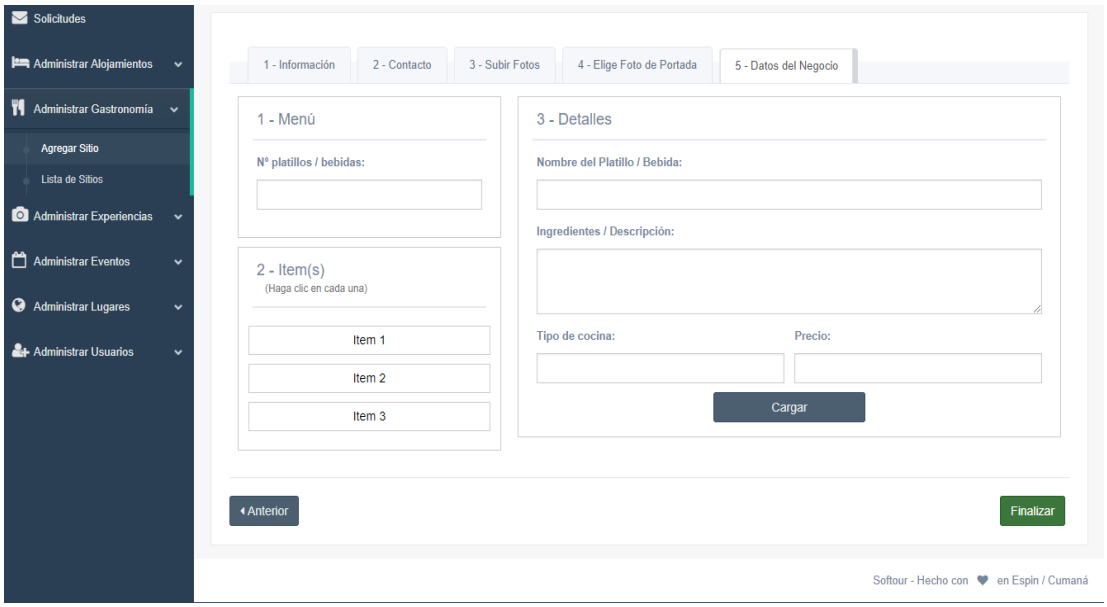


Figura X. 15. Interfaz 5 de proceso de Modificar Gastronomía por usuario Administrador

Parte del código desarrollado para la funcionalidad Modificar Gastronomía por usuario Administrador

```
modificar(req, res, next) {
    this._validar.campos(req.body, true, (error, resultados) => {
        if(error) return next(error);
        res.locals.datos = resultados;

        if(res.locals.datos.error){
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_modificar(req, res);
        }

        else {
            let tipo_entidad = req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.
                tipo_entidad.substring(1);

            this._Lugar.buscar({"url": req.params.url.toLowerCase(), "__t": tipo_entidad}, (error, buscado) => {
                if(error){
                    res.locals.datos = {"error": error};
                    res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
                    this._respuesta.para_modificar(req, res);
                }

                if(!buscado) return next(new this._error_NoEncontrado());
                res.locals.datos.value.geo_ubicacion = {
                    "type": "Point",
                    "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
                };
                res.locals.datos.value.ubicacion = {
                    "direccion": res.locals.datos.value.direccion,
                    "pais": res.locals.datos.value.pais,
                    "estado": res.locals.datos.value.estado,
                    "ciudad": res.locals.datos.value.ciudad,
                    "codigo_postal": res.locals.datos.value.codigo_postal
                };

                res.locals.datos.value.url = req.params.url.toLowerCase();
                let lugar = this._fabricar.lugarTipo(res.locals.datos.value.tipo_lugar);
                lugar.modificar(res.locals.datos.value, (error, modificado) => {
                    if(error) {
                        res.locals.datos.error = error;
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_modificar(req, res);
                    }
                    if(!modificado) return next(new this._error_NoEncontrado());

                    res.locals.datos = {"value": modificado};
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_modificar(req, res);
                });
            });
        }
    });
}
```

Figura X. 16. Código desarrollado para el proceso Modificar Gastronomía por usuario Administrador.

Apéndice Y. Ciclo de Versión Administrador: Iteración Listar Gastronomía.

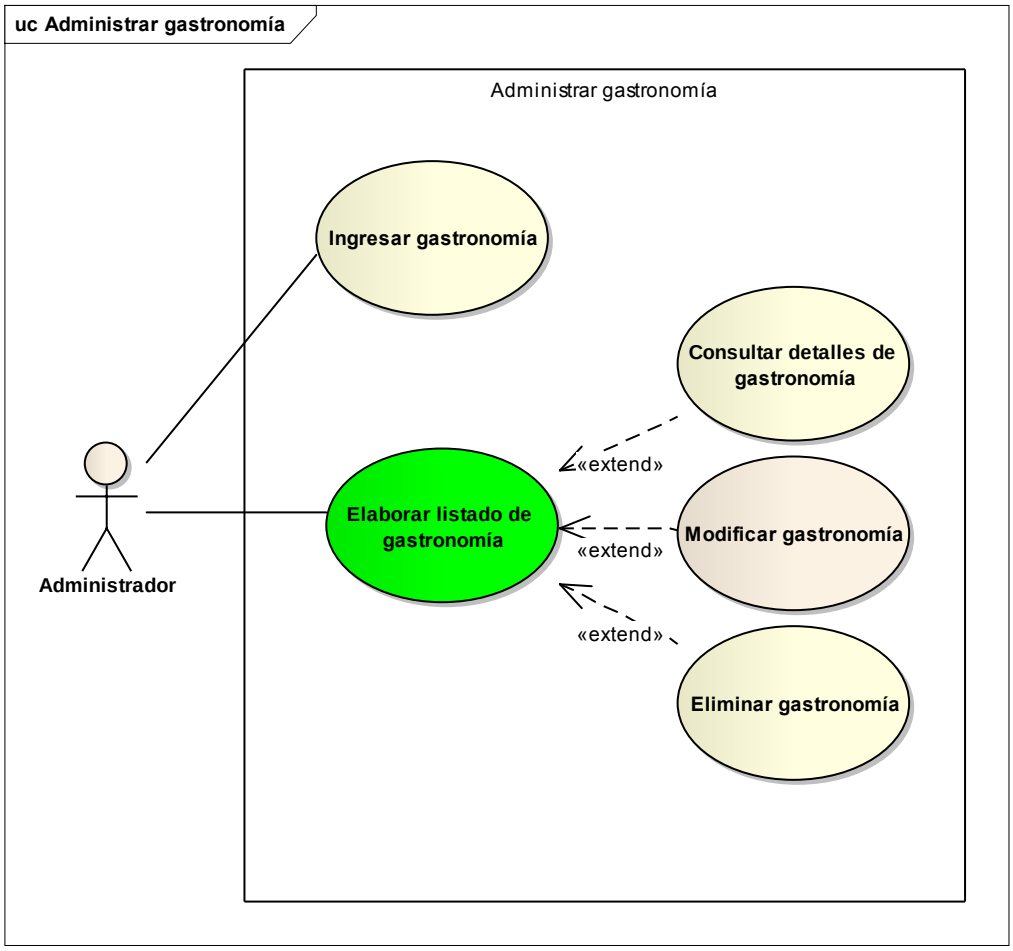


Figura Y. 1.Caso de uso: Elaborar Listado Gastronomía por administrador

Tabla Y. 1. Descripción Caso de uso Elaborar Listado Gastronomía por usuario administrador

Nombre del Caso de uso	Elaborar listado de gastronomías.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en elaborar un listado de las gastronomías registradas en la de base de datos del sistema, de manera satisfactoria.	
Precondición	Para elaborar el listado de las gastronomías, deben estar registradas en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Elaborar listado de gastronomías".
	2	El sistema muestra el listado de las gastronomías almacenados en la base de datos del sistema, de acuerdo a un orden predeterminado, brindando las siguientes opciones de extensión: a. "Consultar gastronomía". b. "Modificar gastronomía" c. "Eliminar gastronomía"
	3	El Administrador visualiza el listado de las gastronomías y selecciona alguna de las opciones de extensión propuestas.
	4	El sistema extiende <<extend>> hacia el caso de uso que representa la opción seleccionada.
	5	El Administrador selecciona la opción cerrar.
	6	El sistema finaliza el caso de uso.
Postcondición	Se elabora el listado de gastronomías existentes en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no seleccionar ninguna opción de extensión y selecciona cancelar el caso de uso, ya sea seleccionando el botón cerrar o el botón cancelar.
	4.1.	El sistema finaliza el caso de uso.
Reglas del negocio	1. El listado de gastronomías debe ser elaborado de acuerdo a un orden preestablecido. 2. Se podrá seleccionar, desde el listado general de gastronomías, una gastronomía en específica para realizar las operaciones de consultar, modificar o eliminar.	
Inclusiones	Ninguna.	
Extensiones	1. "Consultar gastronomía". 2. "Modificar gastronomía". 3. "Eliminar gastronomía".	

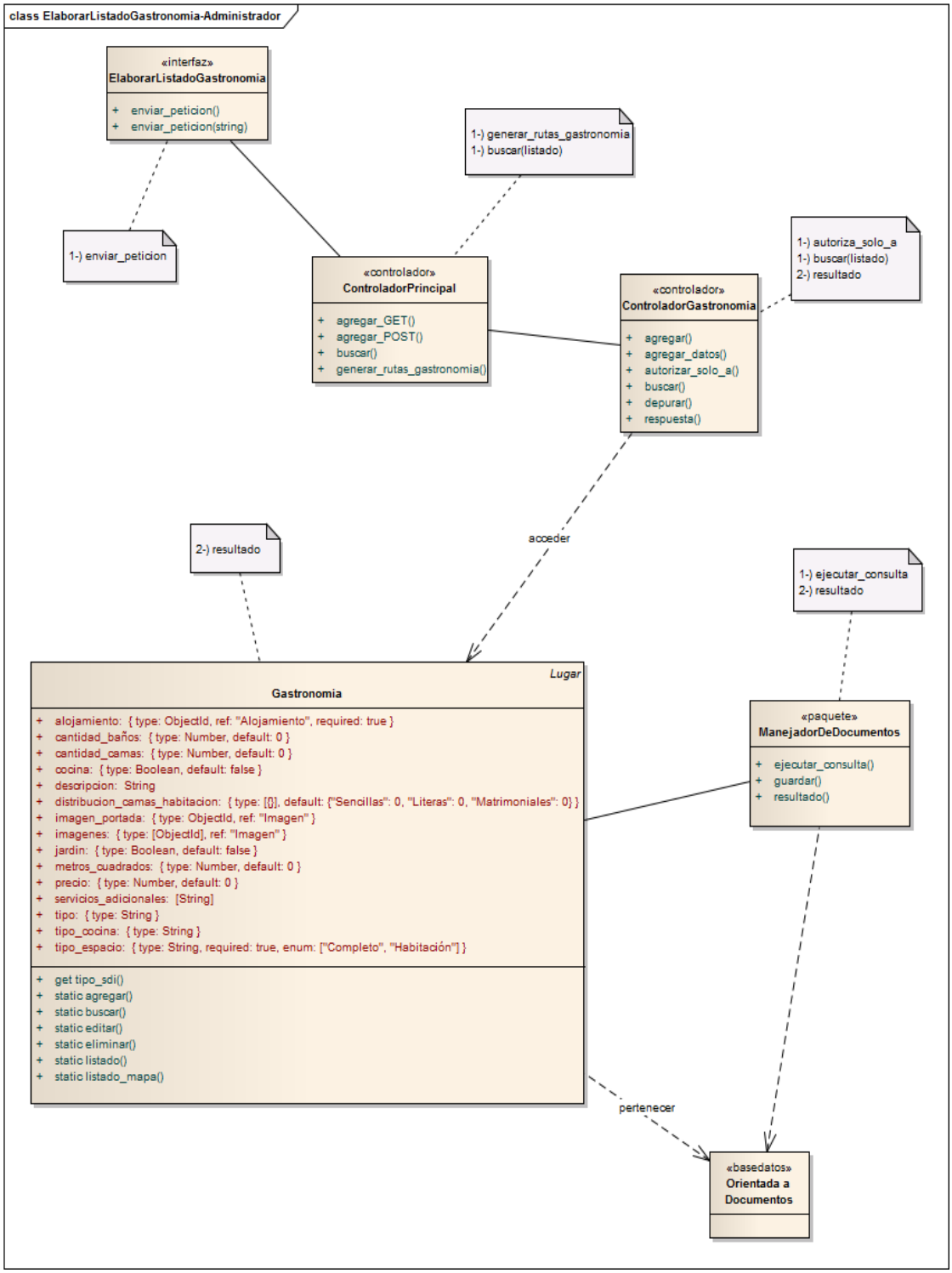


Figura Y. 2.Diagrama de Clases: Elaborar Listado Gastronomía por usuario Administrador

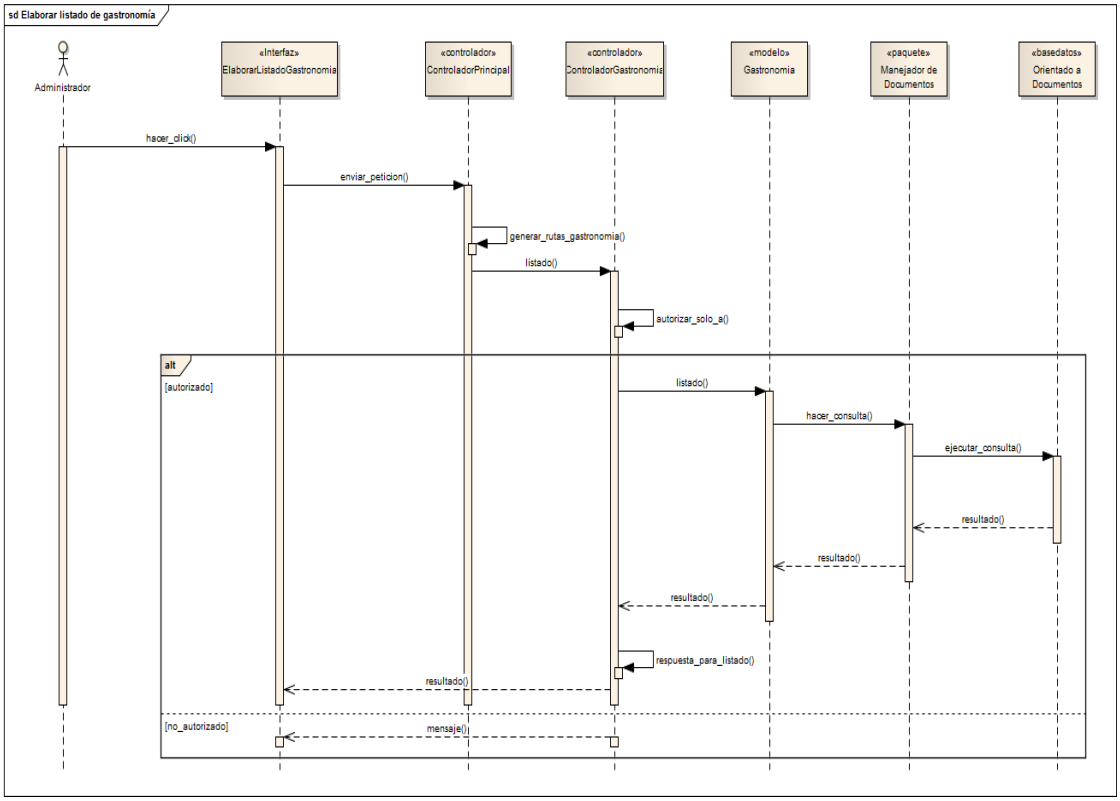


Figura Y. 3.Diagrama de secuencia: Elaborar Listado Gastronomía por usuario Administrador

Figura Y. 4. Colección Gastronomía 1: Listar Gastronomía por usuario Administrador

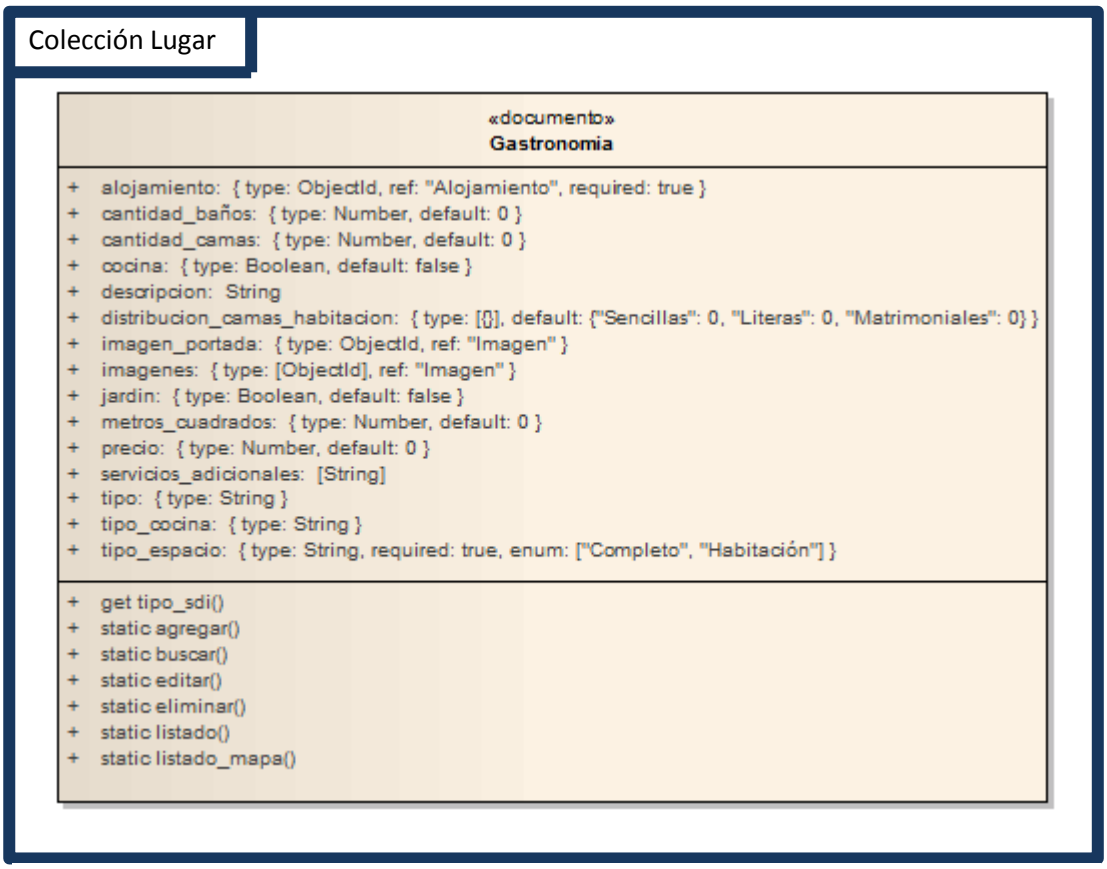


Figura Y. 5. Colección Gastronomía 2: Listar Gastronomía por usuario Administrador

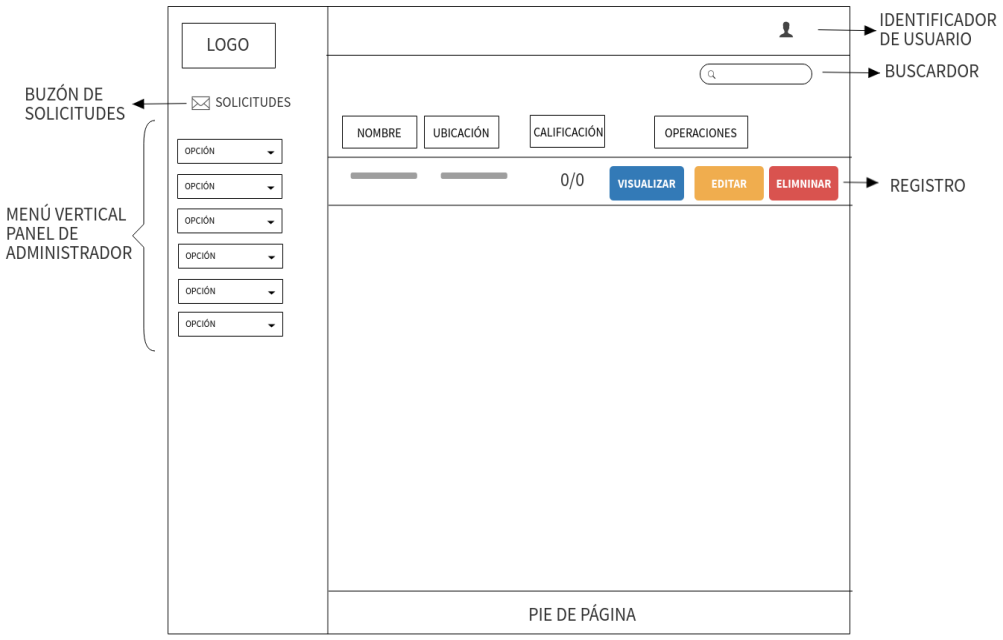


Figura Y. 6. Plantilla de interfaz 1: Listar Gastronomía por usuario administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Lista de Gastronomía y el sistema responde mostrando la interfaz correspondiente.

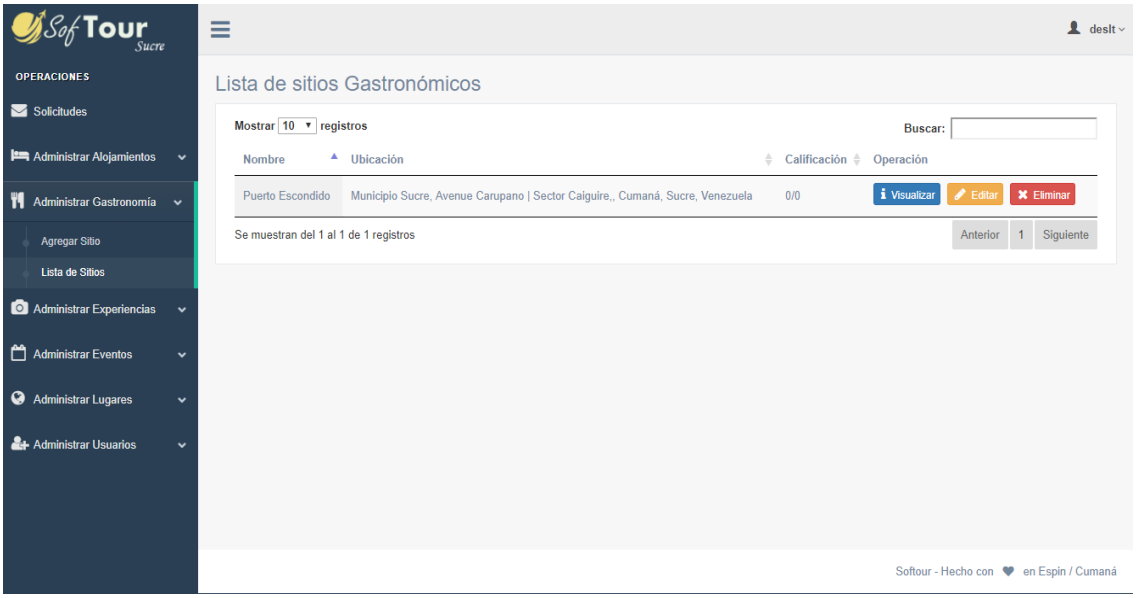


Figura Y. 7. Interfaz Listado de Gastronomía por usuario Administrador

Parte del código desarrollado para la funcionalidad Listado de Gastronomía por usuario Administrador

```
listado(req, res, next) {
    let query = {};

    if(req.session.passport.user.tipo_usr == "PrestadorServicio" && req.params.tipo_entidad != "lugar")
        query.prestador_servicio = req.session.passport.user.id;

    if(req.params.tipo_entidad == "lugar")
        query.__t = null;
    else{
        query.__t = req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1);
        query.aprobado = true;
    }

    this._Lugar.listado({"consulta": query}, (error, lista) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_listado(req, res);
        }

        res.locals.datos = {"value": lista};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_listado(req, res);
    })
}
```

Figura Y. 8. Código desarrollado para el proceso Listar Gastronomía por usuario Administrador.

Apéndice Z. Ciclo de Versión Administrador: Iteración Visualizar Gastronomía.

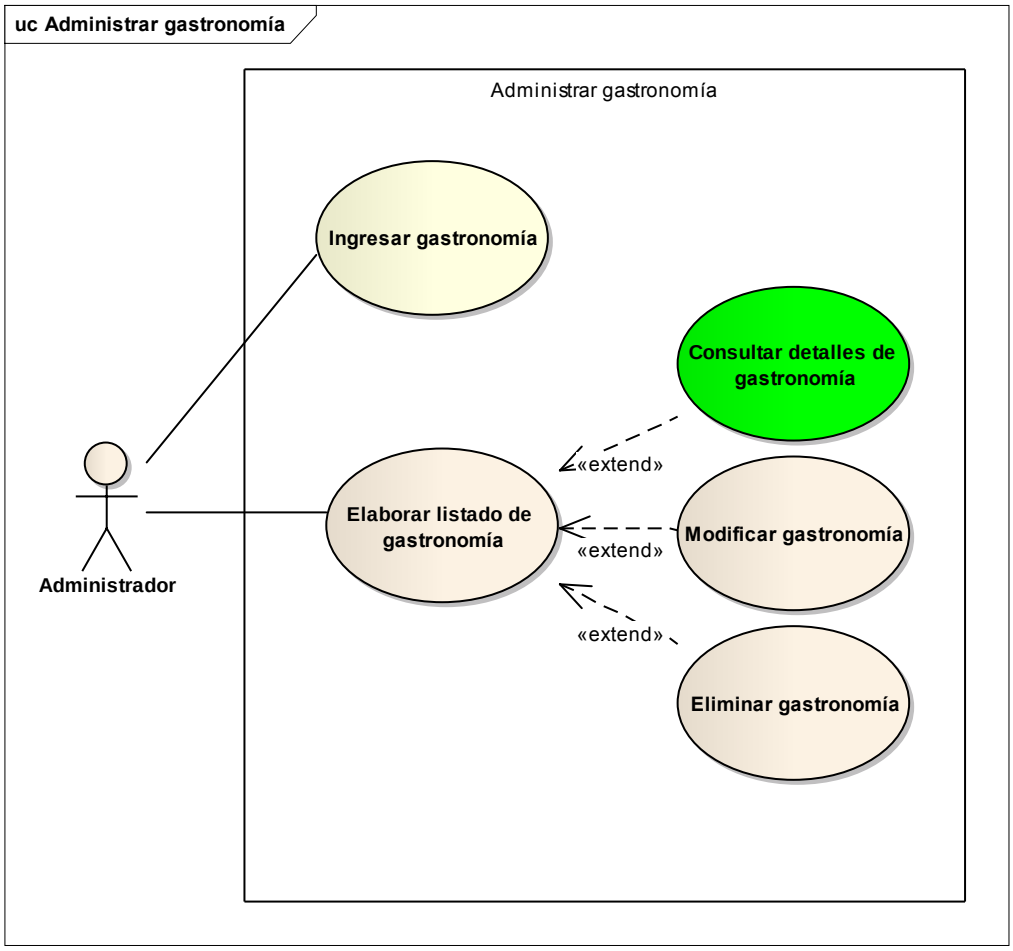


Figura Z. 1.Caso de uso: Consultar Detalle de Gastronomía por administrador

Tabla Z. 1.Descripción Caso de uso Consultar Detalle de Gastronomía por usuario administrador

Nombre del Caso de uso	Consultar gastronomía.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Casos de Uso	El caso de uso consiste en consultar los atributos de la gastronomía de manera satisfactoria.	
Precondición	Para consultar una gastronomía, debe estar registrada en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Consultar gastronomía”.
	2	El sistema ejecuta la búsqueda de la gastronomía, mostrando un formulario con la consulta.
	3	El Administrador visualiza la consulta de los atributos de la gastronomía, luego selecciona aceptar.
	4	El sistema finaliza el caso de uso.
Postcondición	Se consultan los atributos de la gastronomía de la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
Reglas del negocio	1. La gastronomía debe estar registrada para que pueda ser consultada.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

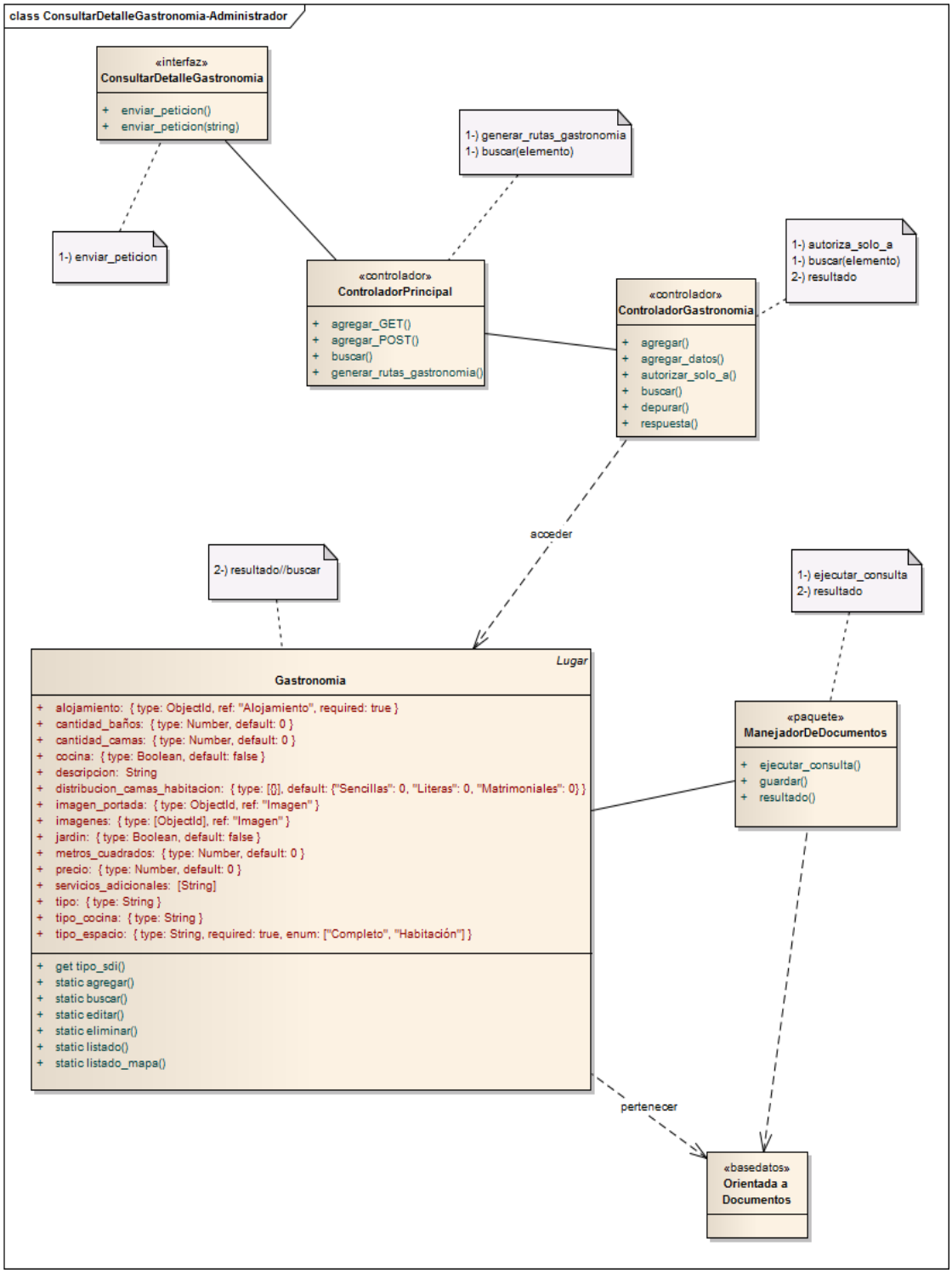


Figura Z. 2.Diagrama de Clases: Consultar Detalle Gastronomía por usuario Administrador

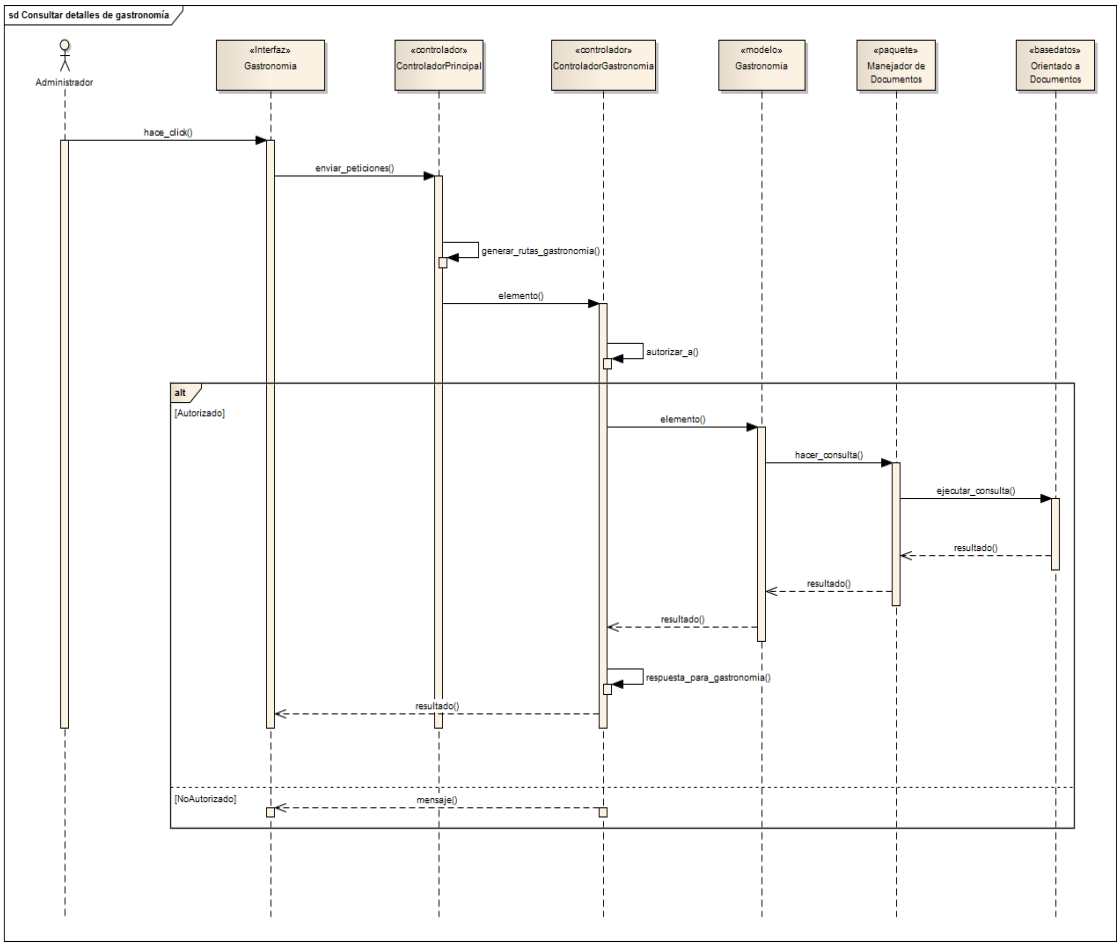


Figura Z. 3. Diagrama de secuencia: Consultar Detalle Gastronomía por usuario Administrador

Figura Z. 4. Colección Gastronomía: Consultar Gastronomía por usuario Administrador.

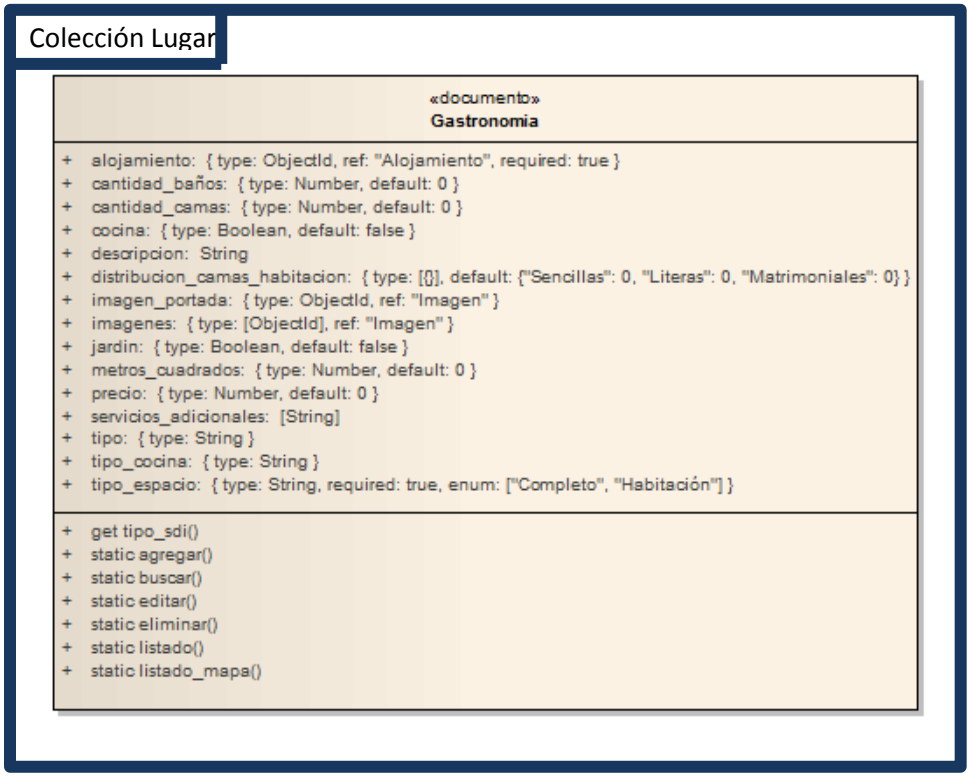


Figura Z. 5. Colección Gastronomía: Consultar Gastronomía por usuario Administrador.

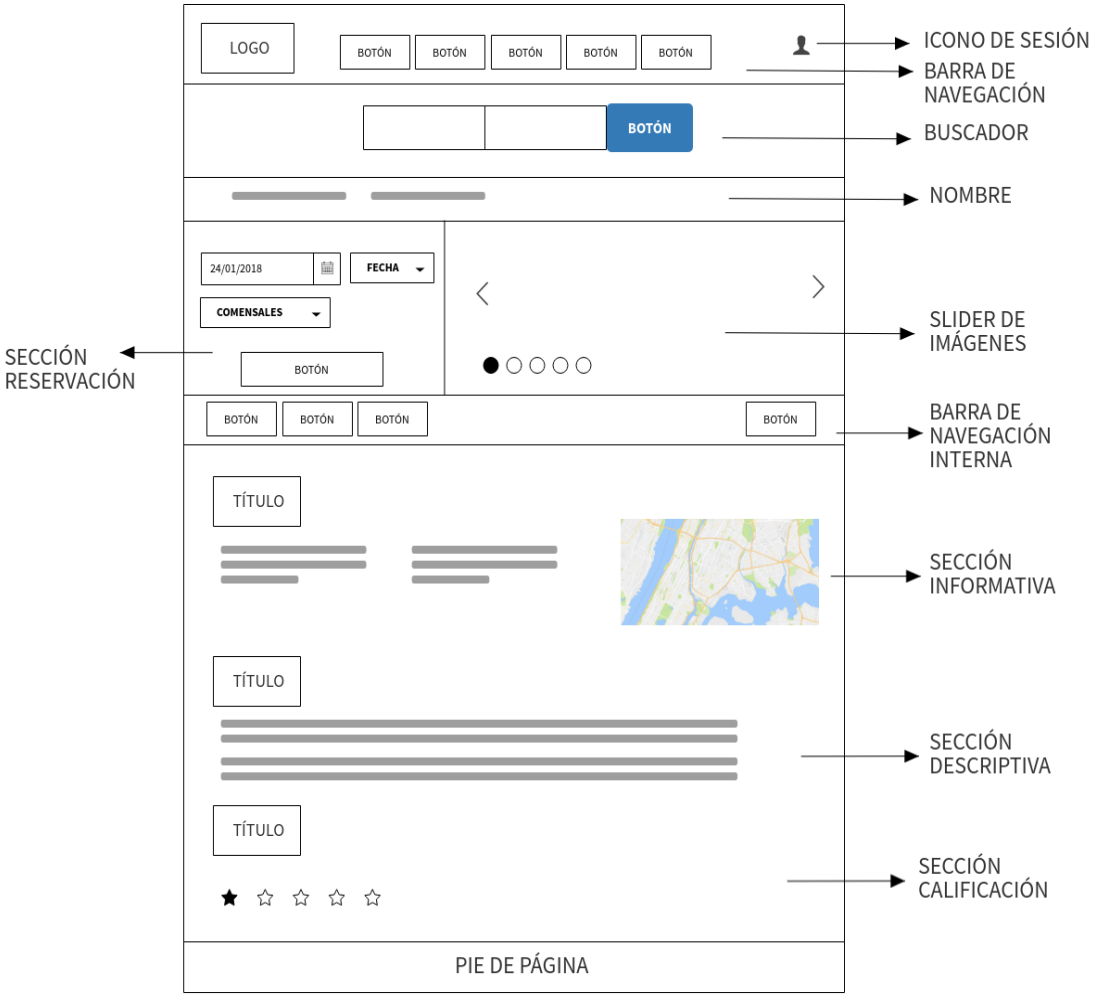


Figura Z. 6. Plantilla de interfaz: Consultar Gastronomía por usuario administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Consultar Gastronomía y el sistema responde mostrando la interfaz con las especificaciones

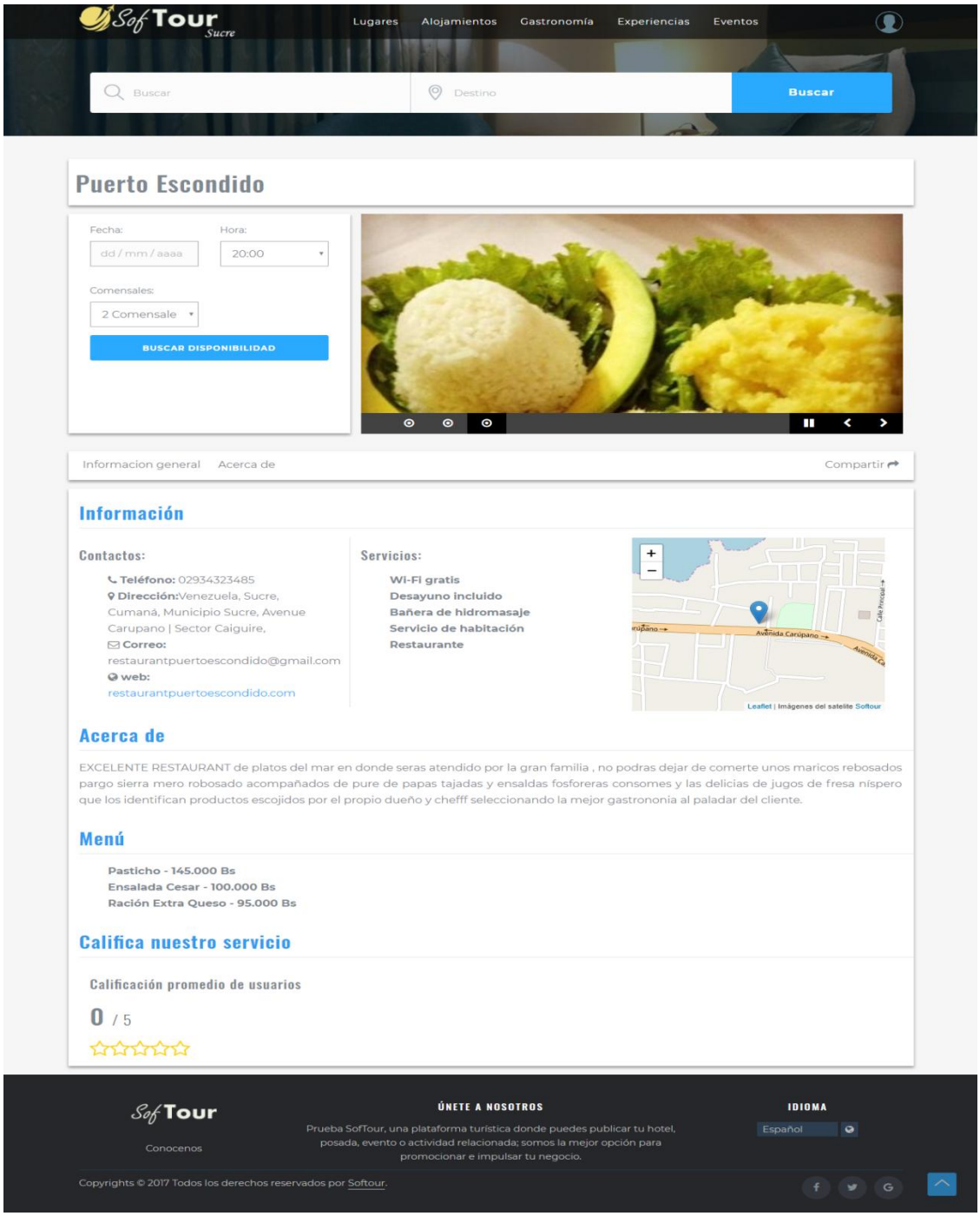


Figura Z. 7.Interfaz de proceso de Consultar Detalle Gastronomía por usuario Administrador

Parte del código desarrollado para la funcionalidad Consultar Detalle Gastronomía por usuario Administrador

```
consultar(req, res, next) {
    let parametros = {
        "__t": req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1),
        "url": req.params.url.toLowerCase()
    };
    if(parametros.__t)
        parametros.aprobado = true;

    this._Lugar.buscar_consulta(parametros, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        if(buscado.__t){
            this.CategoriaAlojamiento.find({"alojamiento": buscado._id}, (error, lista_categorias) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    if (req.isAuthenticated())
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_consultar_producto(req, res);
                }

                res.locals.datos = {
                    "value": buscado,
                    "lista_categorias": lista_categorias
                };

                if (req.isAuthenticated())
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                this._respuesta.para_consultar(req, res);
            });
        }
        else {
            res.locals.datos = { "value": buscado };

            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
    })
}
```

Figura Z. 8. Código desarrollado para el proceso Consultar Gastronomía por usuario Administrador.

Apéndice AA. Ciclo de Versión Administrador: Iteración Eliminar Gastronomía

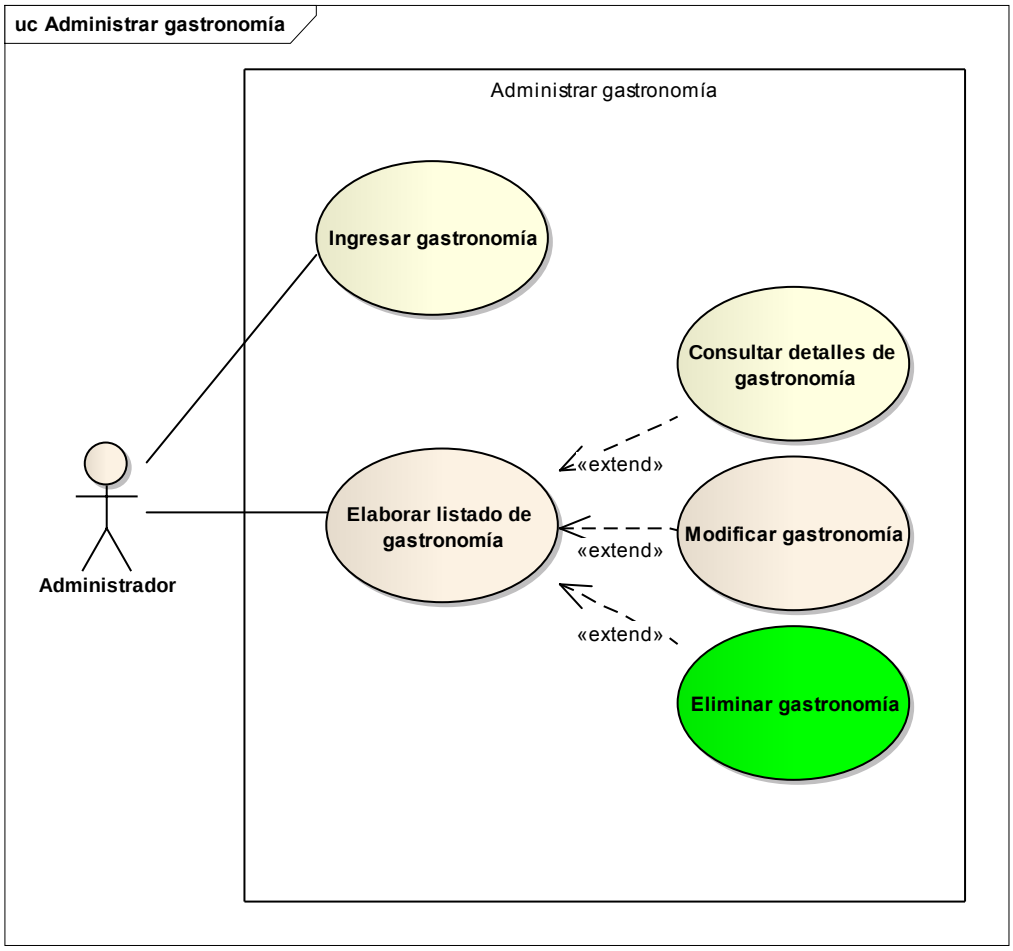


Figura AA. 1. Caso de uso: Eliminar Gastronomía por administrador

Tabla AA. 1. Descripción Caso de uso Eliminar Gastronomía por usuario administrador

Nombre del Caso de uso	Eliminar gastronomía.	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en eliminar una gastronomía de manera satisfactoria.	
Precondición	Para eliminar una gastronomía, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Eliminar gastronomía”.
	2	El sistema ejecuta la búsqueda de la gastronomía, mostrando un formulario con la consulta de la gastronomía a eliminar.
	3	El Administrador visualiza la gastronomía que desea eliminar y selecciona aceptar.
	4	El sistema realiza la eliminación de la gastronomía en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se elimina la gastronomía de la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no continuar con el casos de uso y selecciona cancelar el caso de uso, ya sea seleccionando el botón cerrar o el botón cancelar.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. La gastronomía debe estar registrado para que pueda ser eliminado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

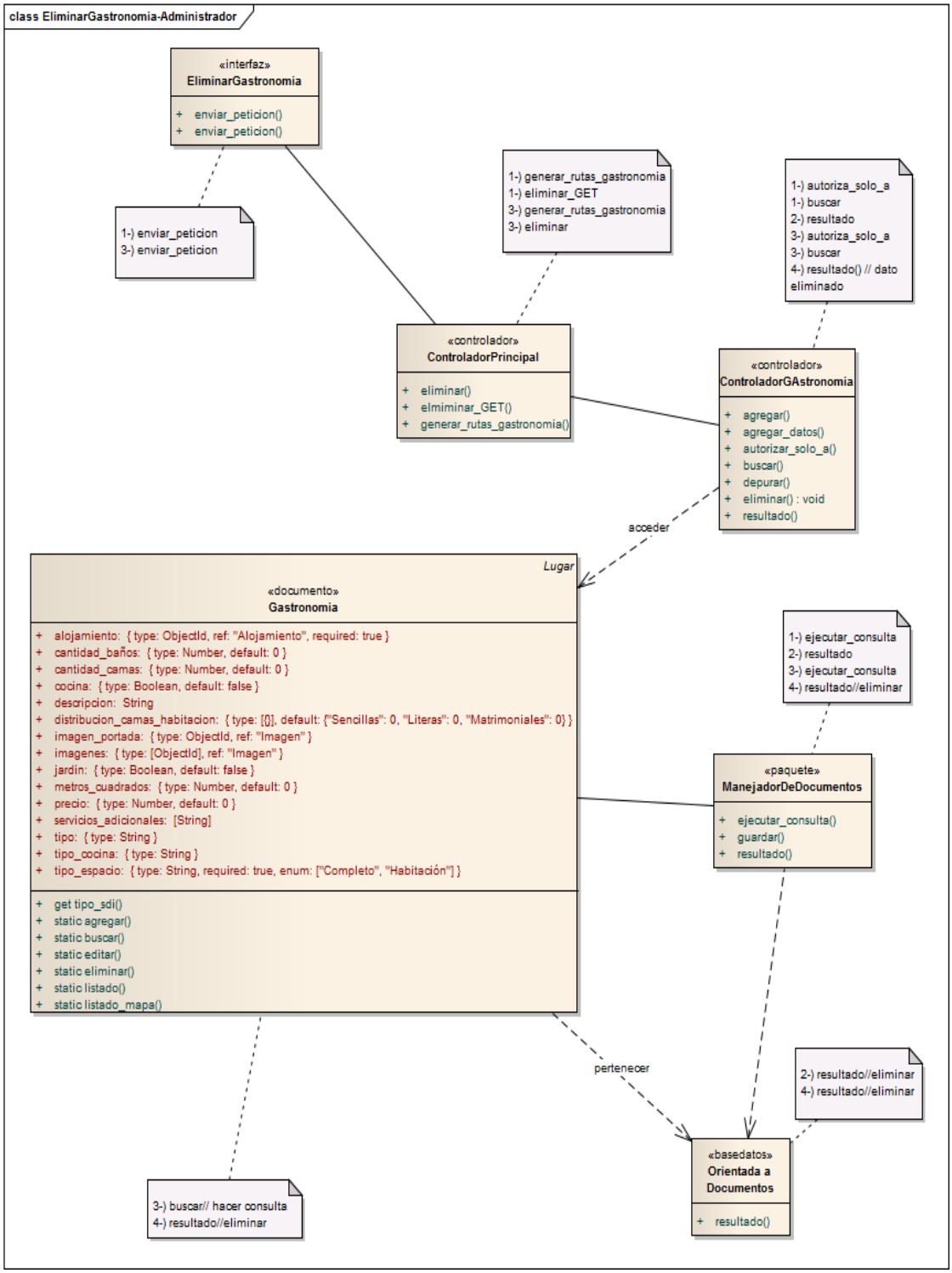


Figura AA. 2. Diagrama de Clases: Eliminar Gastronomía por usuario Administrador

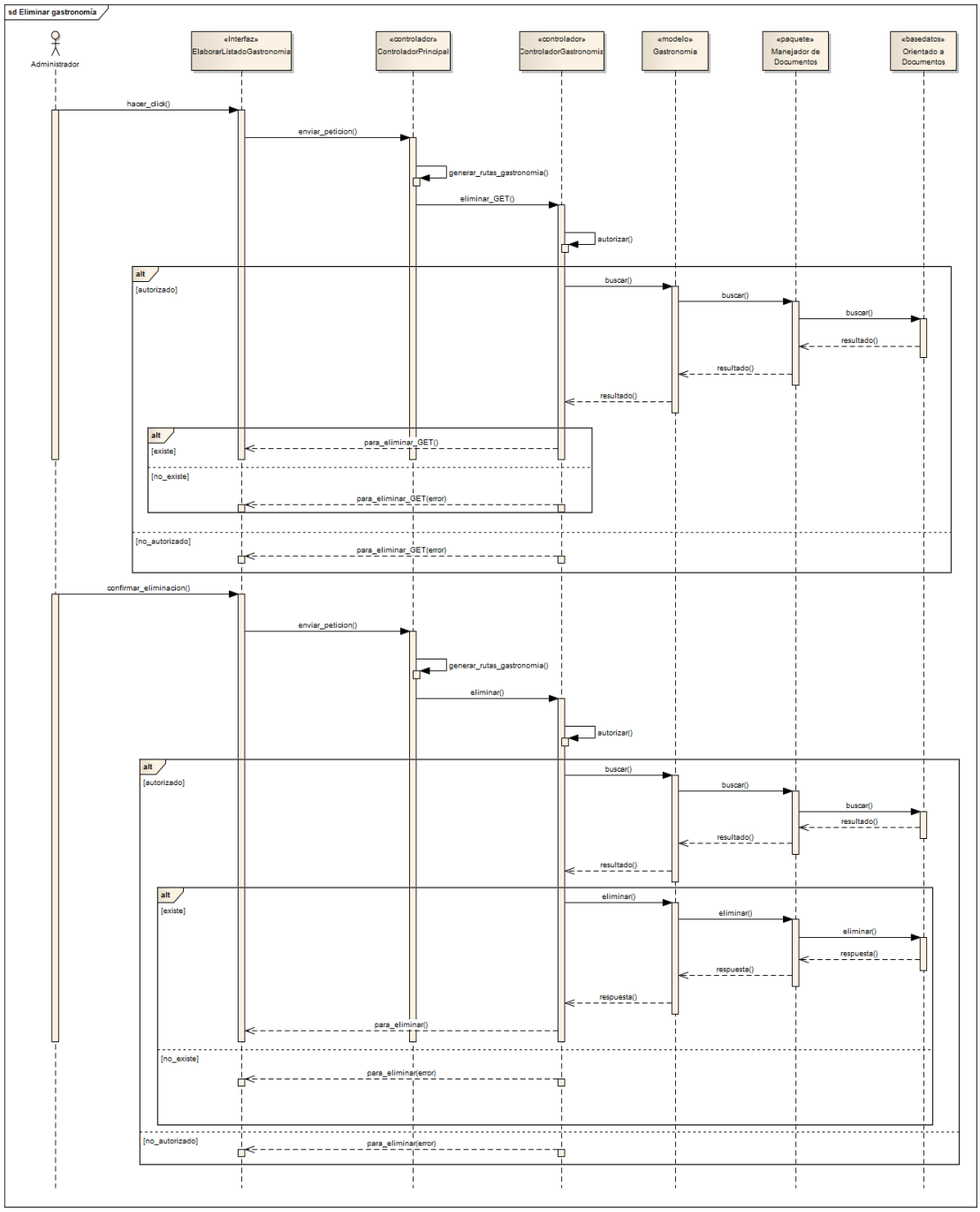


Figura AA. 3. Diagrama de secuencia: Eliminar gastronomía por usuario Administrador

Figura AA. 4. Colección Gastronomía 1: Eliminar Gastronomía por usuario Administrador.

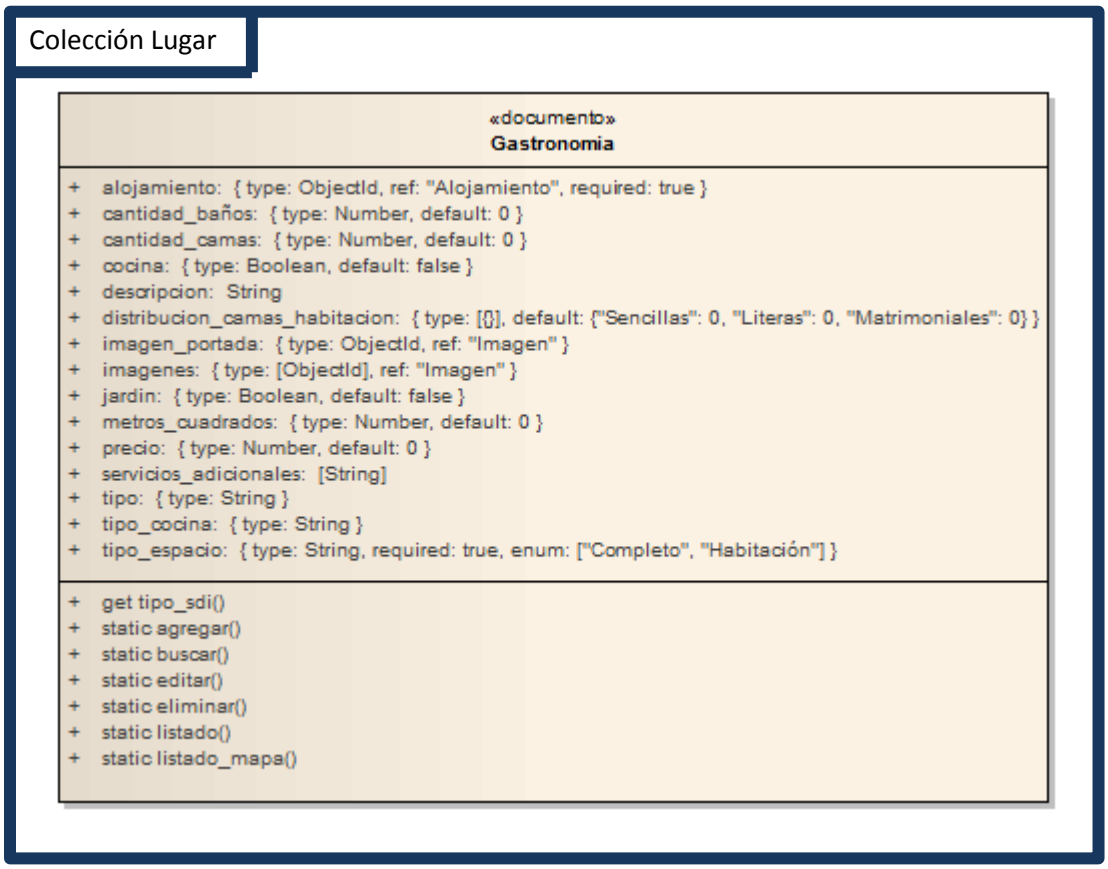


Figura AA. 5. Colección Gastronomía 2: Eliminar Gastronomía por usuario Administrador.

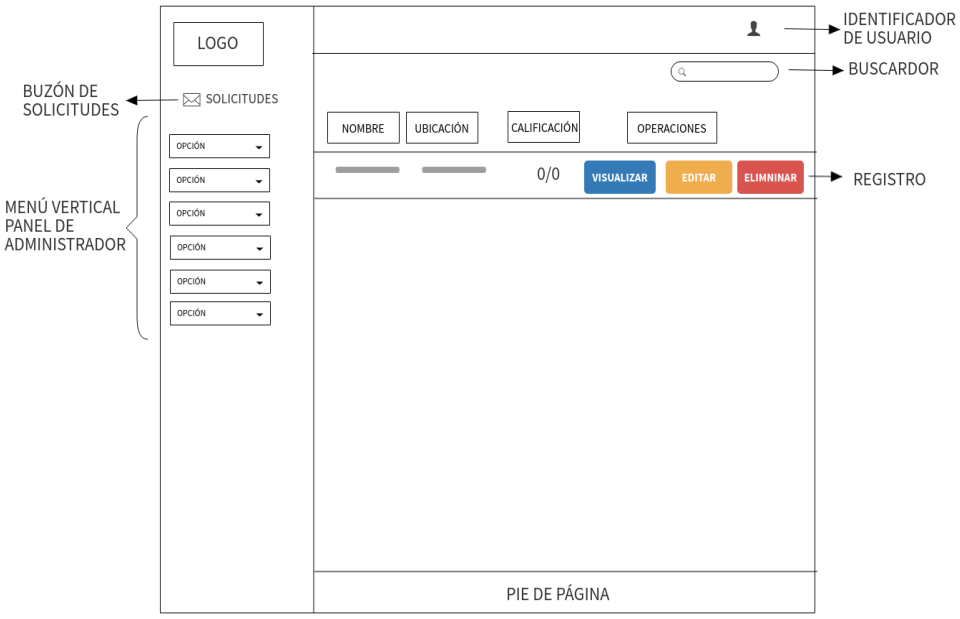


Figura AA. 6. Plantilla de interfaz 1: Eliminar Gastronomía por usuario administrador.

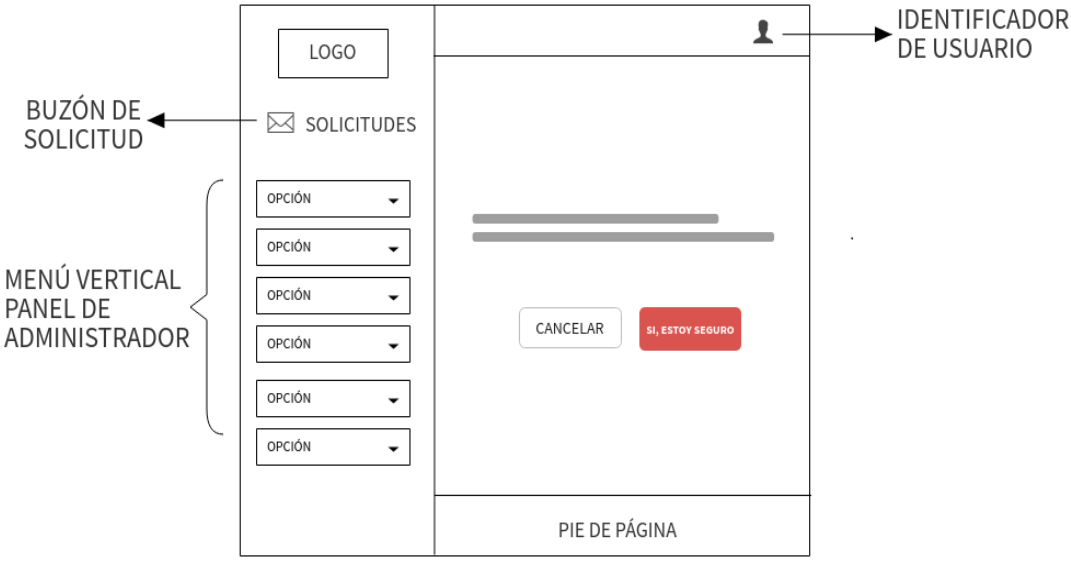


Figura AA. 7. Plantilla de interfaz 1: Eliminar Gastronomía por usuario administrador.

Flujo a seguir para la funcionalidad.

El Administrador selecciona la opción Eliminar Gastronomía y el sistema responde mostrando la interfaz de confirmación de acción.

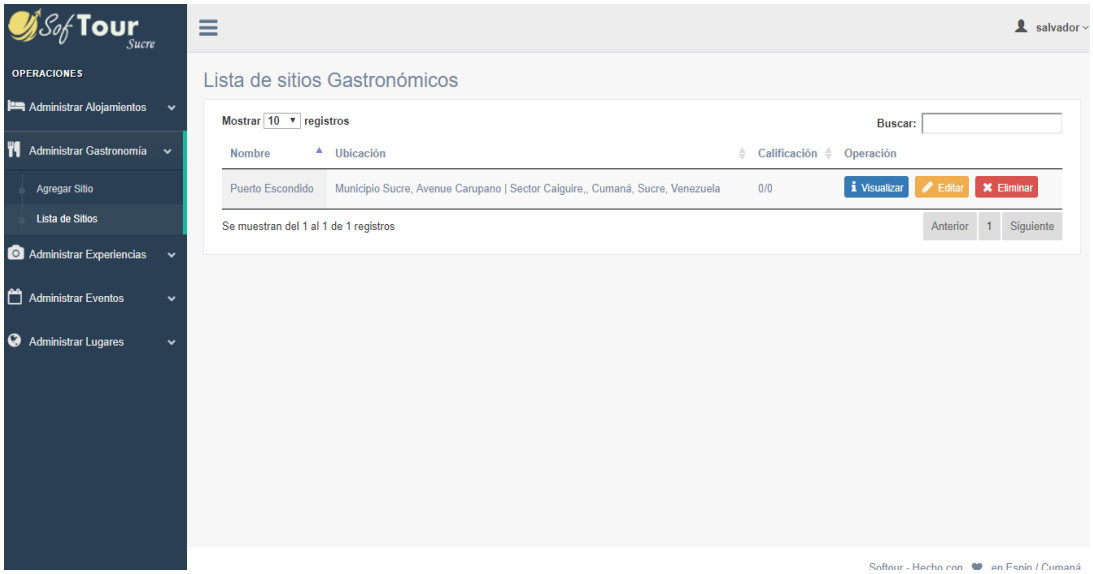


Figura AA. 8. Interfaz 1 de proceso de Eliminar Gastronomía por usuario Administrador

Figura AA. 9. Interfaz 2 de proceso de Eliminar Gastronomía por usuario Administrador

Parte del código desarrollado para la funcionalidad Eliminar Gastronomía por usuario Administrador

```
eliminar(req, res, next) {
    let tipo_entidad = req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.
        substring(1);

    this._Lugar.eliminar({"url": req.params.url.toLowerCase(), "__t": tipo_entidad}, (error, eliminado) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_eliminar(req, res);
        }
        if(!eliminado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": {"url": "listado"}};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_eliminar(req, res);
    });
}
```

Figura AA. 10. Código desarrollado para el proceso Eliminar Gastronomía por usuario Administrador.

Apéndice BB. Ciclo de Versión Administrador: Iteración Ingresar Prestador de Servicios.

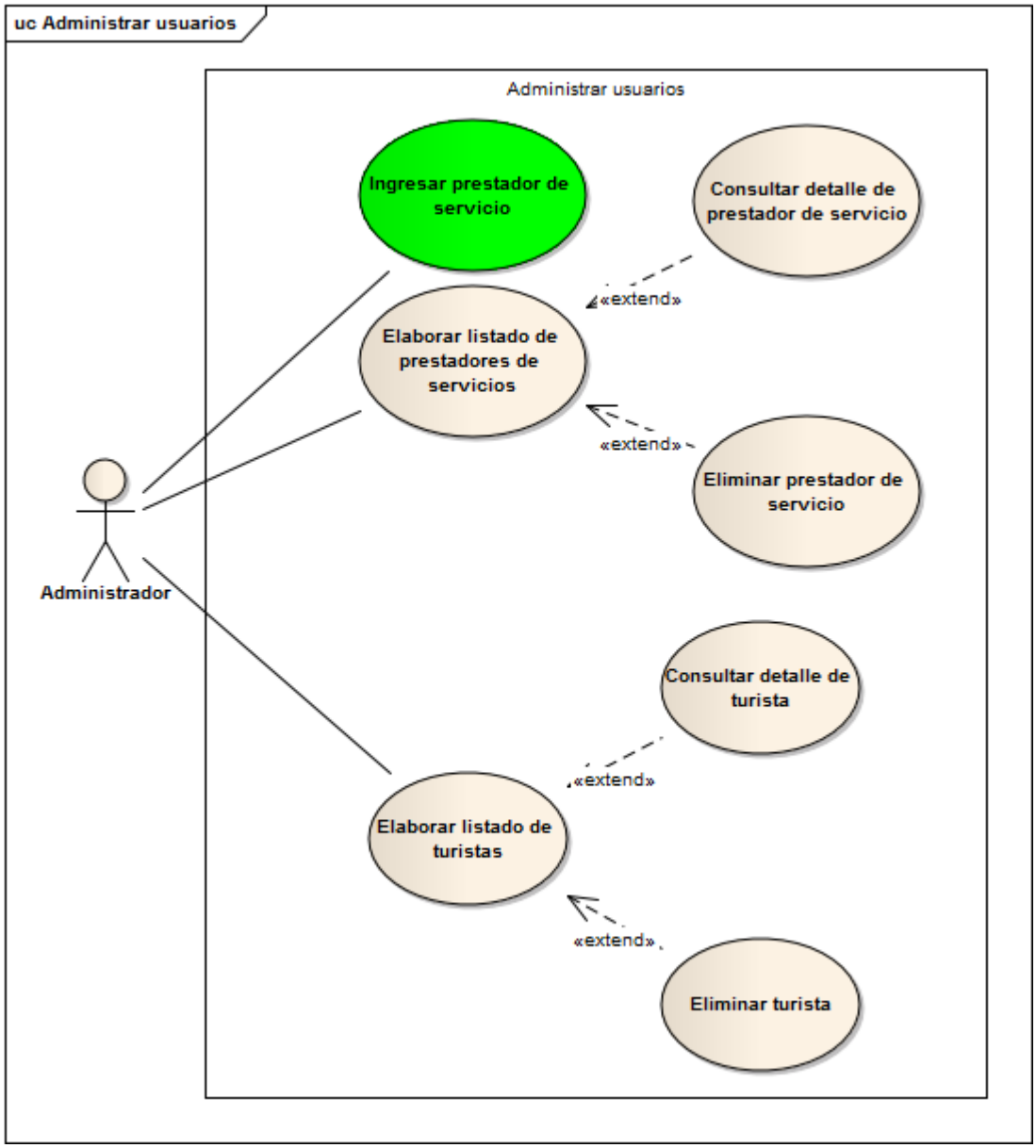


Figura BB. 1.Caso de uso: Ingresar Prestador de Servicios por usuario administrador

Tabla BB. 1. Descripción Caso de uso Ingresar Ingresar Prestador de Servicios por usuario administrador

Nombre del Caso de uso	Registrar Prestador De Servicio	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en registrar un prestador de servicios de manera satisfactoria.	
Precondición	Para registrar un prestador de servicios el mismo no debe encontrarse en la BD del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Agregar Prestador de Servicios”.
	2	El sistema muestra el formulario correspondiente para ingresar los atributos del prestador de servicios que se desea registrar.
	3	El Administrador ingresa los atributos del prestador de servicios que se desea registrar.
	4	El sistema registra el prestador de servicios en la BD y finaliza el caso de uso.
Postcondición	Se registró el prestador de servicios de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Administrador decide no continuar con el caso de uso y selecciona otra opción a desarrollar.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. El prestador de servicio no debe estar registrado para que pueda ser ingresado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

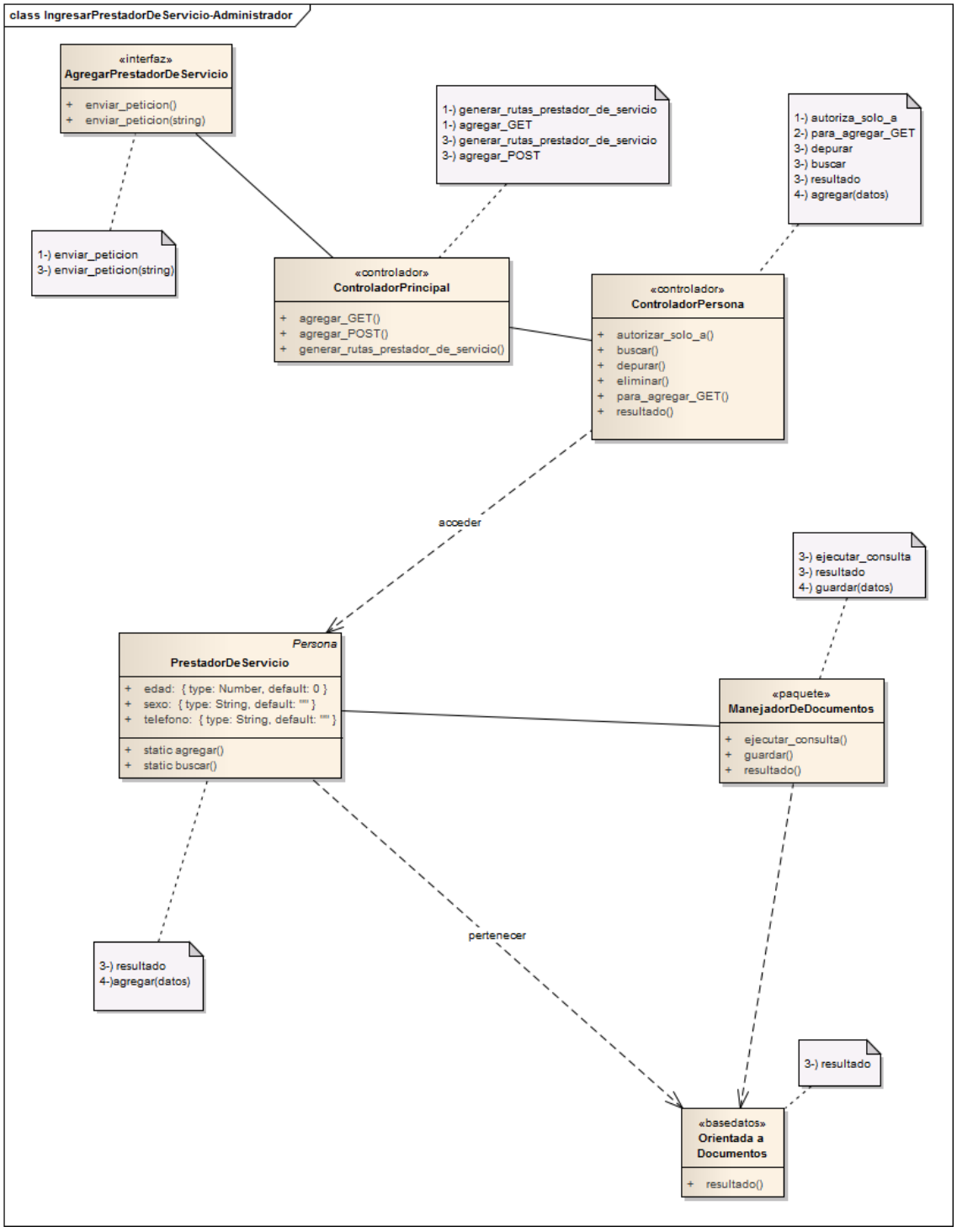


Figura BB. 2.Diagrama de Clases: Ingresar Prestador de Servicios por usuario administrador

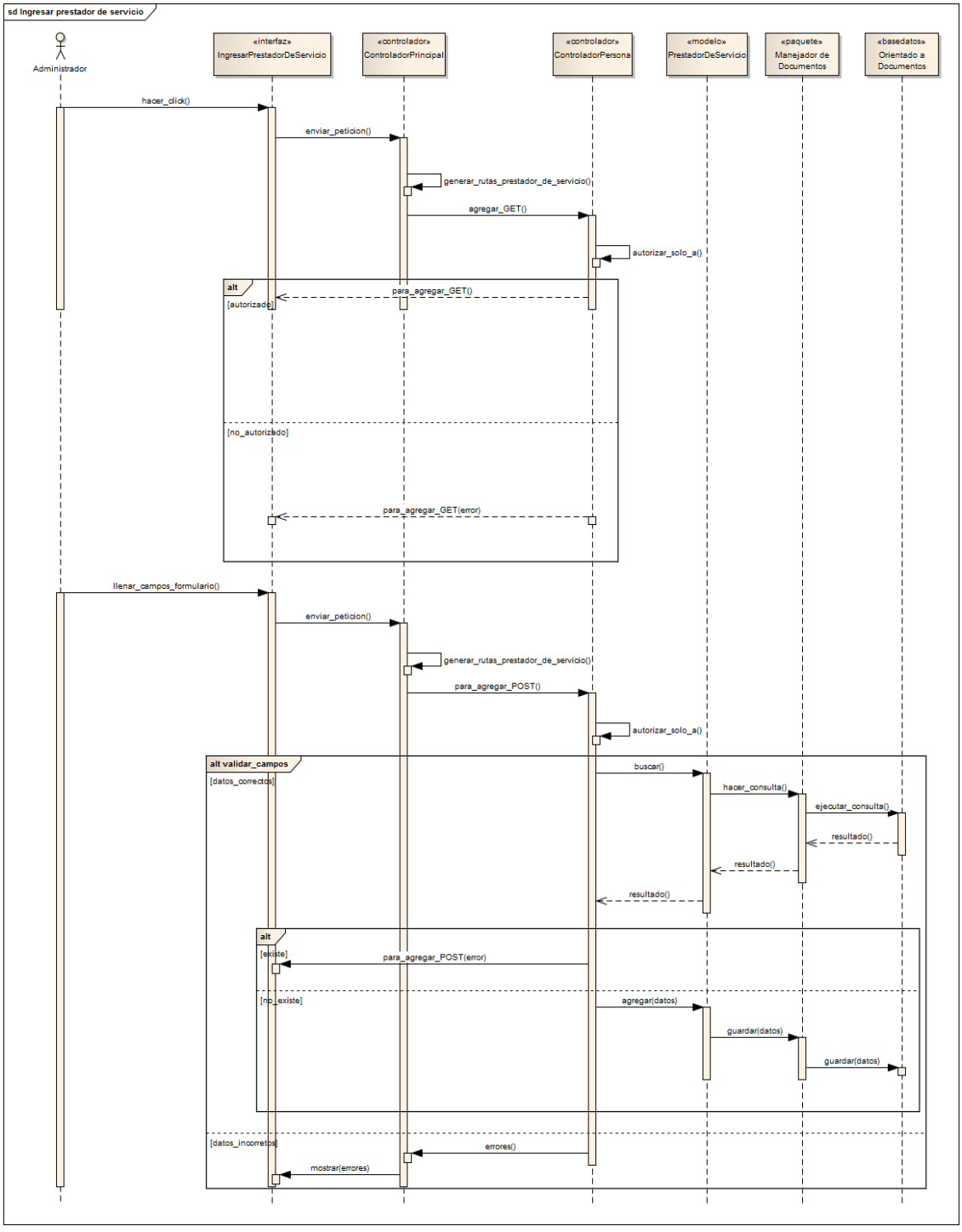


Figura BB. 3. Diagrama de secuencia: Ingresar Prestador de Servicios por usuario administrador

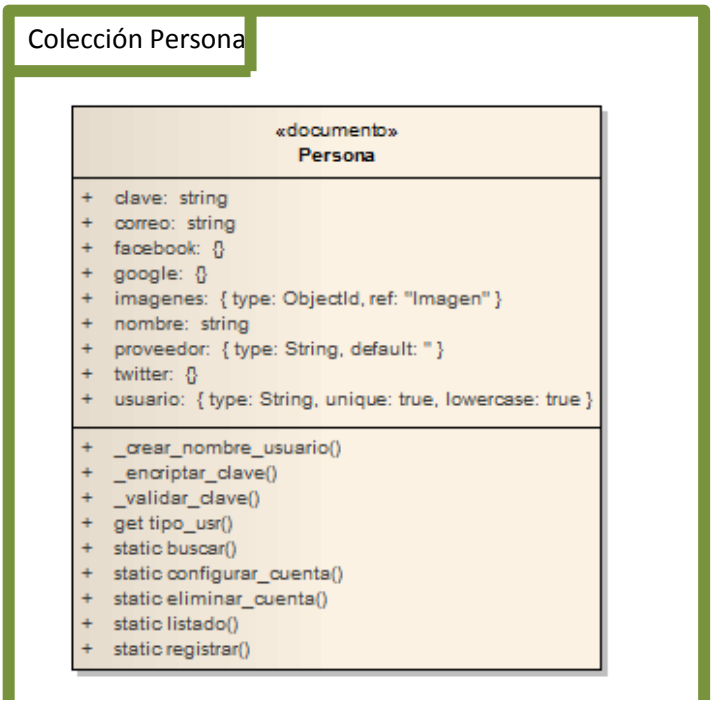


Figura BB. 4. Colección Persona 1: Ingresar Prestador de Servicio por usuario Administrador.

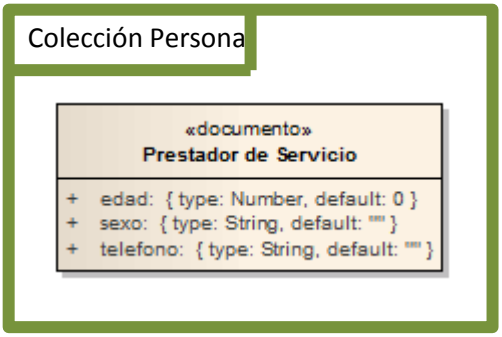


Figura BB. 5. Colección Persona 2: Ingresar Prestador de Servicio por usuario Administrador.

A continuación, se muestran el modelo de plantilla para el proceso de ingresar prestador de servicio

Figura BB. 6.Plantilla de interfaz: Ingresar Prestador de Servicio por usuario administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Agregar prestador de servicio y el sistema responde mostrando el formulario correspondiente.

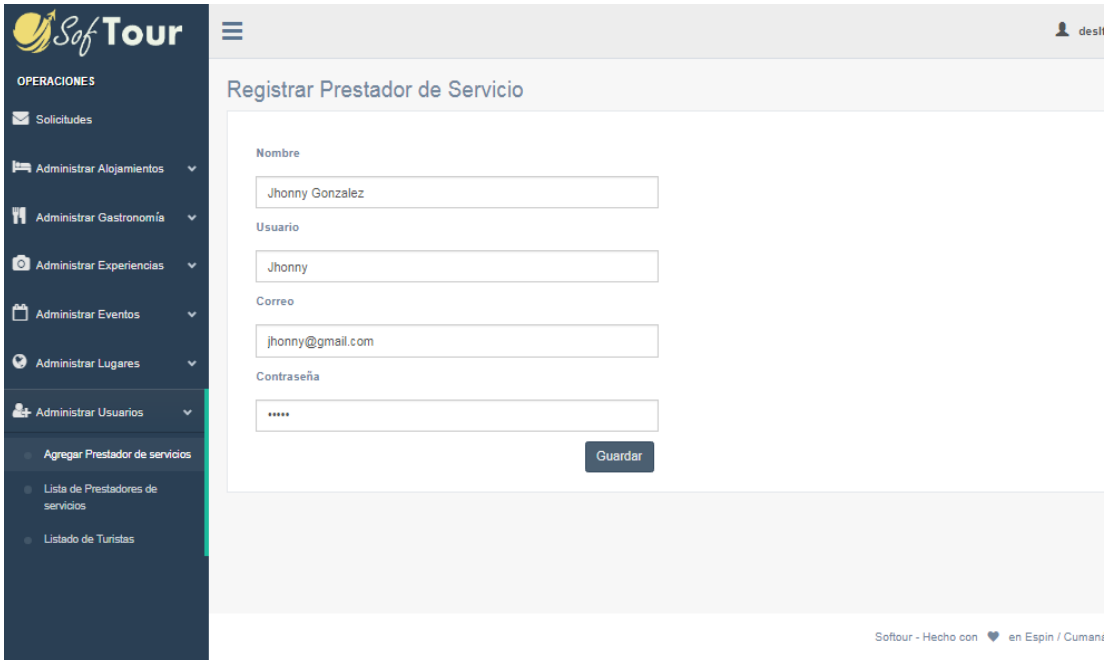


Figura BB. 7.Interfaz 1 de proceso de Agregar Prestador de servicio por usuario Administrador

Código desarrollado para la funcionalidad Agregar Prestador de servicio por usuario Administrador

```
ingresar_POST(req, res, next) {

    res.locals.datos = this._validar.campos(req.body, req.body.tipo_usr);

    if(res.locals.datos.error)
        this._respuesta.para_ingresar_POST(req, res);

    else
        // si SESSION es true, entonces q genere la sesion si es false entonces significa que el admin intenta registrar a un usuario
        Prestador Servicio y no se requiere sesionarlo
        return this._passport.authenticate('registrar-usuario', {
                failureFlash: true,
                session: res.locals.datos.value.tipo_usr == "Turista" ? true : false
            }, (error, usuario, datos_error, next) => {
            if(error) return next(error);

            if(datos_error)
                res.locals.datos.error = datos_error;
            else
                res.locals.datos = {"value": usuario};

            this._respuesta.para_ingresar_POST(req, res);
        }) (req, res, next);
}
```

Figura BB. 8. Código desarrollado para el proceso Ingresar Prestador de Servicio por usuario Administrador.

Prueba agregar prestador de servicio

El sistema contempla el registro de prestadores de servicios que son encargados de los diferentes eventos, actividades o cualquier negocio que se publicite en el sistema, el usuario Administrador cuenta con la posibilidad de agregar a dichos prestadores de servicios.

La siguiente prueba validara que esta función se dé correctamente, para eso se le aplico la técnica de caja negra rigiéndose bajo los siguientes parámetros:

Tabla BB. 2. Clases de equivalencia Agregar Prestador de servicio por Administrador

Condición de entrada	Tipo	Clases de equivalencias válidas	Clases de equivalencias no válidas
Nombre	Alfabético	1: Cadena de caracteres	2: Campo en blanco 3: Números
Usuario	Alfanumérico	1: Cadena de caracteres 3: Números	2: Campo en blanco
Correo	correo	4: símbolos, caracteres, números	5: Correo repetido
Contraseña	Valor	1: Cadena de caracteres 3: Números	2: Campo en blanco

Tabla BB. 3. Casos de prueba Agregar Prestador de servicio por Administrador

Condición de entrada	A1	A2	A3
Nombre	Carlos Ruiz	Jaime Lobos	Luis Que2
Usuario	Car22	Lobos	
Correo	Ruiz_21@gmail.com	lobos@hotmail.com	Zerpagmail.com
Contraseña	123cr	0000	-555
Clases Equivalencia	1,3	1,3	1,2,3
Resultado	Registrado	Registrado	No registrado

Se procede a llenar el formulario de acuerdo a lo planteado para el caso de prueba A1 como se muestra en la siguiente imagen.

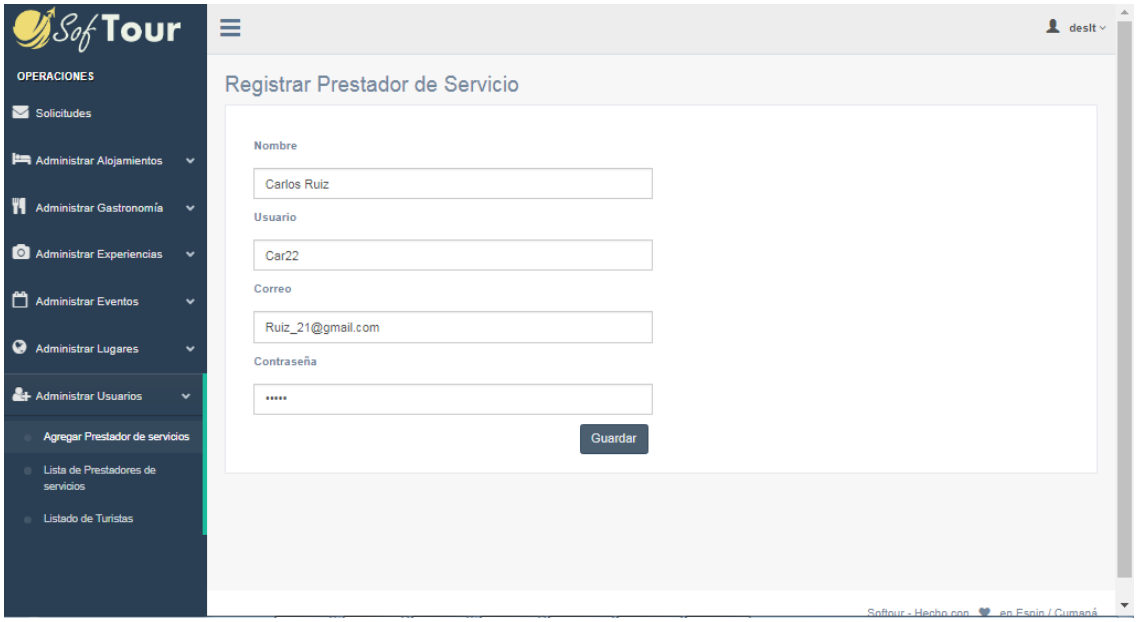


Figura BB. 9. caso de prueba A1 Agregar Prestador de Servicio por Administrador

Para el caso de prueba A1 no se encontraron fallos ni inconvenientes, de tal manera que el registro básico se cumplió de forma exitosa como se evidencia en la siguiente figura.

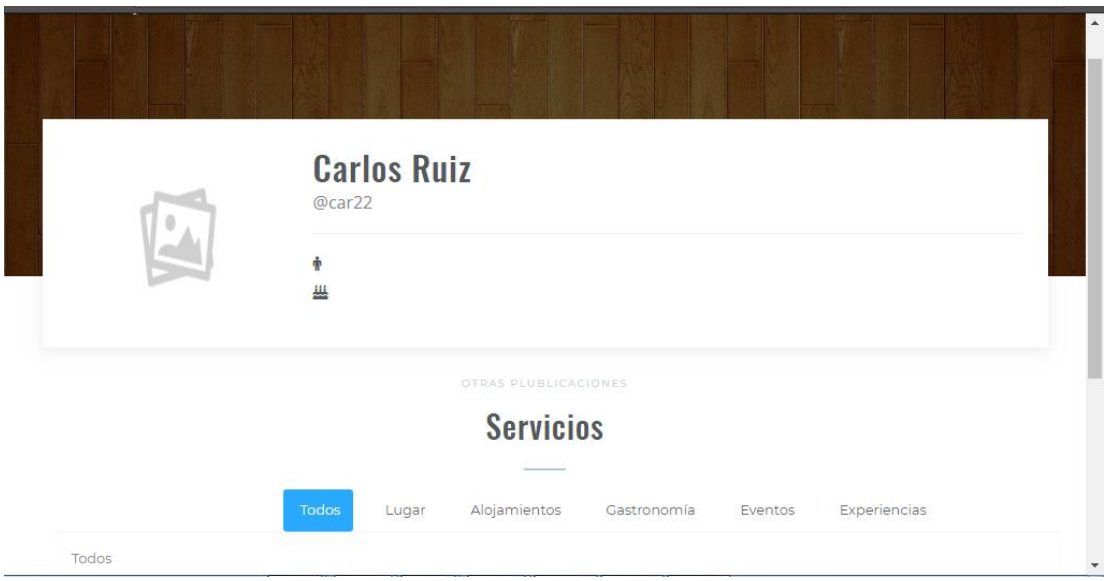


Figura BB. 10.Caso de prueba A1 Agregar Prestador de Servicio por Administrador

Se repite el formulario para el caso de prueba A2 como se muestra en la siguiente figura.

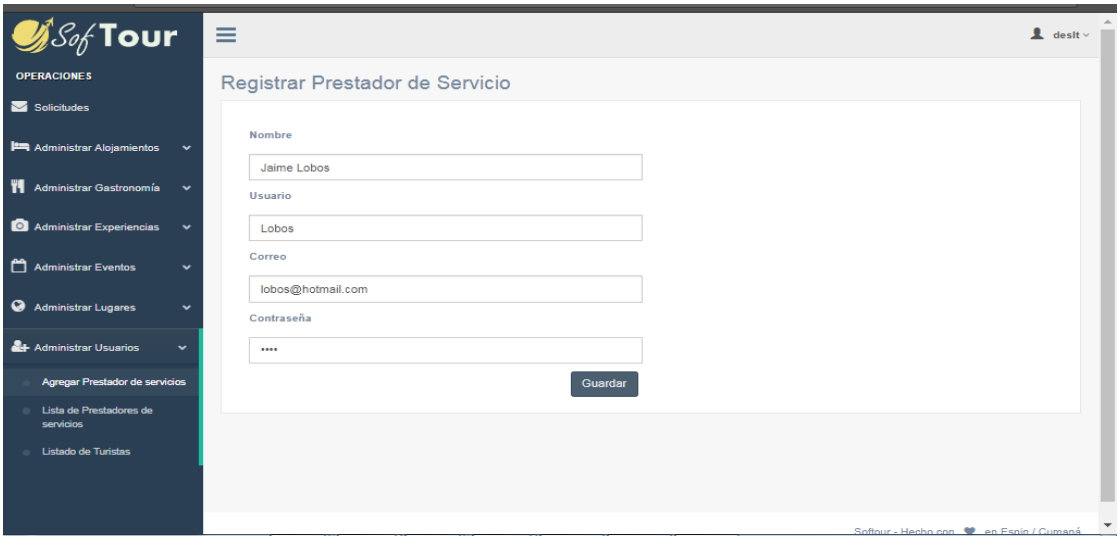


Figura BB. 11.Caso de prueba A2 Agregar Prestador de Servicio por Administrador

El registro se cumplió de forma exitosa como se evidencia en la siguiente figura.

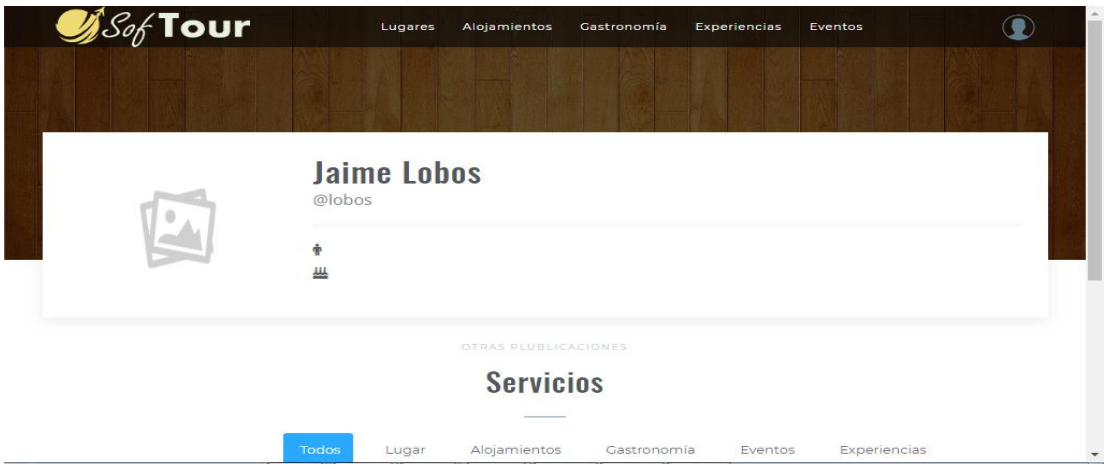


Figura BB. 12.Caso de prueba A2 Agregar Prestador de Servicio por Administrador

Continuando con el formulario para el caso de prueba A3, a continuación se muestra en la siguiente figura.

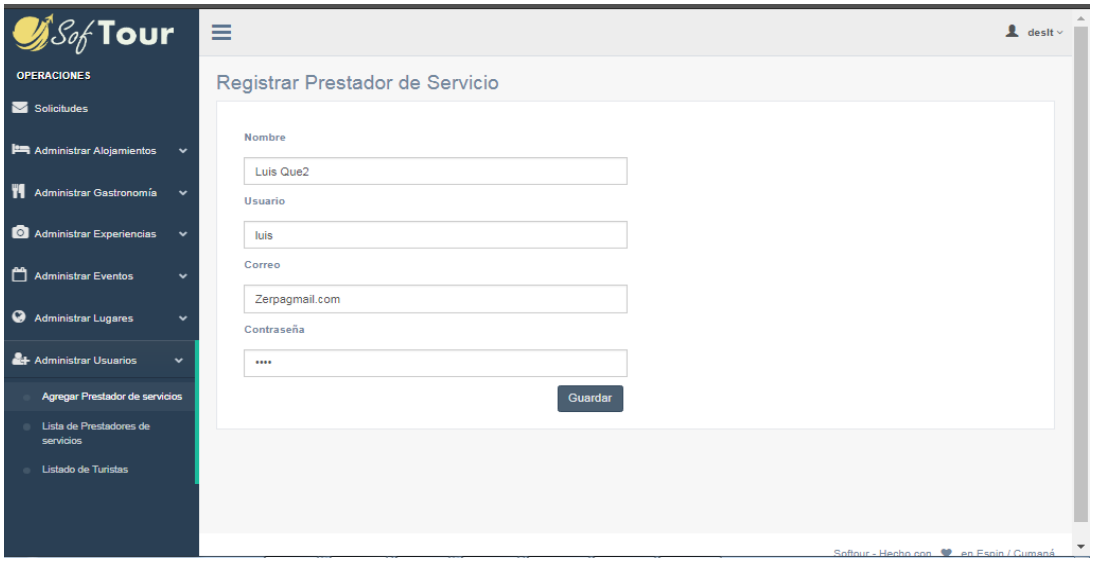


Figura BB. 13.Caso de prueba A3 Agregar Prestador de Servicio por Administrador

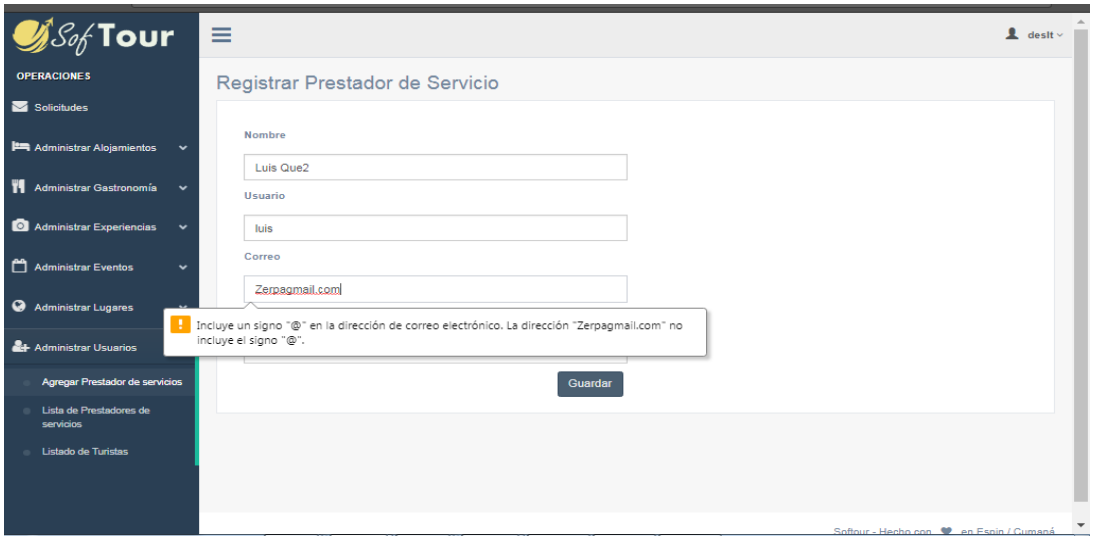


Figura BB. 14.Caso de prueba A3 Agregar Prestador de Servicio por Administrador

Conclusión de prueba

La función cumple con su objetivo y sus campos se encuentran validados para no aceptar correos no válidos.

Pruebas de interfaz y usabilidad

En análisis previo se determinó que las vistas de usuario Administrador cumple con principios de usabilidad.

Apéndice CC. Ciclo de Versión Administrador: Iteración Elaborar Listado de Prestador de Servicios.

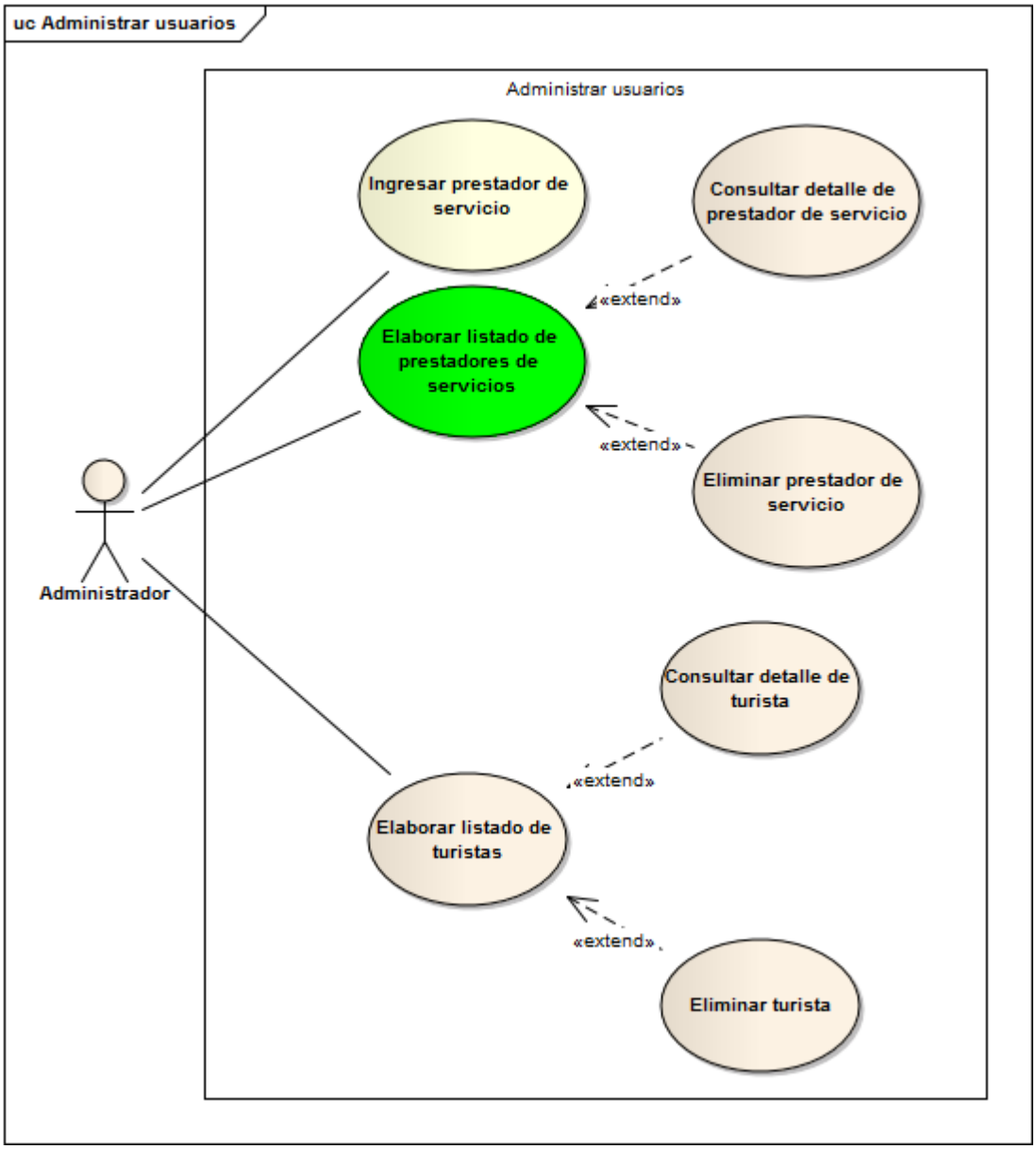


Figura CC. 1.Caso de uso: Elaborar Listado de prestadores de servicios por administrador

Tabla CC. 1. Descripción Caso de uso Elaborar Listado de prestadores de servicios por administrador

Nombre del Caso de uso	Elaborar Listado de Prestadores de Servicios	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en elaborar el listado de los prestadores de servicios actualmente registrados en el sistema.	
Precondición	Para elaborar el listado de los prestadores de servicios al menos debe existir uno de ellos registrado en la BD del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Lista de Prestadores de Servicio".
	2	El sistema realiza la búsqueda de los prestadores de servicios registrados en la BD del sistema y muestra un listado de los mismos.
	3	El Administrador visualiza el listado de los prestadores de servicios.
	4	El sistema finaliza el caso de uso.
Postcondición	Se elaboró el listado de los prestadores de servicio satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
	2.1.	El sistema, al realizar la búsqueda, no encuentra prestador de servicios registrado en la BD, muestra un mensaje "No se han agreado Prestadores de Servicios a la base de datos- ¿Desea hacerlo ahora?" y cierra el caso de uso.
	3.1.	El Administrador decide extender <<extend>> la operación sobre los prestadores de servicios que se muestran en el listado a través de los siguientes casos de uso: a.- Consultar Prestador de Servicios b.- Eliminar Prestador de Servicios .
Reglas del negocio	1. Debe existir al menos un prestador de servicios registrado para elaborar el listado. 2. Una vez elaborado el listado de los Prestadores de Servicios, se podrán escoger operaciones como "Consultar" y "Eliminar" sobre algún prestador de servicios específico.	
Inclusiones	Ninguna.	
Extensiones	Consultar Prestador de Servicios. Eliminar prestador de servicios.	

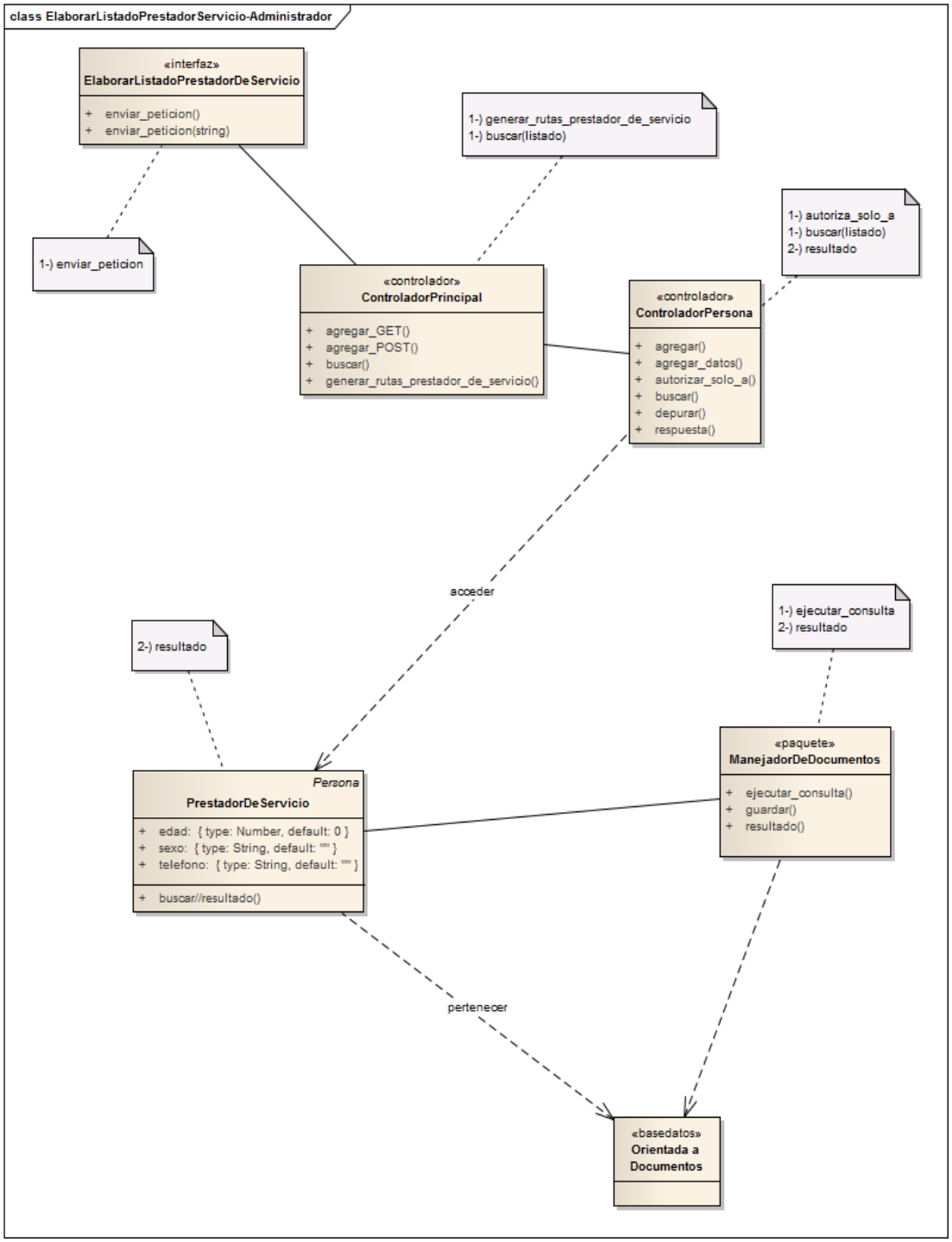


Figura CC. 2. Diagrama de Clases: Elaborar Listado de prestadores de servicios por administrador

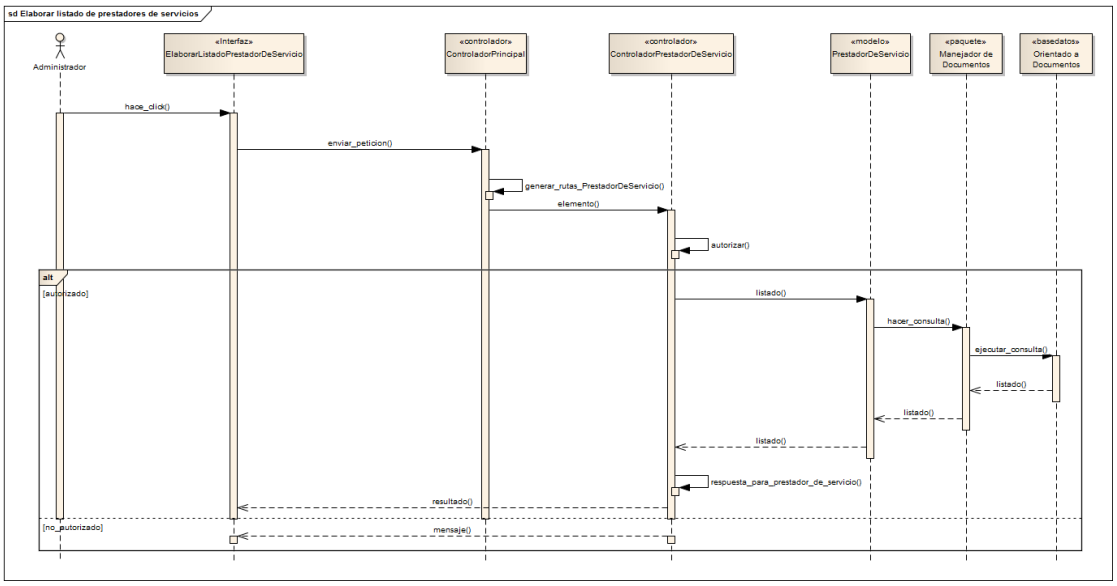


Figura CC. 3.Diagrama de secuencia: Elaborar Listado de prestadores de servicios por administrador

A continuación, se muestran el modelo de plantilla para el proceso de listado de prestador de servicio.

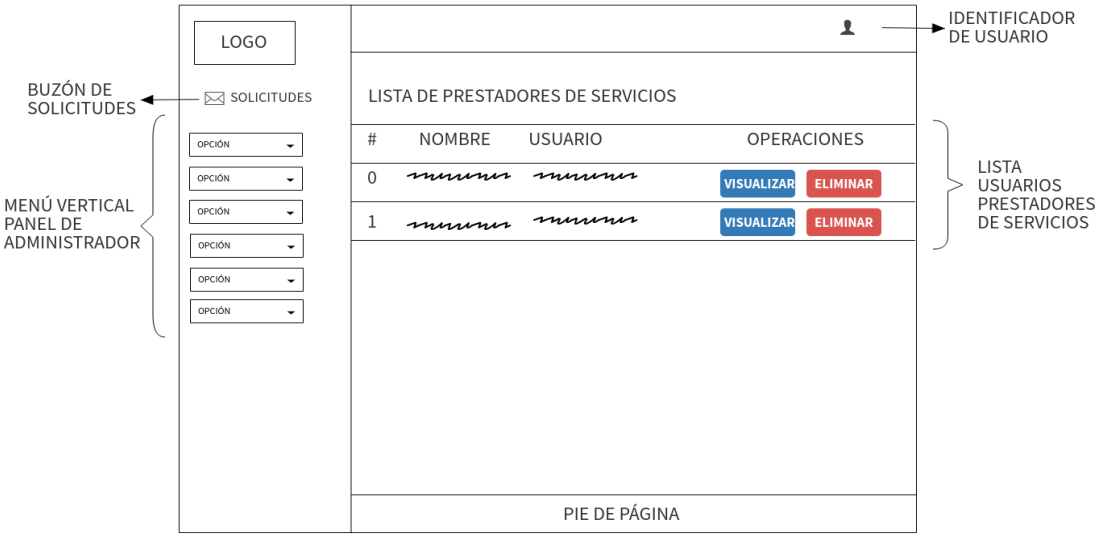


Figura CC. 4. Plantilla de interfaz: Listado Prestador de Servicio por usuario administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Lista de Prestadores de servicios y el sistema responde mostrando la interfaz correspondiente.

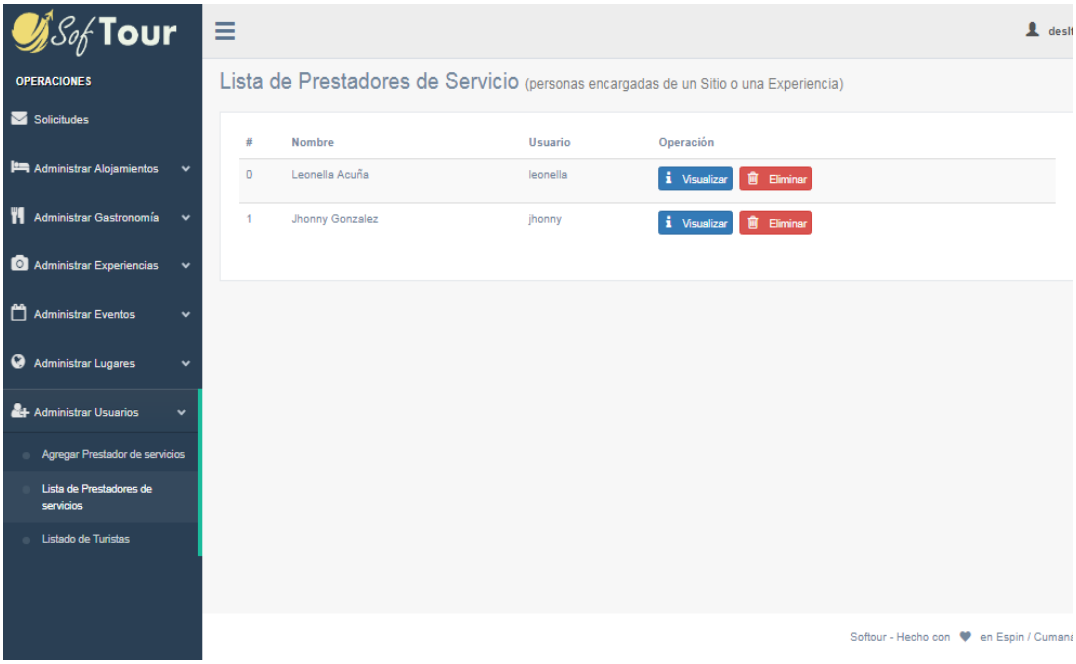


Figura CC. 5. Interfaz Listado de prestadores de servicios por administrador

Parte del código desarrollado para la funcionalidad Listado de prestadores de servicios por administrador.

```
listado(req, res, next) {
    //let tipo_usr
    if(req.params.tipo_usr == "prestadores-servicio")
        req.params.tipo_usr = "PrestadorServicio";
    else
        req.params.tipo_usr = "Turista";
    this._Persona.listado({"tipo_usr": req.params.tipo_usr, "cant_resultados": null}, (error, lista) => {
        if(error) {
            res.locals.datos.error = error;
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_listado(req, res);
        }
        res.locals.datos = {"value": lista};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;

        this._respuesta.para_listado(req, res);
    })
}
```

Figura CC. 6. Código desarrollado para el proceso Listar Prestador de Servicio por usuario Administrador.

Apéndice DD. Ciclo de Versión Administrador: Iteración Eliminar Prestador de Servicios.

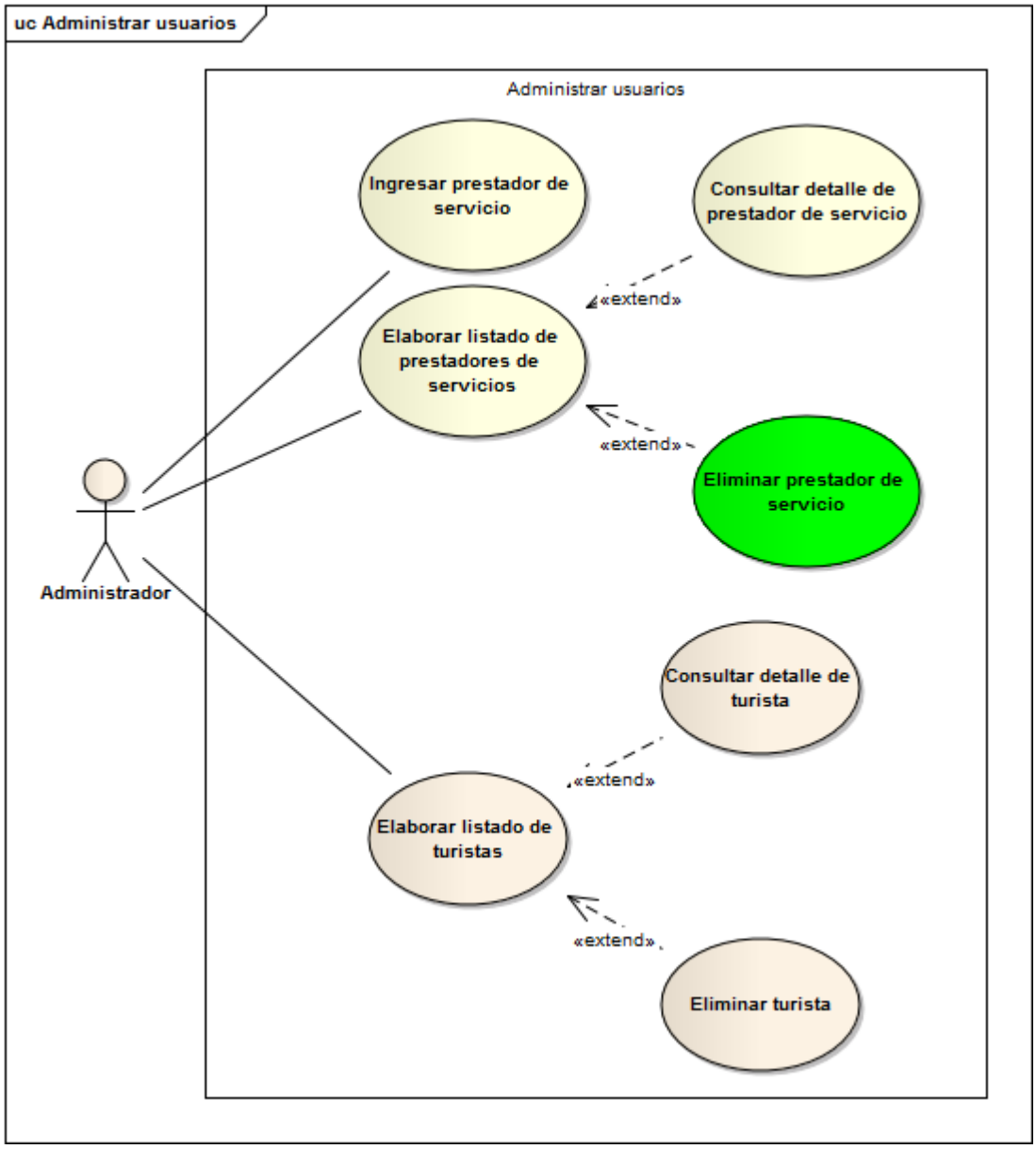


Figura DD. 1. Caso de uso: Eliminar prestador de servicio por administrador

Tabla DD. 1. Descripción Caso de uso Eliminar prestador de servicio por administrador

Nombre del Caso de uso	Eliminar Prestador de Servicios	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en eliminar el prestador de servicios de manera satisfactoria.	
Precondición	Para eliminar un prestador de servicios, el mismo debe estar registrado en el sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción “Eliminar” en el listado de los prestadores de servicios.
	2	El sistema muestra un mensaje “¿Está seguro de eliminar la cuenta del prestador seleccionado?”
	3	El administrador selecciona la opción “Si, estoy seguro”
	4	El sistema realiza la eliminación del Prestador de Servicios en el BD y finaliza el caso de uso.
Postcondición	Se elimina el Prestador de Servicios de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1	El administrador selecciona la opción “cancelar”
	3.2	El sistema envía al administrador a la pantalla principal.
Reglas del negocio	1. El prestador de servicios debe aparecer desplegado en el listado con la opción eliminar.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

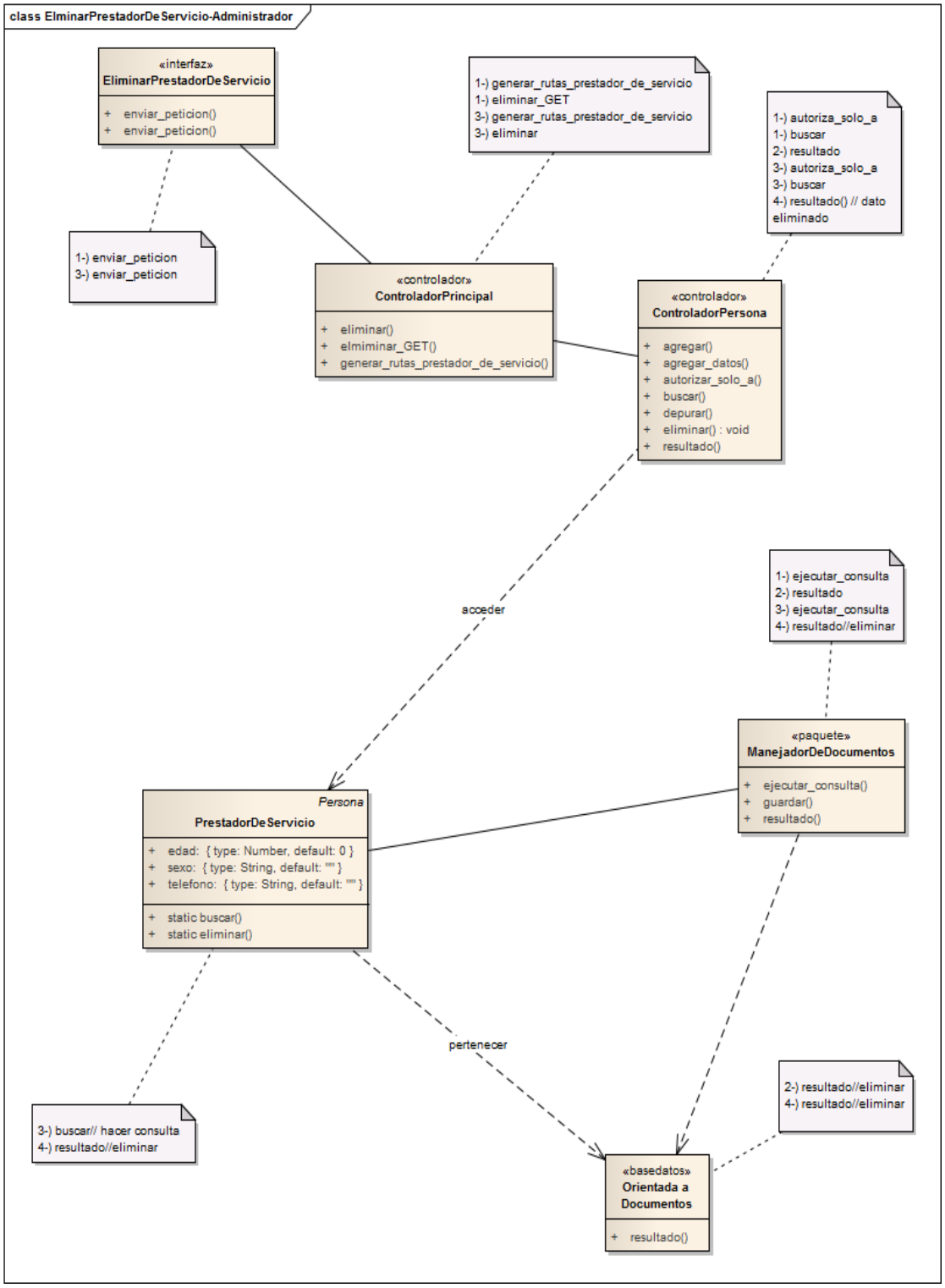


Figura DD. 2. Diagrama de Clases: Eliminar Prestador de Servicios por usuario administrador

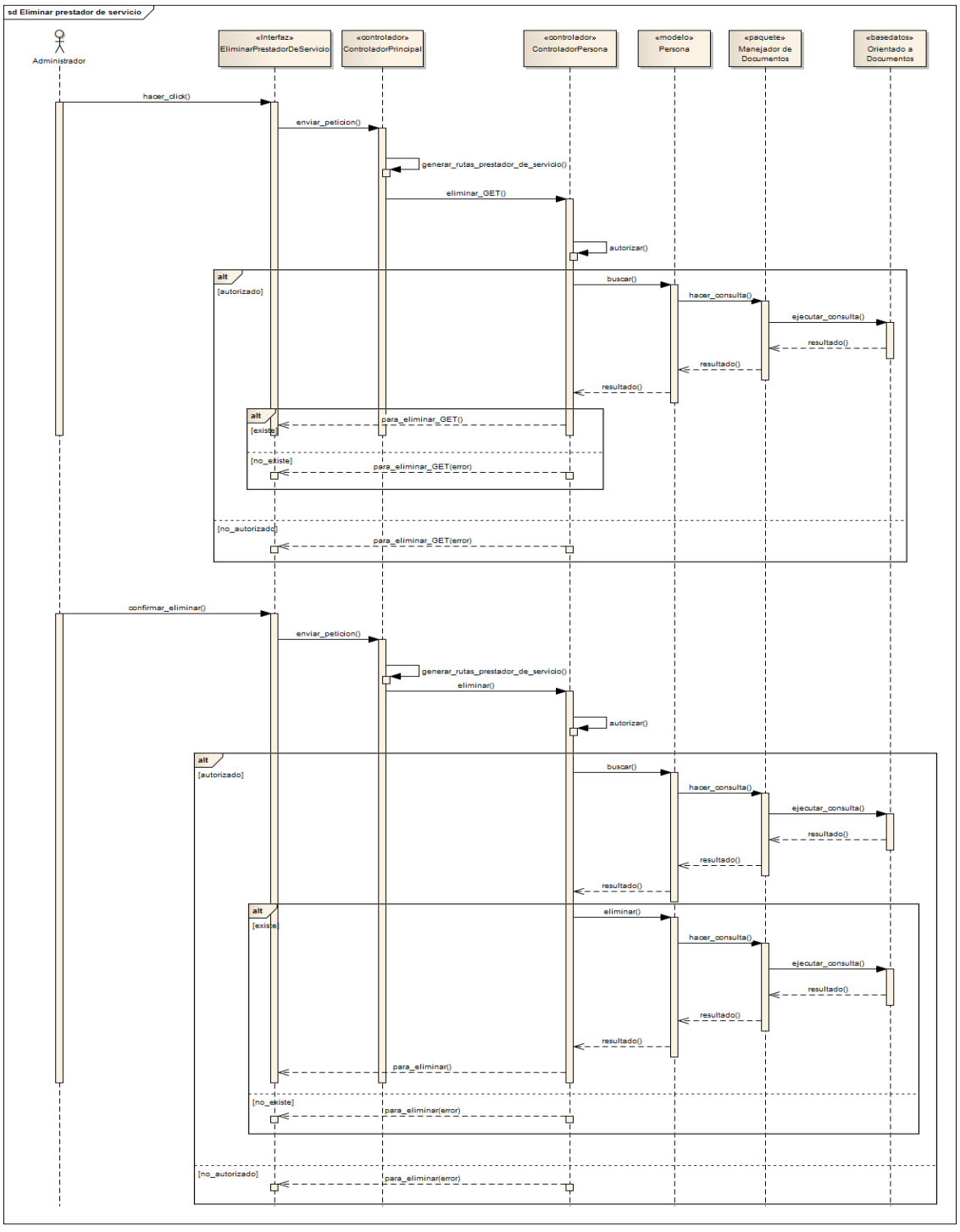


Figura DD. 3.Diagrama de secuencia: Eliminar Prestador de Servicio por usuario administrador

A continuación, se muestran el modelo de plantilla para el proceso de eliminar prestador de servicio

Figura DD. 4. Plantilla de interfaz: Eliminar Prestador de Servicio por usuario administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Eliminar prestador de servicio y el sistema responde mostrando la interfaz de confirmación de acción.

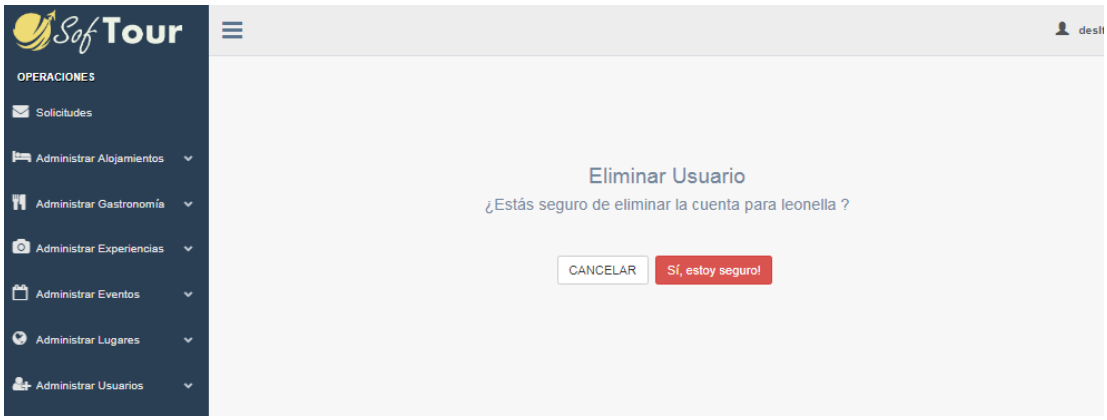


Figura DD. 5. Interfaz de proceso de Eliminar Prestador de servicio por usuario administrador

Parte del código desarrollado para la funcionalidad Eliminar Prestador de Servicio por usuario administrador.

```
eliminar(req, res, next) {

    this._Persona.eliminar(req.params.usuario.toLowerCase(), (error, eliminado) => {
        if(error) {
            res.locals.datos.error = error;
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_eliminar(req, res);
        }
        if(!eliminado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": {"usuario": "listado-prestadores-servicio"}};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_eliminar(req, res);
    });
}
```

Figura DD. 6.Código desarrollado para el proceso Eliminar Prestador de Servicio por usuario Administrador.

Prueba eliminar prestador de servicio

Esta prueba se llevó a cabo utilizando datos previos de pruebas anteriores, consistió en validar el listado de prestadores de servicios y proceder a hacer clic en el botón Eliminar y comprobar que se cumpla la función.

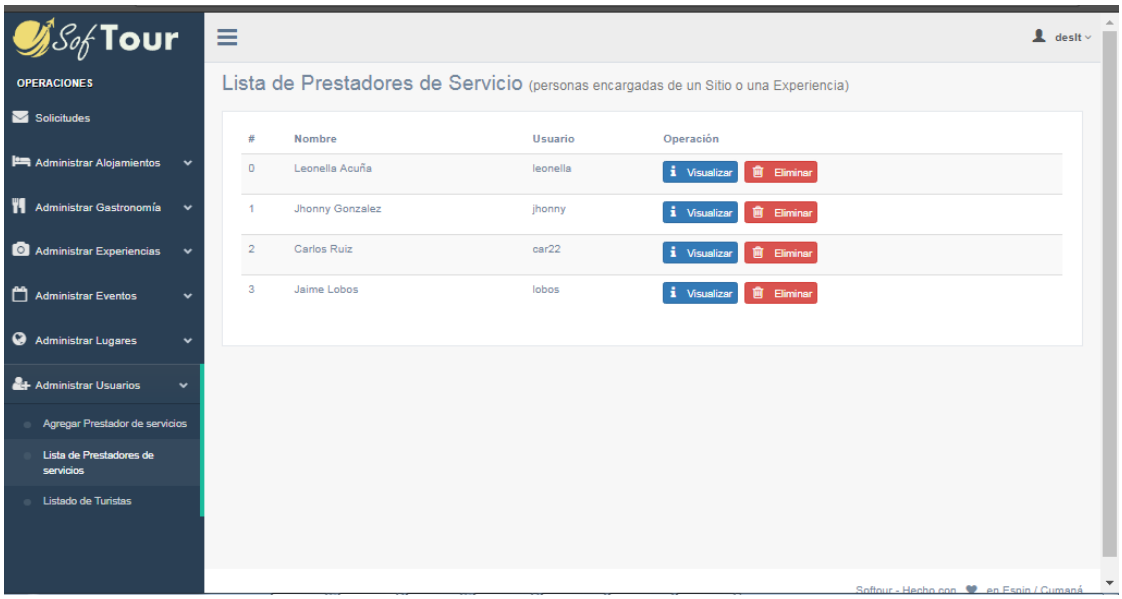


Figura DD. 7.Prueba eliminar prestador de servicio por administrador

El sistema consulta al usuario si quiere seguir con la acción mediante una interfaz única para esto, como se muestra en la siguiente imagen:

Figura DD. 8.Prueba eliminar prestador de servicio por administrador

A continuación, se muestra el Proceso cumplido exitosamente

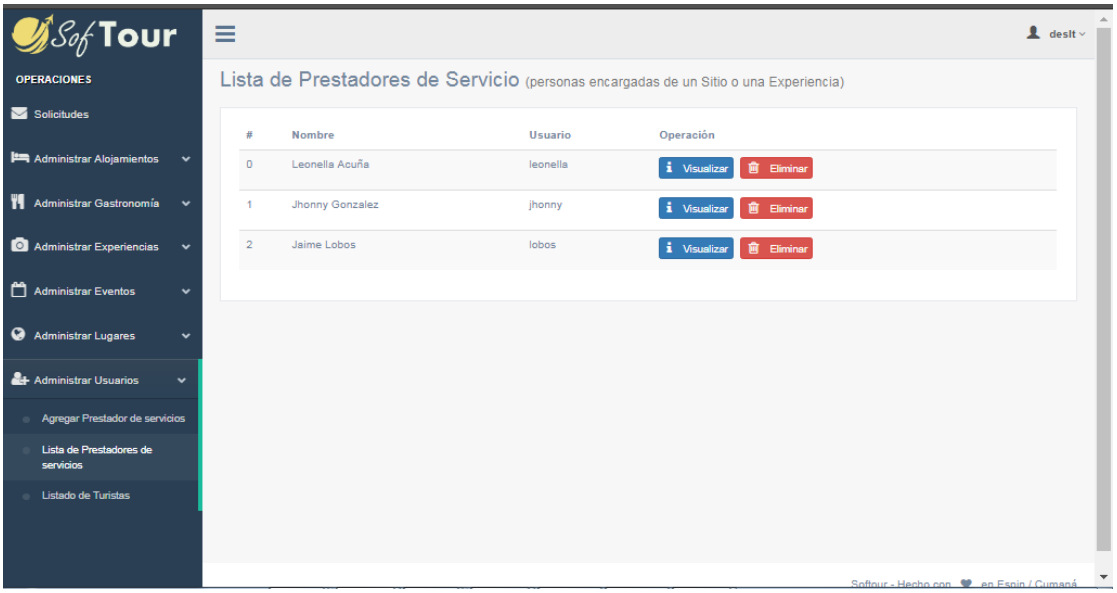


Figura DD. 9.Prueba eliminar prestador de servicio por administrador

Conclusión de prueba

La función cumple a cabalidad su objetivo.

Prueba de interfaz y usabilidad

Por análisis previo se determinó que el conjunto de interfaz de usuario Administrador cumple con estándares de usabilidad que le aseguran una buena interacción con los usuarios

Apéndice EE. Ciclo de Versión Administrador: Iteración Elaborar Listado Turista.

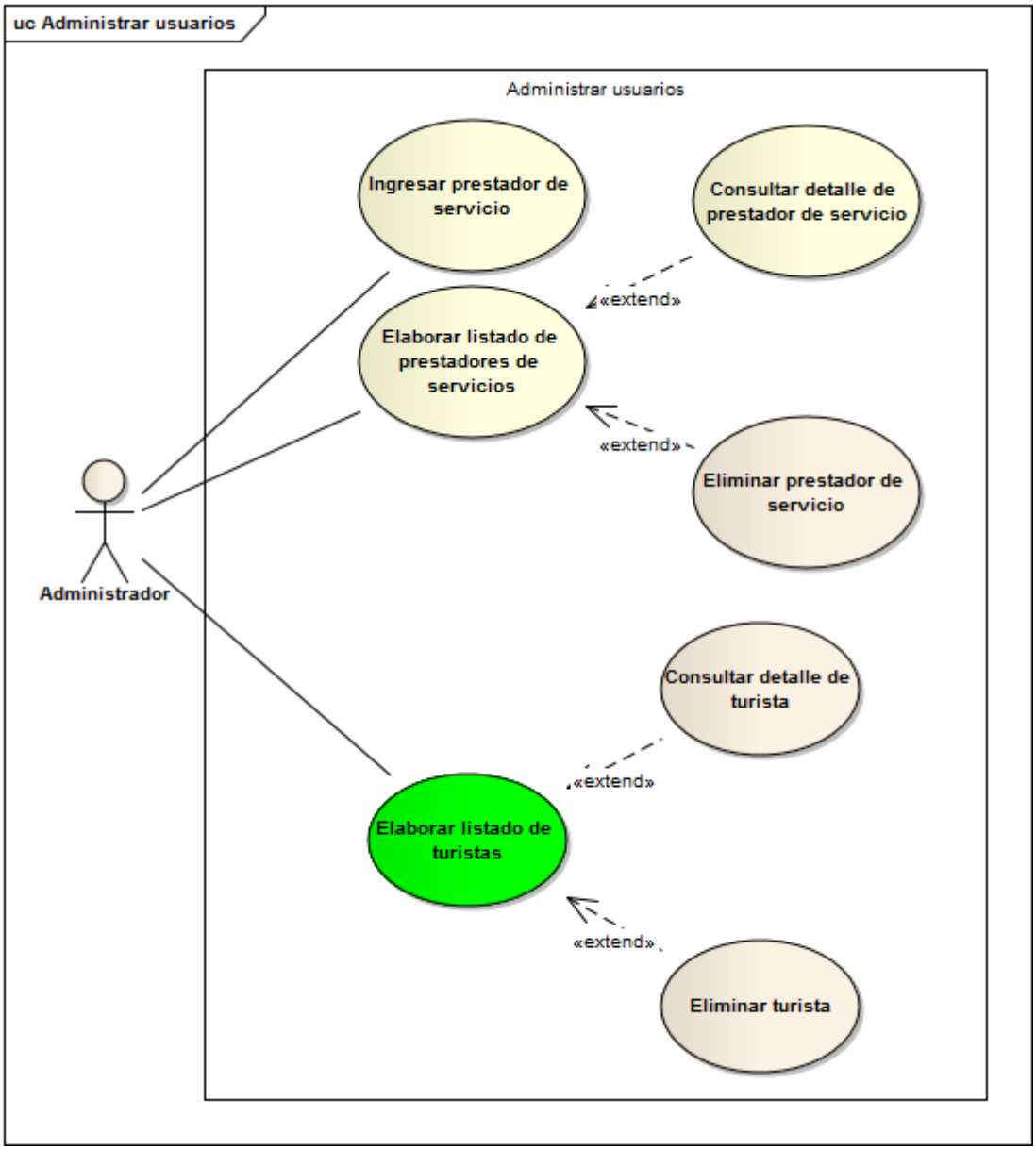


Figura EE. 1.Caso de uso: Elaborar Listado de turistas por administrador

Tabla EE. 1. Descripción Caso de uso Elaborar Listado de turistas por administrador

Nombre del Caso de uso	Elaborar Listado de Turista	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en elaborar el listado de los turistas actualmente registrados en el sistema.	
Precondición	Para elaborar el listado de los turistas al menos debe existir uno de ellos registrado en la BD del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Listado de turistas".
	2	El sistema realiza la búsqueda de los turistas registrados en el BD del sistema y muestra un listado de los mismos.
	3	El Administrador visualiza el listado de los turistas.
	4	El sistema finaliza el caso de uso.
Postcondición	Se elaboró el listado de los turistas satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
	2.1.	El sistema, al realizar la búsqueda, no encuentra turista registrado en el BD, muestra un mensaje "no hay usuarios registrados en el sistema" y cierra el caso de uso.
	3.1.	El Administrador decide extender <<extend>> la operación sobre los turistas que se muestran en el listado a través de los siguientes casos de uso: a.- Consultar turistas b.- Eliminar turistas
Reglas del negocio	1. Debe existir al menos un turista registrado para elaborar el listado. 2. Una vez elaborado el listado de los turistas, se podrán escoger operaciones como "Consultar" y "Eliminar" sobre algún turista específico.	
Inclusiones	Ninguna.	
Extensiones	Consultar turista. Eliminar turista.	

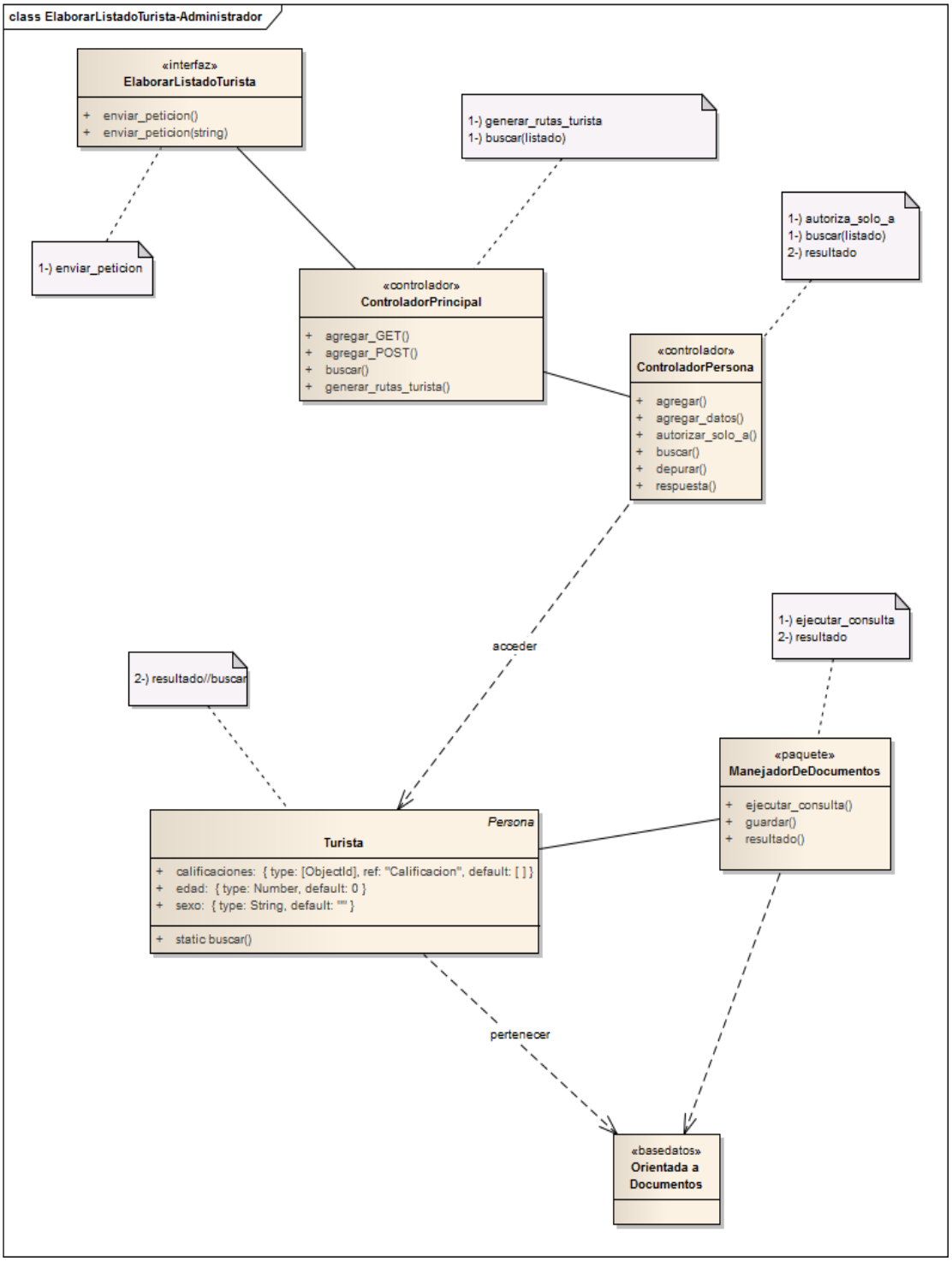


Figura EE. 2.Diagrama de Clases: Elaborar Listado de turistas por administrador

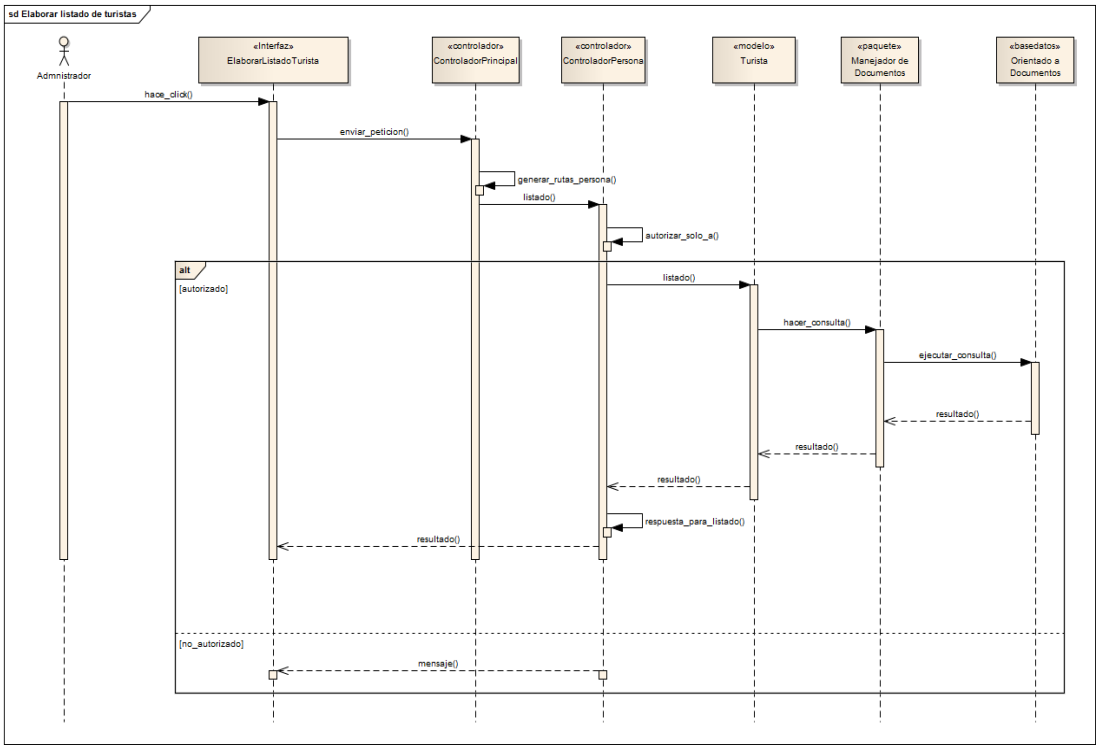


Figura EE. 3.Diagrama de secuencia: Elaborar Listado de turistas por administrador

A continuación, se muestran el modelo de plantilla para el proceso de listado de turista.

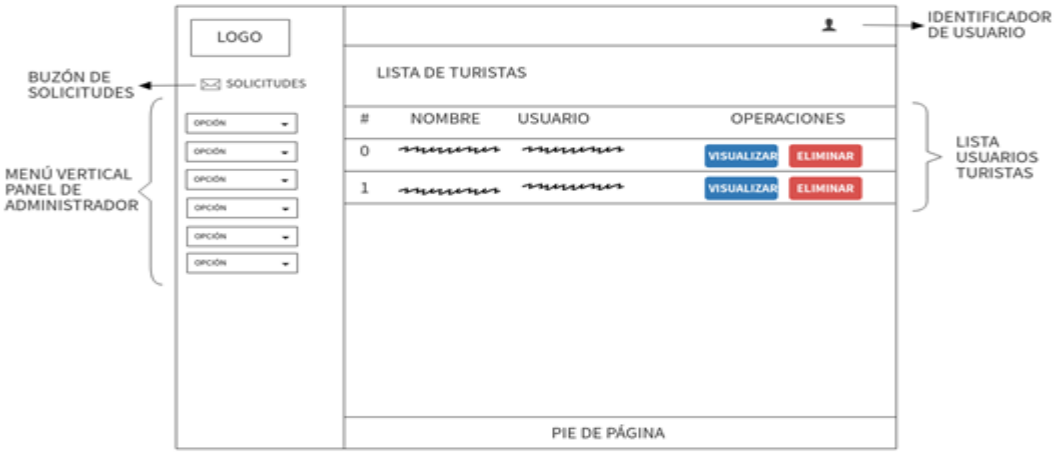


Figura EE. 4.Plantilla de interfaz: Listado Turista por usuario administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Lista de Turistas y el sistema responde mostrando la interfaz correspondiente.

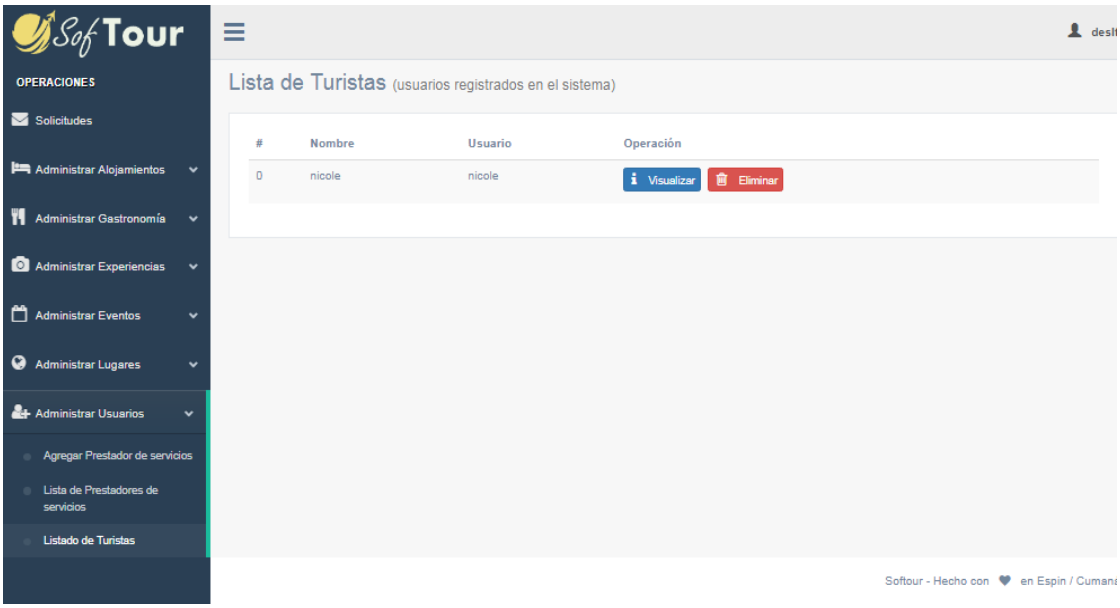


Figura EE. 5. Interfaz Listado de turistas por administrador

Parte del código desarrollado para la funcionalidad Listado de turistas por administrador.

```
listado(req, res, next) {
    //let tipo_usr
    if(req.params.tipo_usr == "prestadores-servicio")
        req.params.tipo_usr = "PrestadorServicio";
    else
        req.params.tipo_usr = "Turista";
    this._Persona.listado({"tipo_usr": req.params.tipo_usr, "cant_resultados": null}, (error, lista) => {
        if(error) {
            res.locals.datos.error = error;
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_listado(req, res);
        }
        res.locals.datos = {"value": lista};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;

        this._respuesta.para_listado(req, res);
    })
}
```

Figura EE. 6. Código desarrollado para el proceso Listar Turista por usuario Administrador.

Apéndice FF. Ciclo de Versión Administrador: Iteración Consultar Detalle Turista.

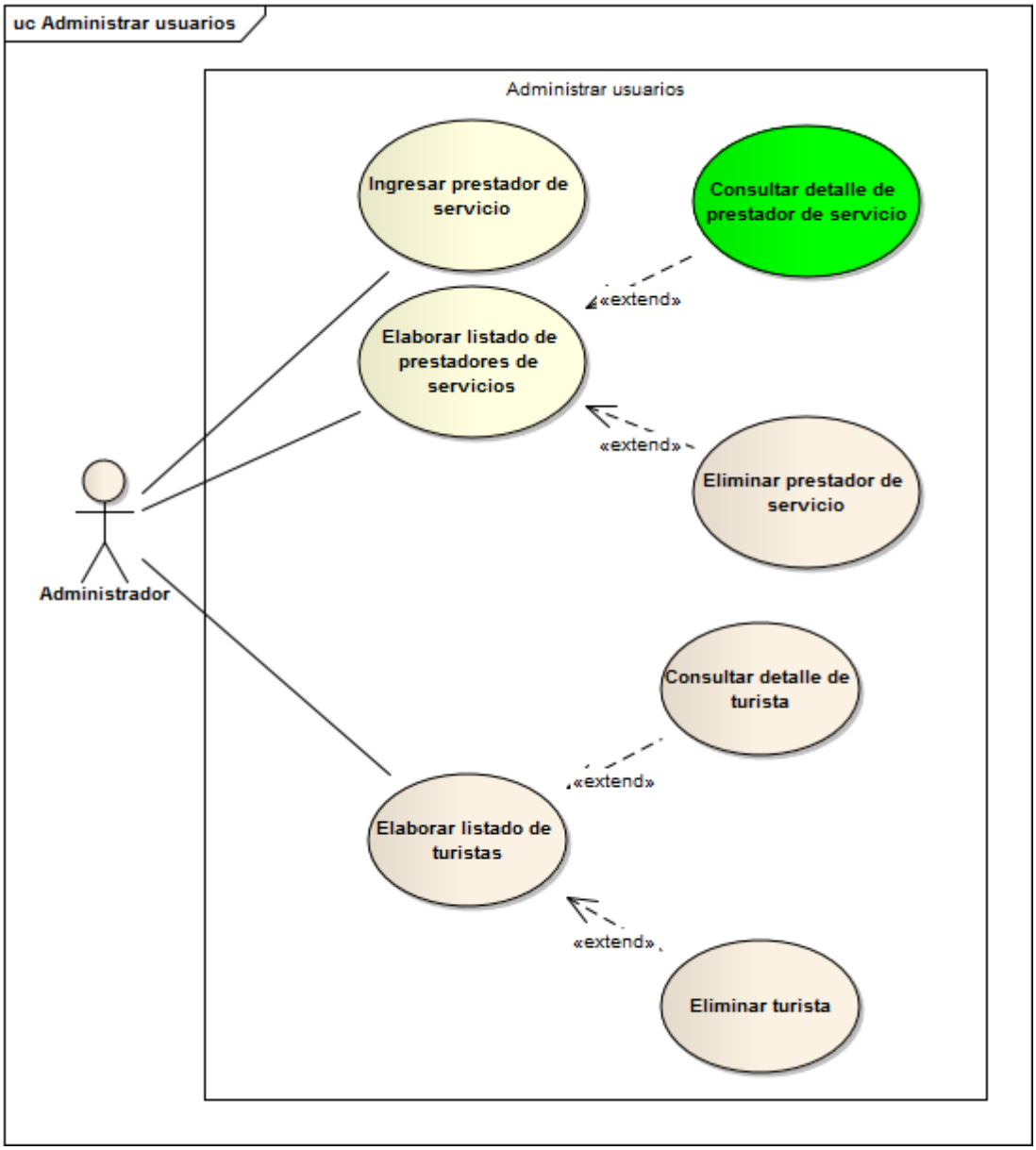


Figura FF. 1. Caso de uso: Consultar Detalle de prestador de servicio por administrador

Tabla FF. 1. Descripción Caso de uso Consultar Detalle de prestador de servicio por administrador

Nombre del Caso de uso	Consultar Prestador de Servicios	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en Consultar el prestador de servicios de manera satisfactoria.	
Precondición	Para consultar un prestador de servicios, el mismo debe estar registrado en el sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Consultar" en el listado de los prestadores de servicios.
	2	El sistema realiza la consulta del Prestador de Servicios en la BD y finaliza el caso de uso.
Postcondición	Se consulta el Prestador de Servicios de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
Reglas del negocio	1. El prestador de servicios debe estar registrado para que pueda consultar.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

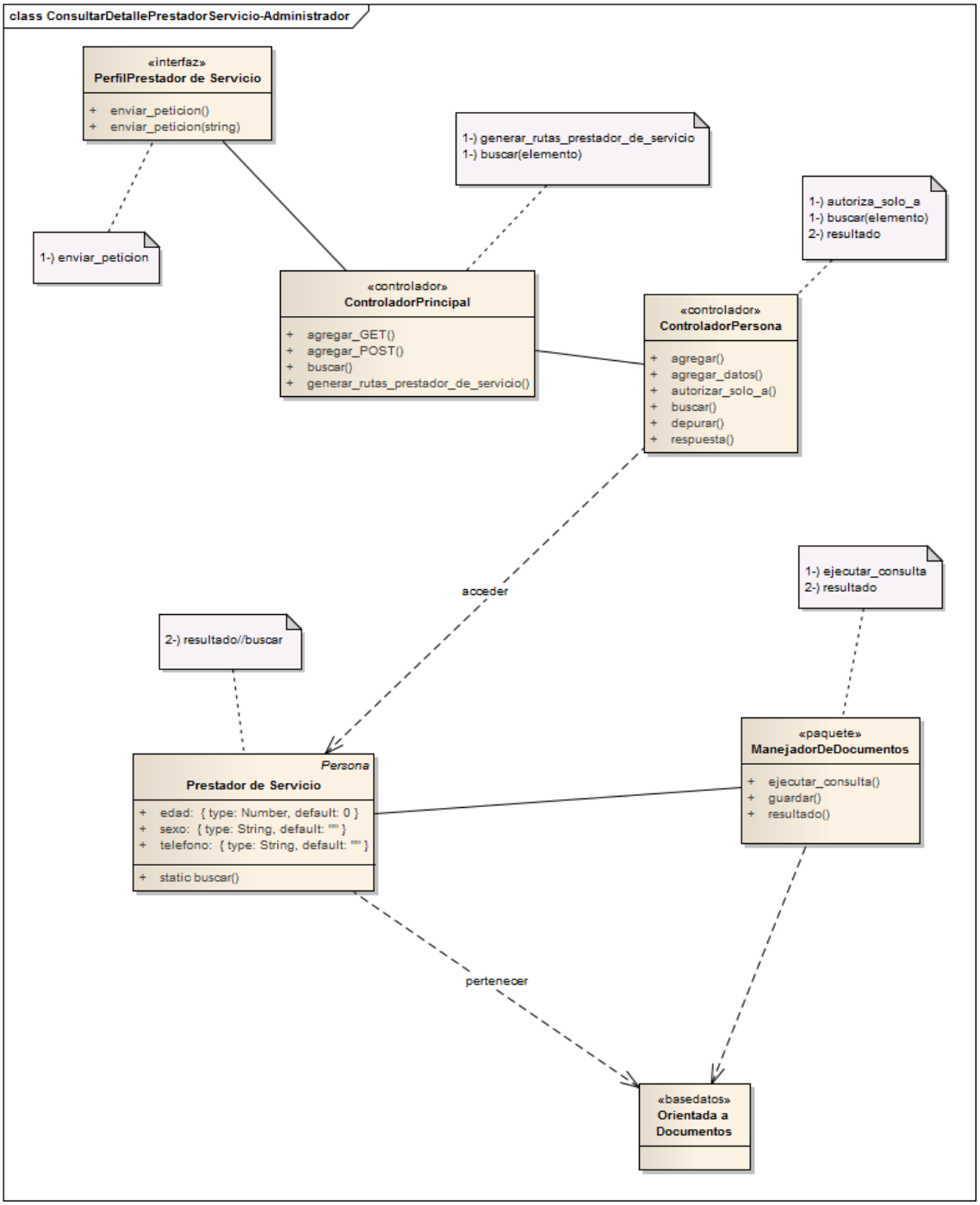


Figura FF. 2.Diagrama de Clases: Consultar Detalle de Prestador de Servicios por usuario administrador

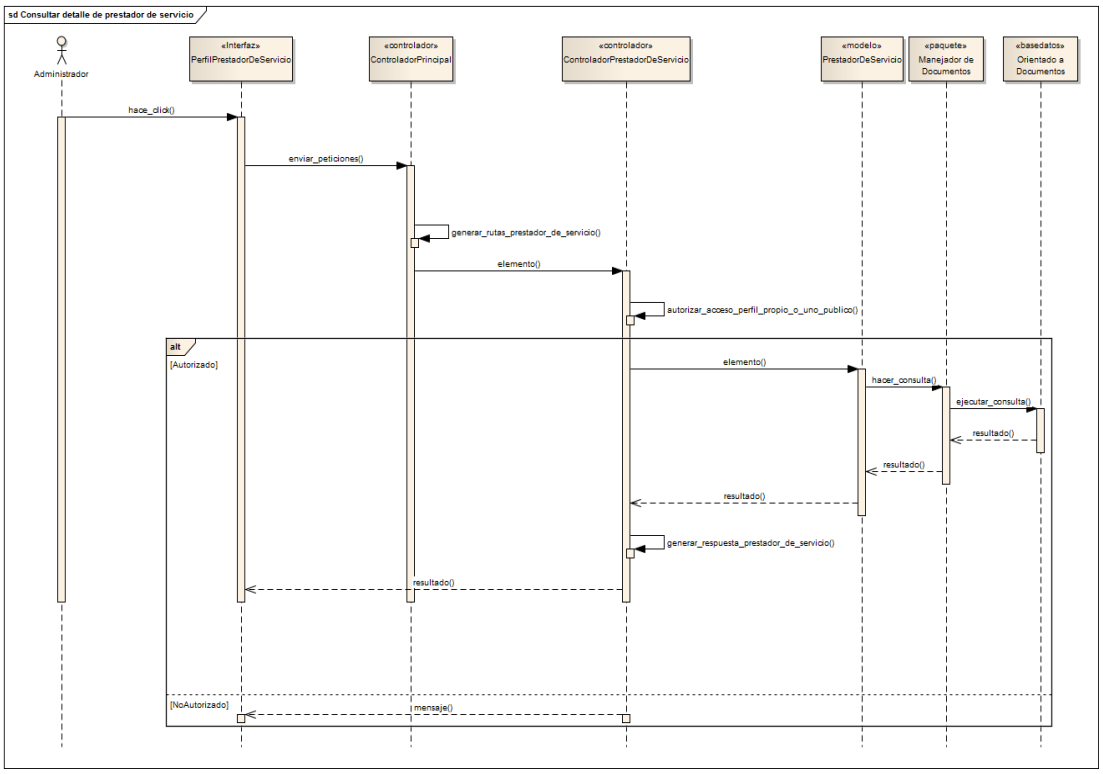


Figura FF. 3. Diagrama de secuencia: Consultar Detalle de Prestador de Servicio por usuario administrador

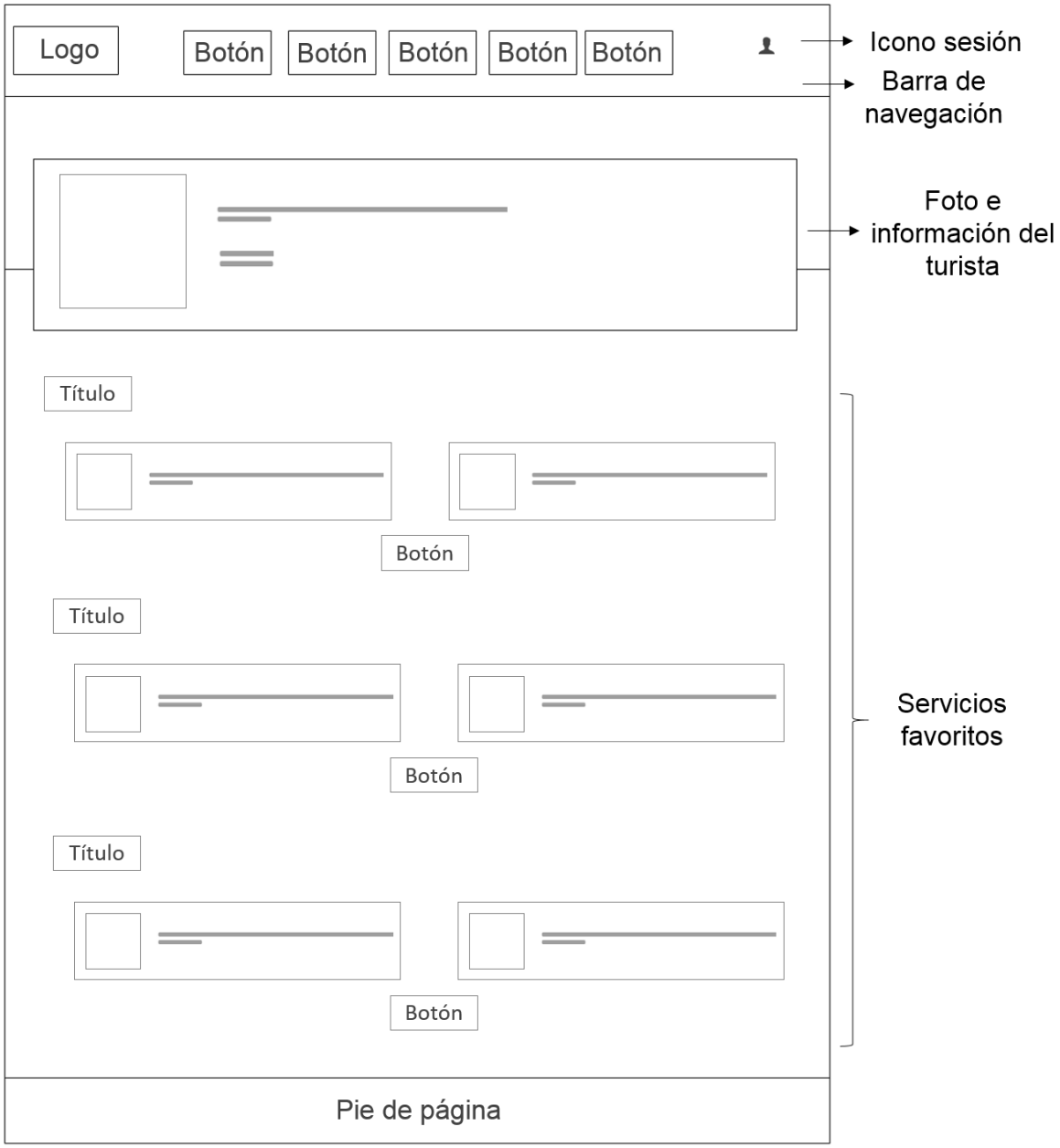


Figura FF. 4. Diseño del formato físico para la pantalla Consultar Detalle de Prestador de servicio por usuario administrador

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Consultar detalle de prestador de servicio

y el sistema responde mostrando la interfaz con las especificaciones.

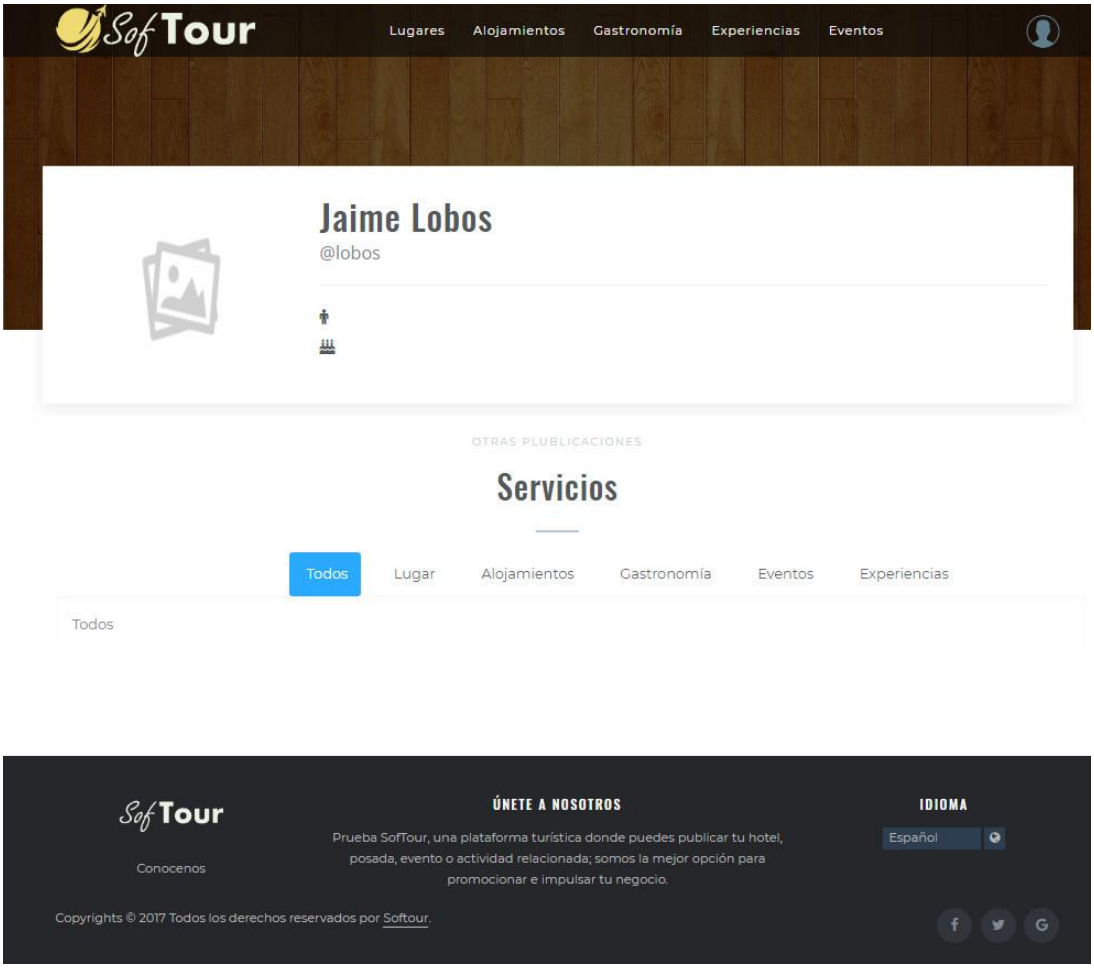


Figura FF. 5. Interfaz de proceso de Consultar Detalle de Prestador de servicio por usuario administrador

Parte del código desarrollado para la funcionalidad Consultar Detalle Prestador de Servicio por usuario administrador

```
consultar(req, res, next) {
    this._Persona.buscar(req.params.usuario.toLowerCase(), (error, buscado) => {
        if(error) {
            res.locals.datos.error = error;
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        if(buscado.tipo_usr == "PrestadorServicio")
            async.parallel({
                lista_alojamientos: (cb) => this._Alojamiento.find({"prestador_servicio": buscado.id}, cb),
                lista_gastronomia: (cb) => this._Gastronomia.find({"prestador_servicio": buscado.id}, cb),
                lista_eventos: (cb) => this._Evento.find({"prestador_servicio": buscado.id}, cb),
                lista_experiencias: (cb) => this._Experiencia.find({"prestador_servicio": buscado.id}, cb)
             }, (error, resultados) => {
                if(error) {
                    res.locals.datos.error = error;
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_consultar(req, res);
                }
                res.locals.datos = {
                    "value": {
                        "persona": buscado,
                        "lista_alojamientos": resultados.lista_alojamientos,
                        "lista_eventos": resultados.lista_eventos,
                        "lista_experiencias": resultados.lista_experiencias,
                        "lista_gastronomia": resultados.lista_gastronomia
                    }
                };
                if (req.isAuthenticated())
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;

                this._respuesta.para_consultar(req, res);

            })
        else if(buscado.tipo_usr == "Administrador") {
                res.locals.datos = {"value": { "persona": buscado } };
                if (req.isAuthenticated())
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;

                this._respuesta.para_consultar(req, res);
            }
        else {
            res.locals.datos = {
                "value": { "persona": buscado }
            };
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;

            this._respuesta.para_consultar(req, res);
        }
    })
}
```

Figura FF. 6 Código desarrollado para el Consultar detalles de Prestador de Servicio por usuario Administrador.

Apéndice GG. Ciclo de Versión Administrador: Iteración Eliminar Turista.

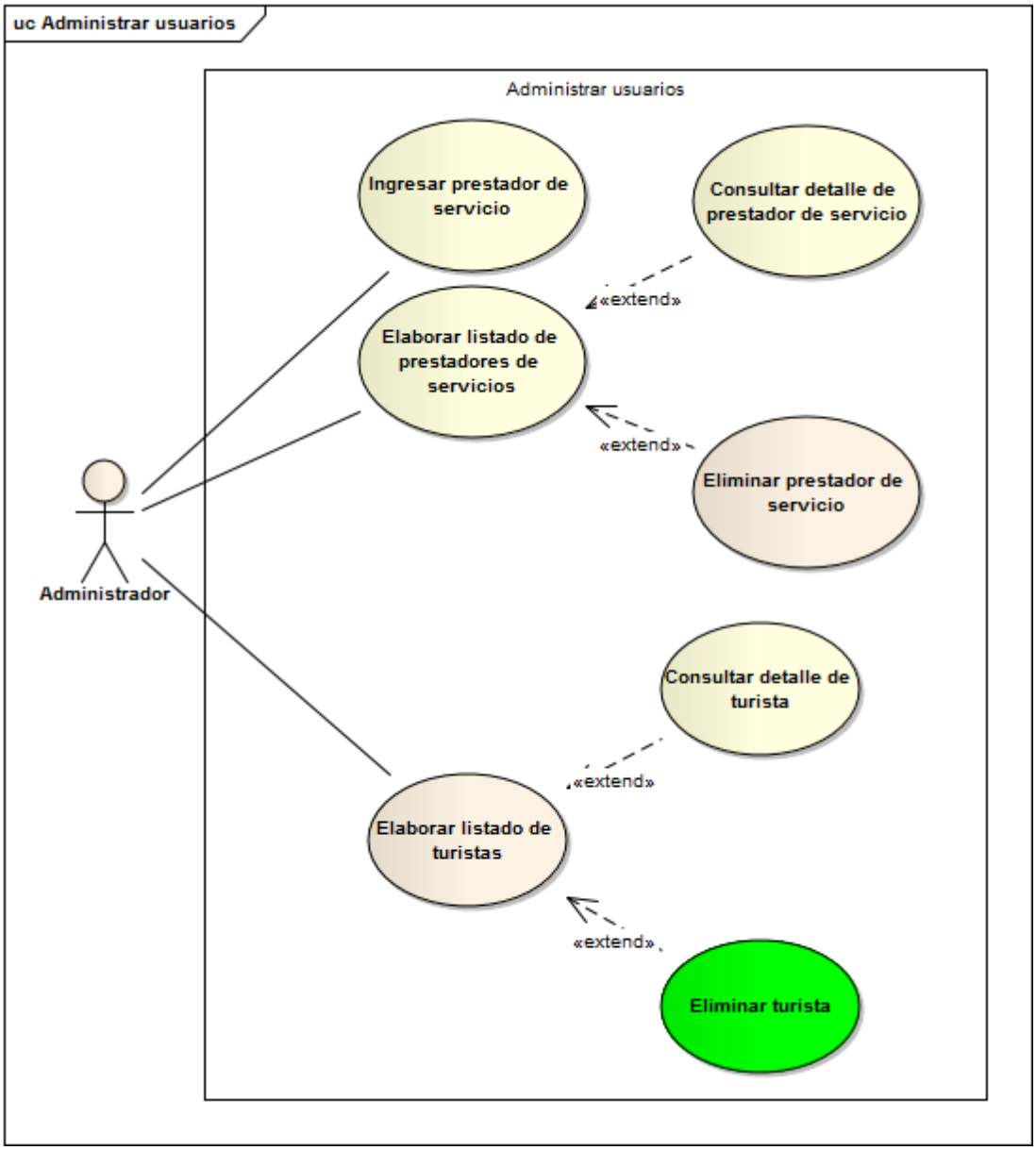


Figura GG. 1.Caso de uso: Eliminar turista por administrador

Tabla GG. 1. Descripción Caso de uso Eliminar turista por administrador

Nombre del Caso de uso	Eliminar turista	
Versión	1.0 25/10/2017	
Actor	Administrador.	
Descripción del Caso de Uso	El caso de uso consiste en eliminar el turista de manera satisfactoria.	
Precondición	Para eliminar un turista, el mismo debe estar registrado en el sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Administrador selecciona la opción "Eliminar turista" en el listado de los turistas.
	2	El sistema ejecuta la búsqueda del Turista
	3	El Administrador visualiza el Turista que desea eliminar y selecciona eliminar
	4	El sistema muestra un mensaje "¿Está seguro de eliminar la cuenta del turista seleccionado?"
	5	El administrador selecciona la opción "Si, estoy seguro"
	6	El sistema realiza la eliminación del turista en la BD y finaliza el caso de uso.
Postcondición	Se elimina el turista de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	5.1	El administrador selecciona la opción "cancelar"
	5.2	El sistema envía al administrador a la pantalla principal.
Reglas del negocio	2. El turista debe aparecer desplegado en el listado con la opción eliminar.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

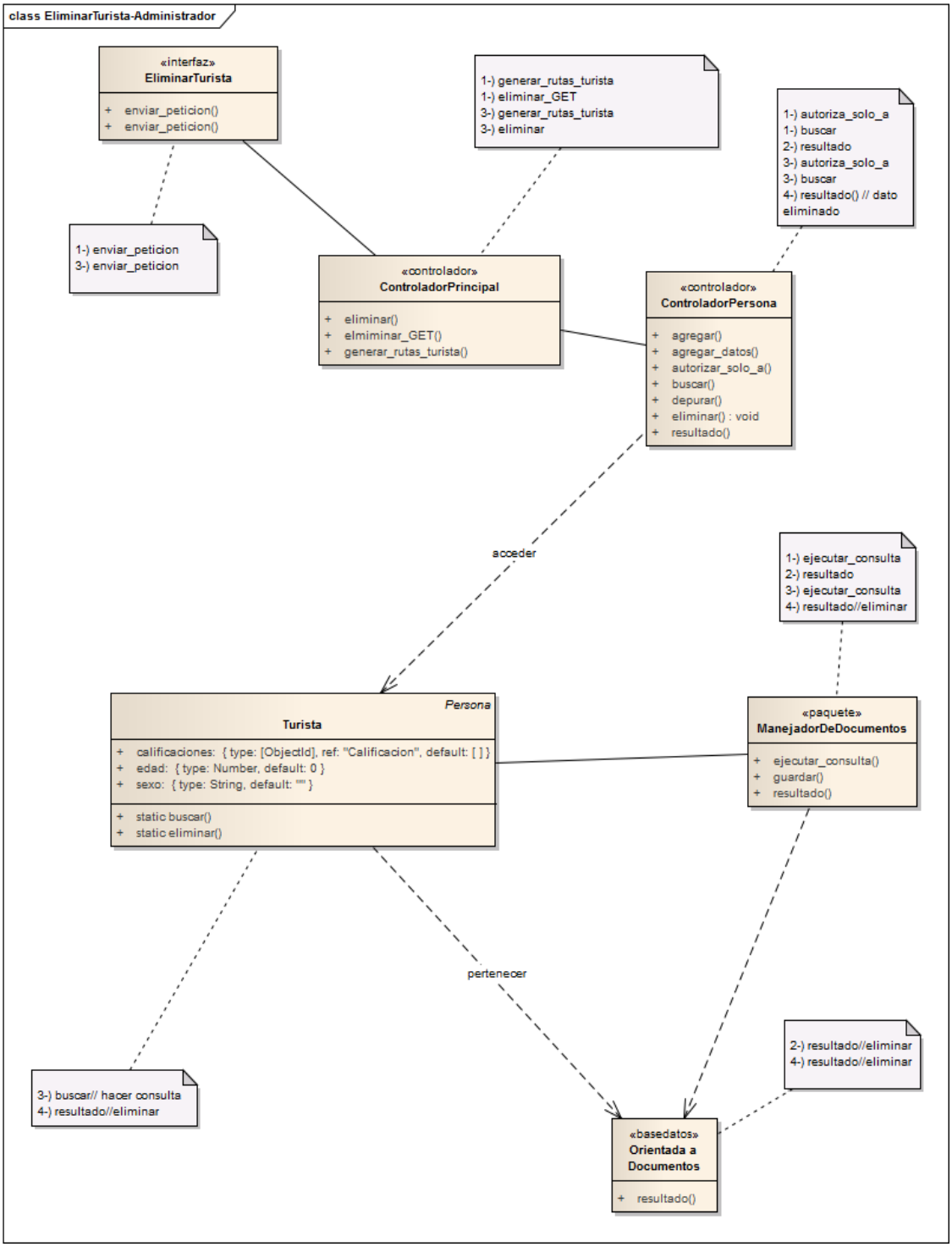


Figura GG. 2. Diagrama de Clases: Eliminar turista por usuario administrador

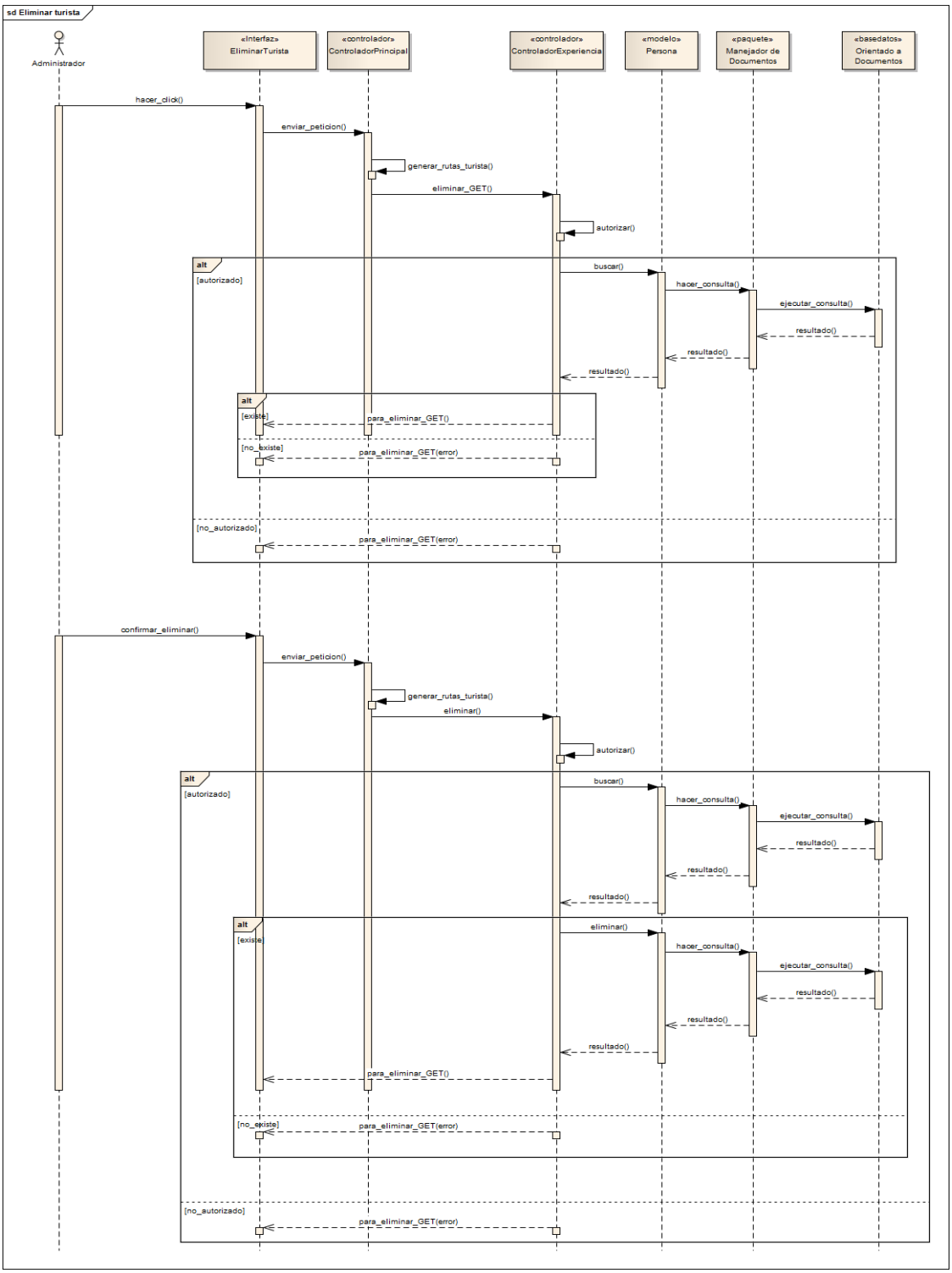


Figura GG. 3. Diagrama de secuencia: Eliminar turista por usuario administrador

A continuación, se muestran el modelo de plantilla para el proceso de eliminar turista.

Figura GG. 4. Plantilla de interfaz: Eliminar Turista por usuario administrador.

Flujo a seguir para la funcionalidad.

El administrador selecciona la opción Eliminar turista y el sistema responde mostrando la interfaz de confirmación de acción.

Figura GG. 5.Interfaz 1 de proceso de Eliminar Turista por usuario administrador

Parte del código desarrollado para la funcionalidad Eliminar Turista por usuario administrador

```
eliminar(req, res, next) {

    this._Persona.eliminar(req.params.usuario.toLowerCase(), (error, eliminado) => {
        if(error) {
            res.locals.datos.error = error;
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_eliminar(req, res);
        }
        if(!eliminado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": {"usuario": "listado-prestadores-servicio"}};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_eliminar(req, res);
    });
}
```

Figura GG. 6. Código desarrollado para el proceso Eliminar Turista por usuario Administrador.

Prueba eliminar turista

Esta función consiste en eliminar usuarios Turista del sistema, para llevar a cabo esta prueba se registraron usuarios ficticios y se procedió a eliminar a los usuarios y comprobar que la función cumplía su fin.

A continuación, se muestra secuencia de imágenes de una de las pruebas realizadas:

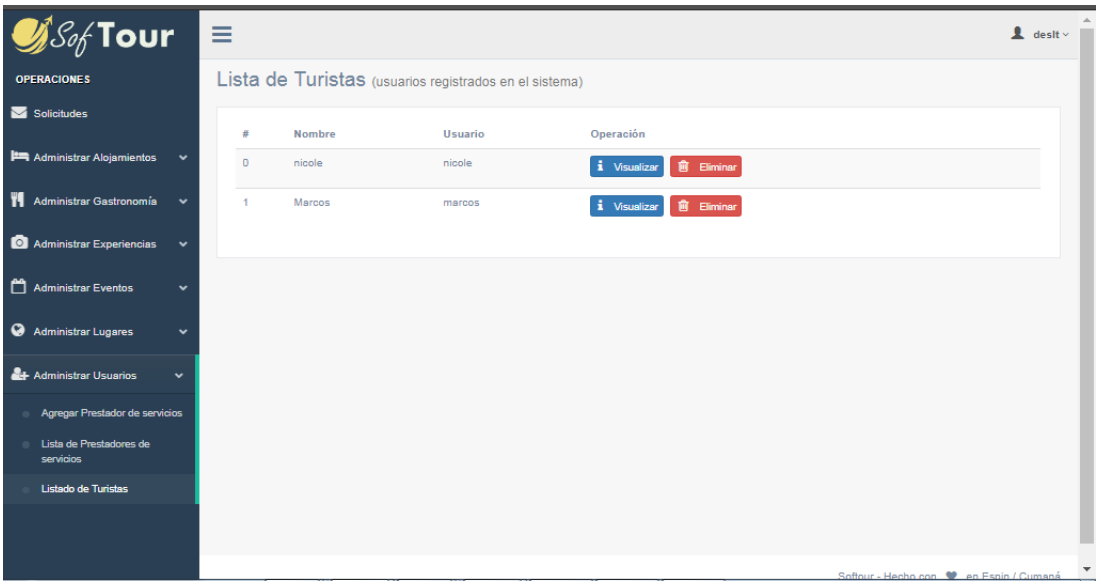


Figura GG. 7. Prueba eliminar turista por administrador

El sistema consulta al usuario si quiere seguir con la acción mediante una interfaz única para esto, como se muestra en la siguiente imagen:

Figura GG. 8. Prueba eliminar turista por administrador

A continuación, se muestra que el Proceso se cumplió exitosamente

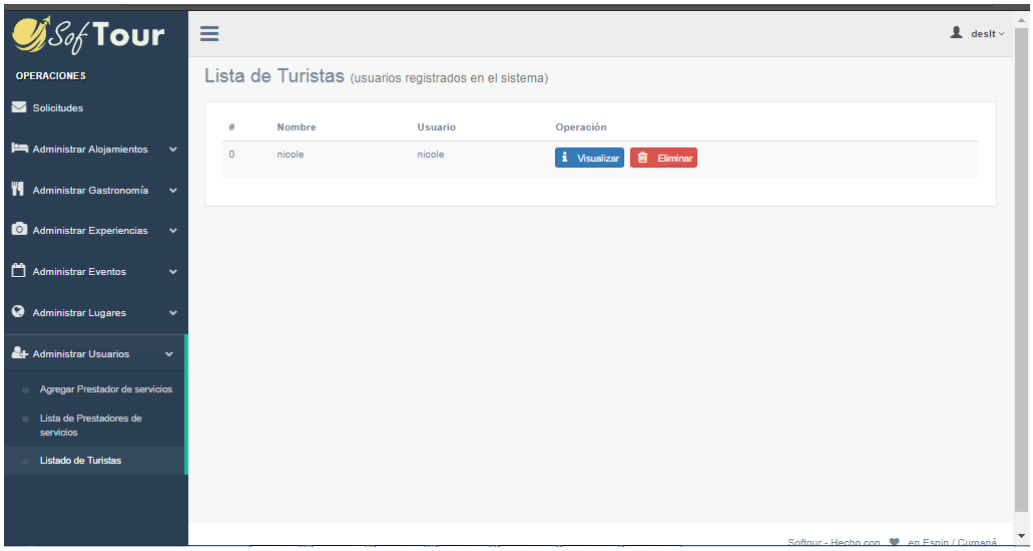


Figura GG. 9. Prueba eliminar turista por administrador

Conclusión de prueba

La función se ejecuta con éxito.

Pruebas de interfaz y usabilidad

De acuerdo a análisis previo se determinó que la interfaz de usuario Administrador se encuentra óptima para interactuar con los usuarios.

Apéndice HH. Ciclo de Versión Prestador de Servicios: Iteración Ingresar Lugar

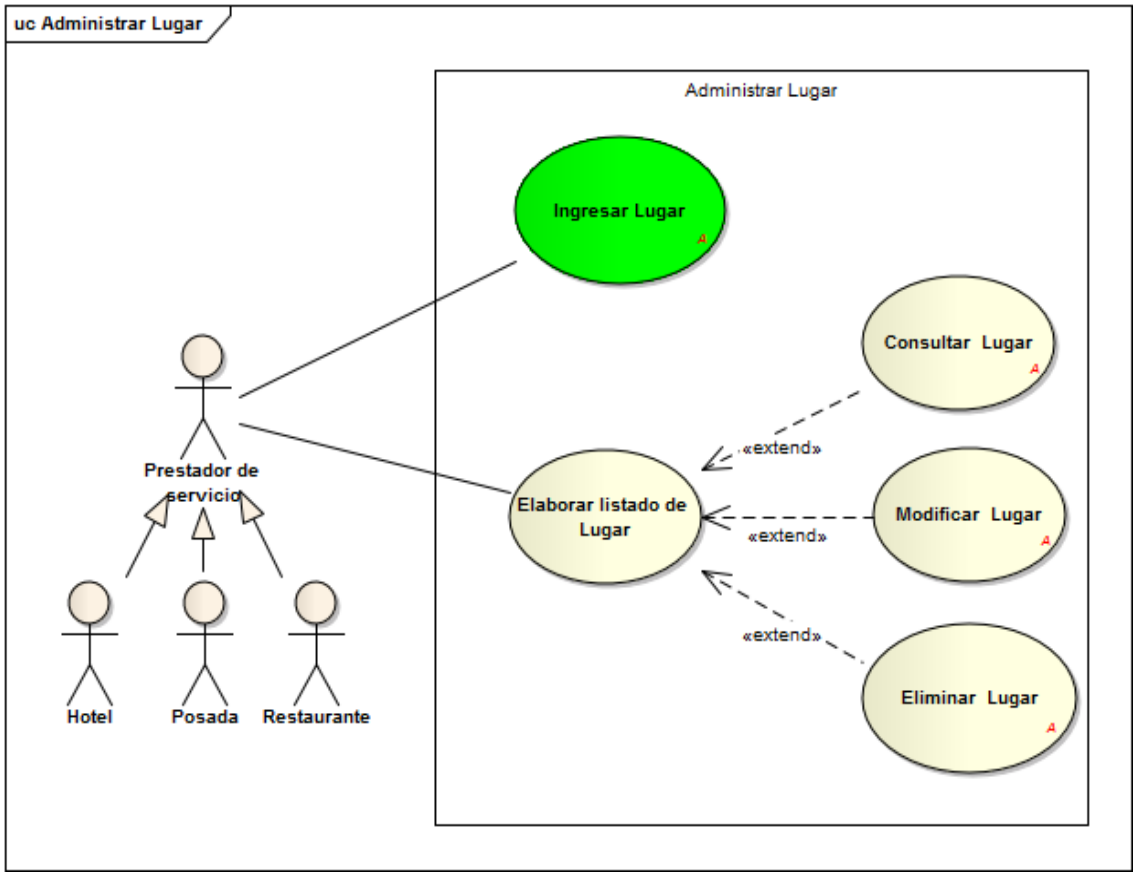


Figura HH. 1. Caso de uso: Ingresar Lugar por prestador de servicio

Tabla HH. 1. Descripción Caso de uso Ingresar Lugar por usuario prestador de servicio

Nombre del Caso de uso	Ingresar lugar.	
Versión	1.0 25/10/2017	
Actor	Prestador de Servicio.	
Descripción del Caso de Uso	El caso de uso consiste en ingresar los atributos del lugar de manera satisfactoria.	
Precondición	Para ingresar un lugar, no debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Prestador selecciona la opción “Agregar”.
	2	El sistema muestra el formulario correspondiente.
	3	El Prestador ingresar los atributos del lugar.
	4	El sistema almacena los atributos del lugar y finaliza el caso de uso.
Postcondición	Se ingresa los atributos del lugar en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Prestador decide no ingresar el lugar y selecciona cancelar el caso de uso, seleccionando otra opción en el sistema.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. Para que un lugar pueda ser ingresado no debe estar registrado en la base de datos del sistema.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

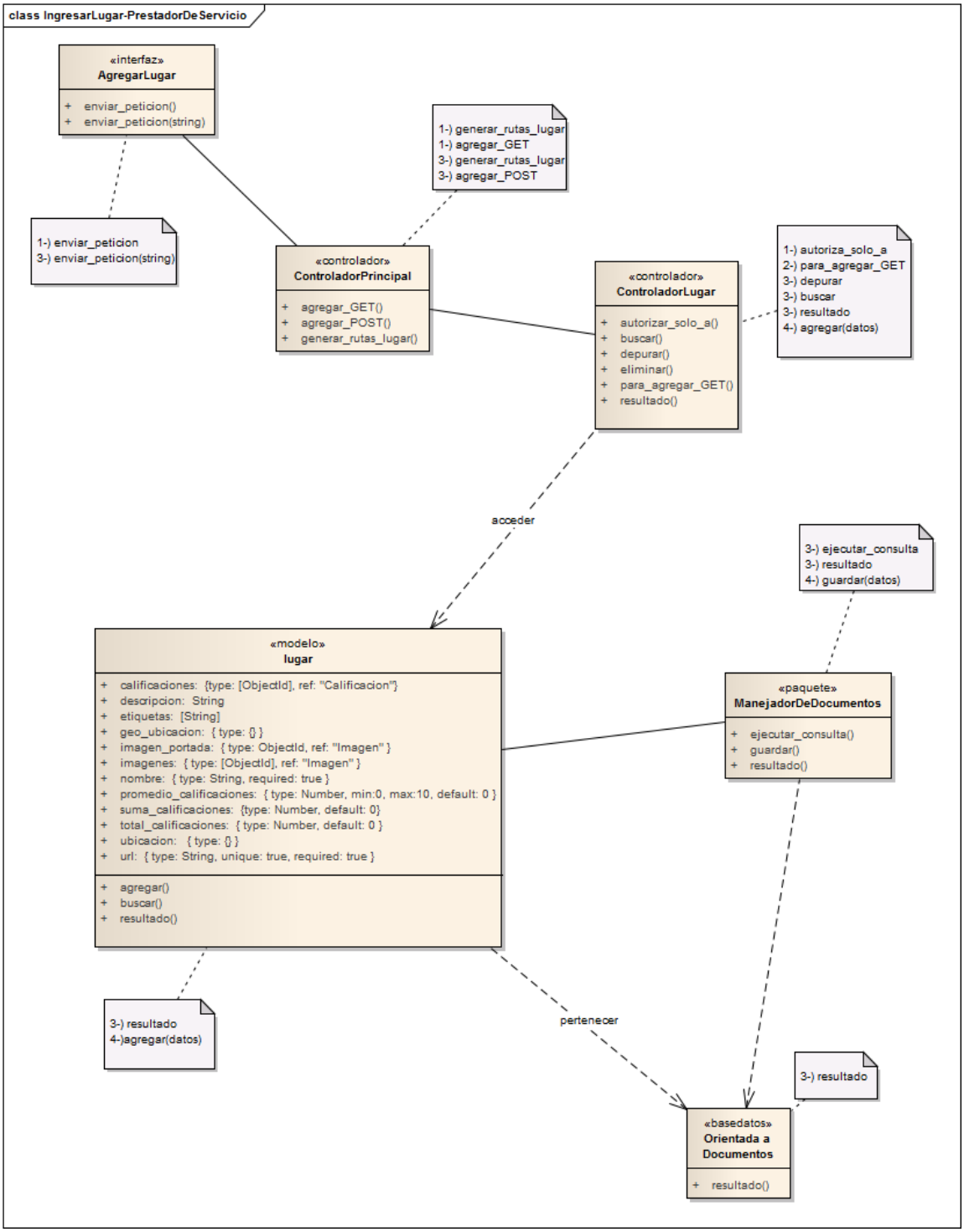


Figura HH. 2. Diagrama de Clases: Ingresar Lugar por usuario Prestador de Servicios

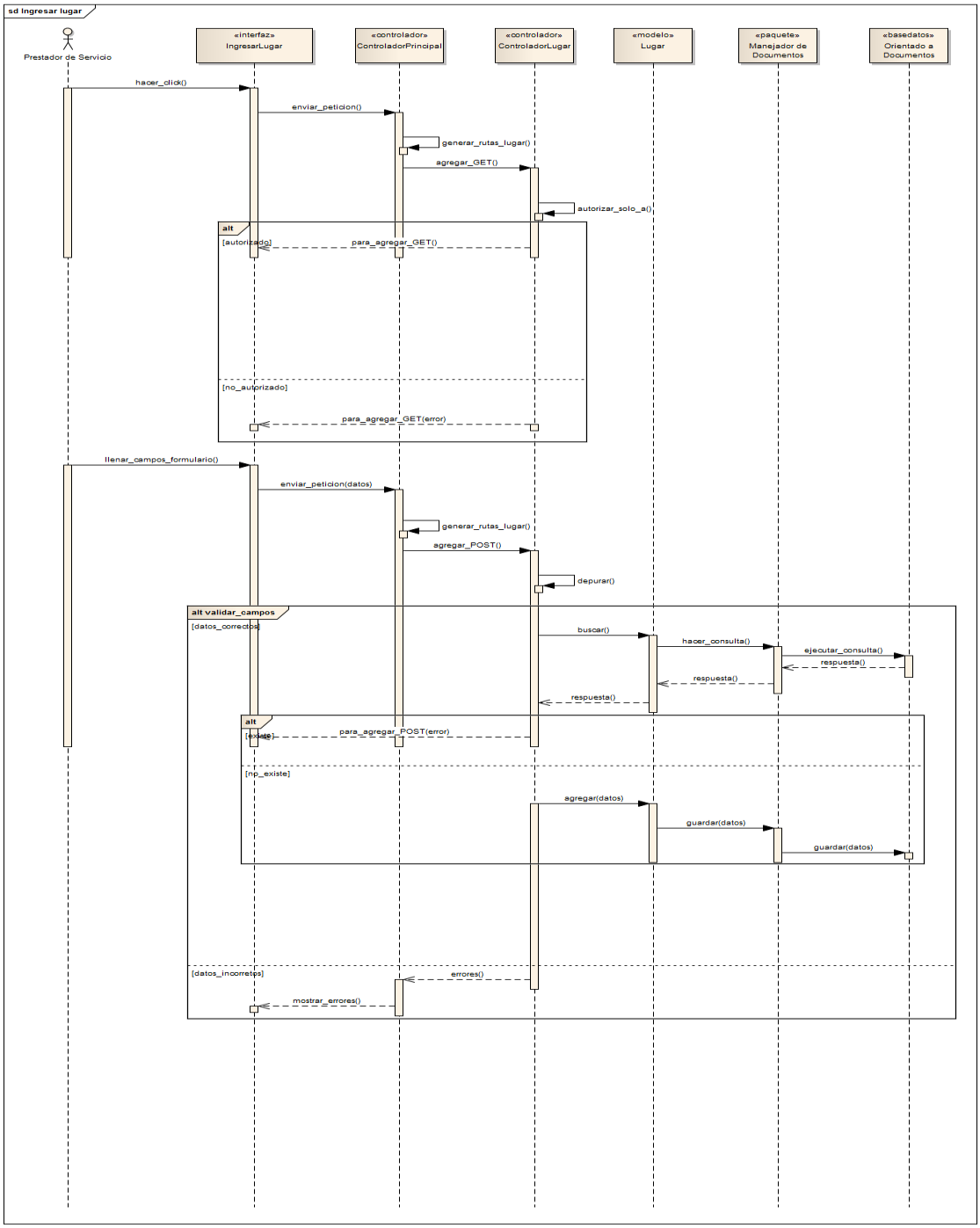


Figura HH. 3.Diagrama de secuencia: Ingresar Lugar por usuario Prestador de Servicio

A continuación, se muestran los modelos de plantillas para el proceso de ingresar lugar.

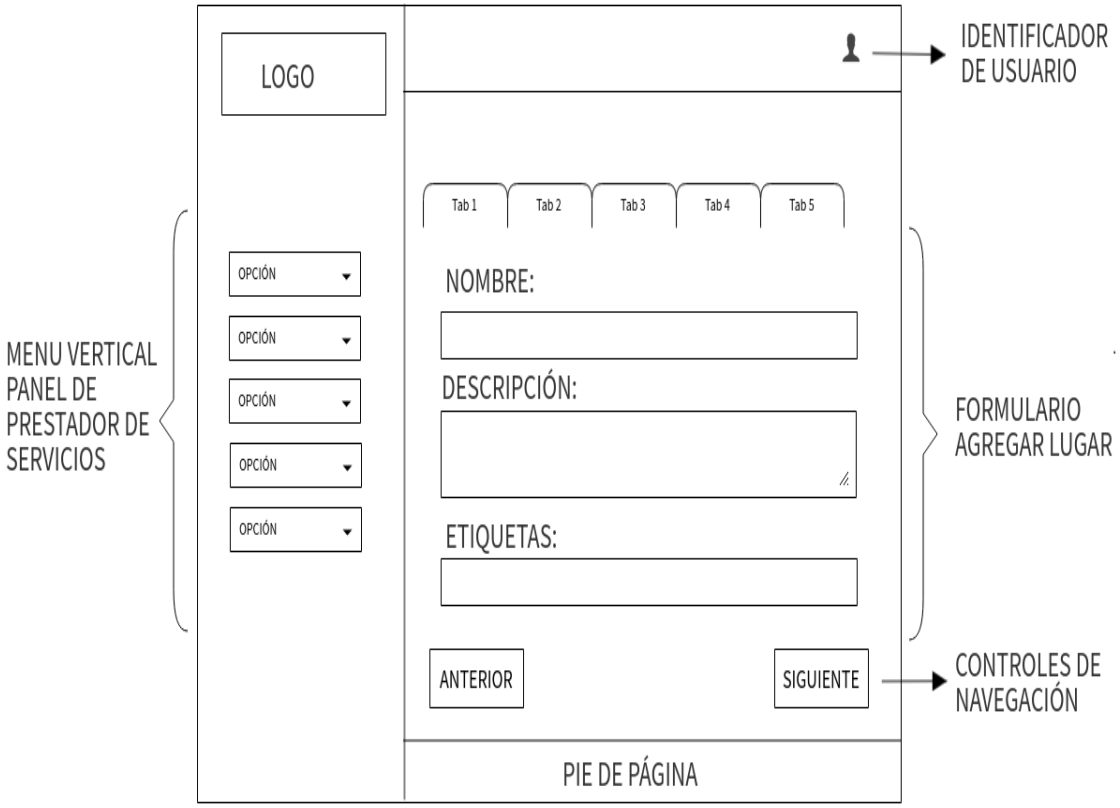


Figura HH. 4.Plantilla de interfaz 1: Ingresar Lugar por usuario prestador de servicio.

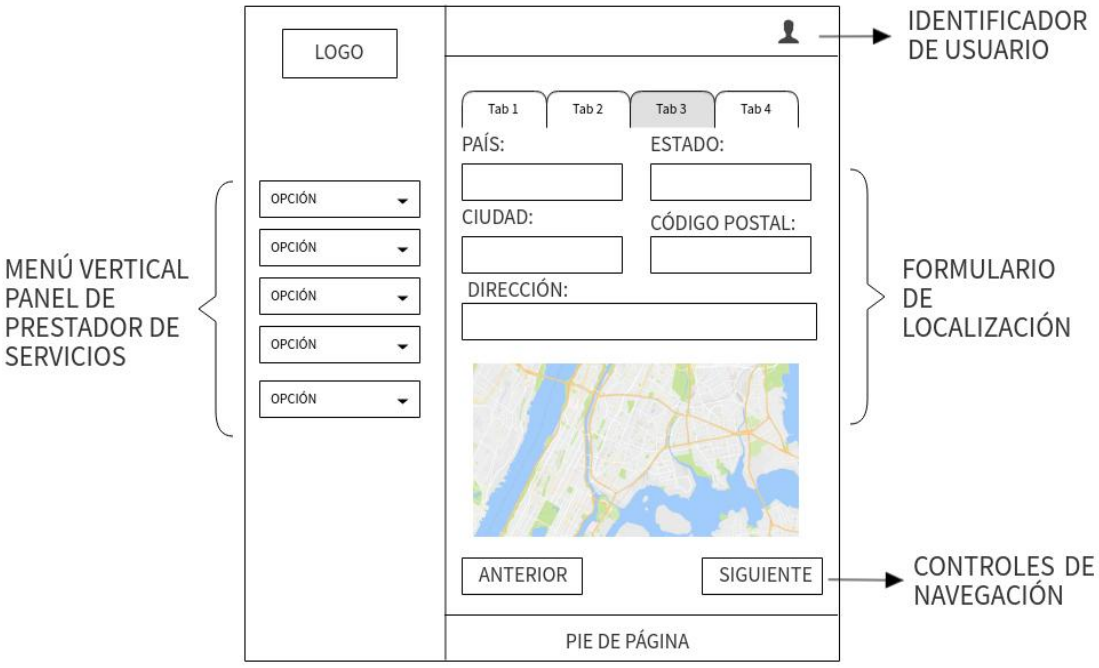


Figura HH. 5. Plantilla de interfaz 2: Ingresar Lugar por usuario prestador de servicio.

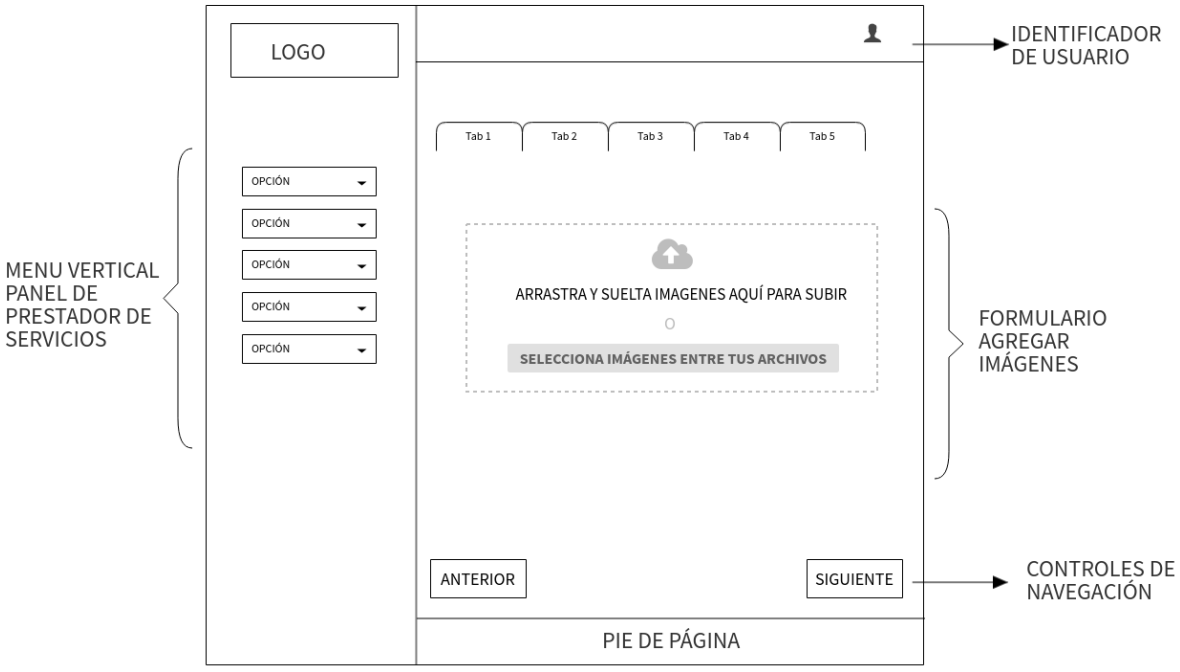


Figura HH. 6. Plantilla de interfaz 3: Ingresar Lugar por usuario prestador de servicio.

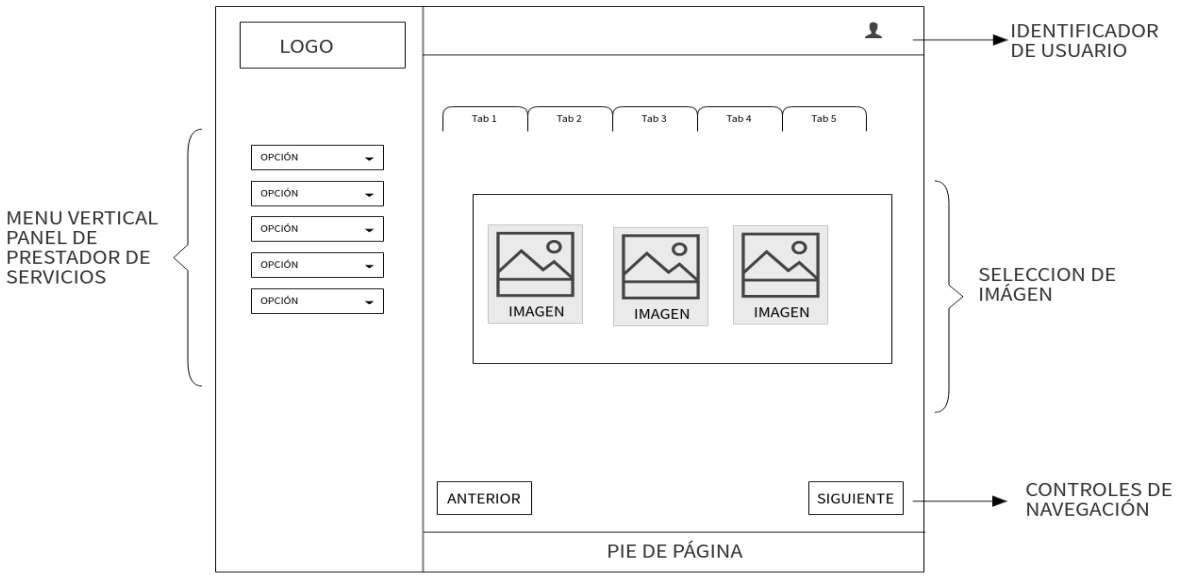


Figura HH. 7. Plantilla de interfaz 4: Ingresar Lugar por usuario prestador de servicio.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Ingresar Lugar y el sistema responde mostrando el formulario correspondiente.

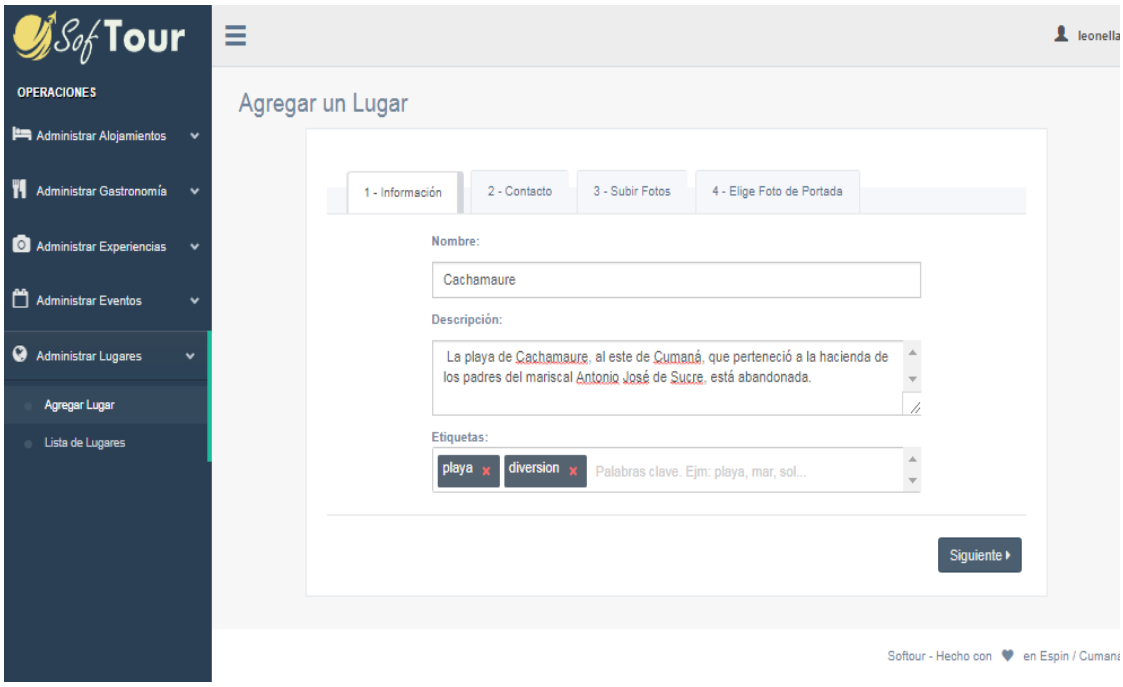


Figura HH. 8. Interfaz 1 de proceso de Ingresar Lugar por usuario Prestador de servicio

El Prestador de Servicio ingresa los atributos del lugar, lo cual se visualiza en diferentes pestañas: Información, Contacto, Subir fotos y Elige foto de portada.

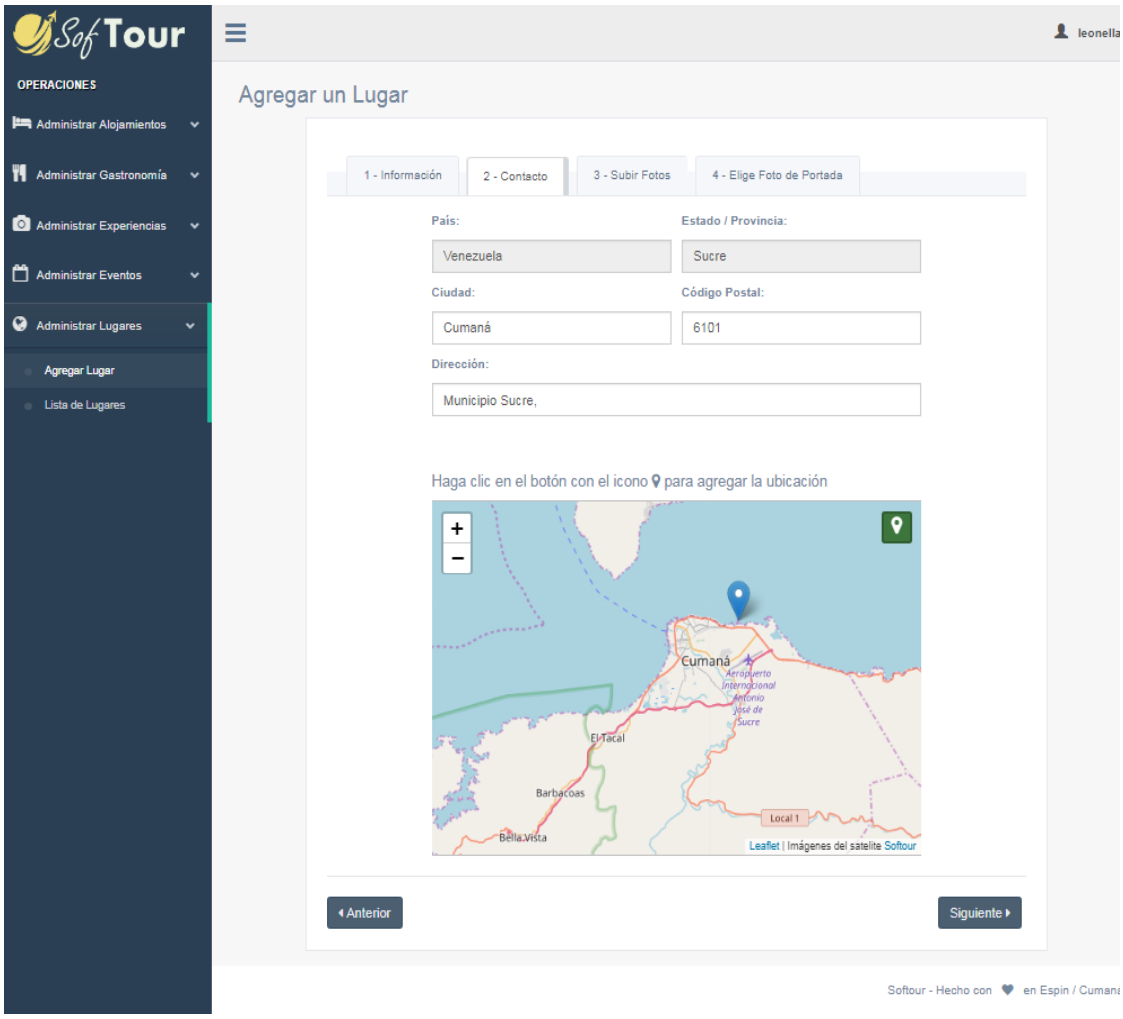


Figura HH. 9. Interfaz 2 de proceso de Ingresar Lugar por usuario Prestador de servicio

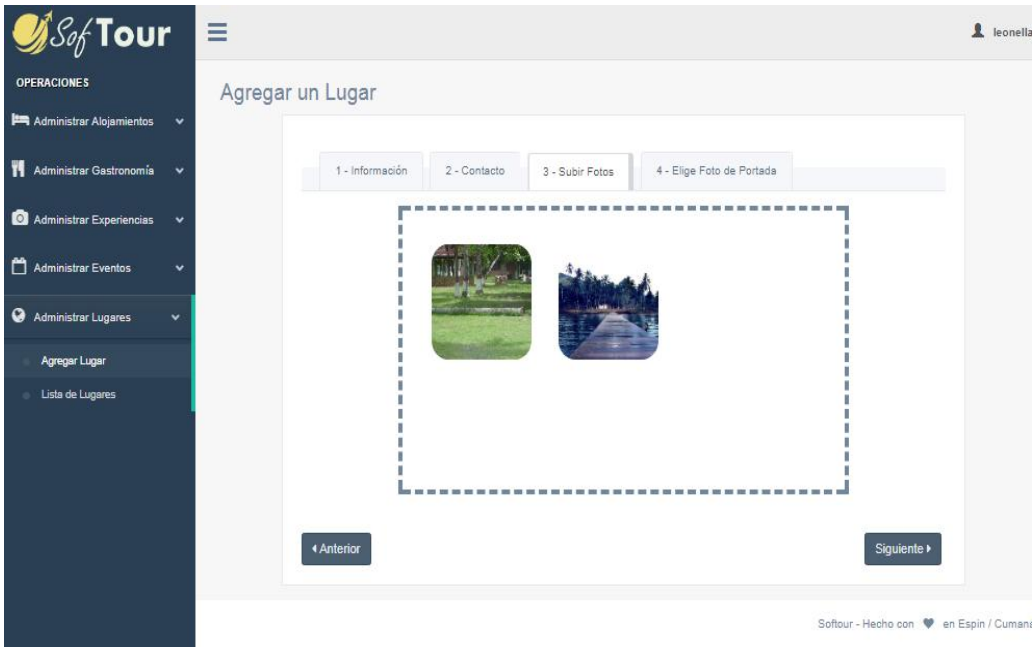


Figura HH. 10 Interfaz 3 de proceso de Ingresar Lugar por usuario Prestador de servicio

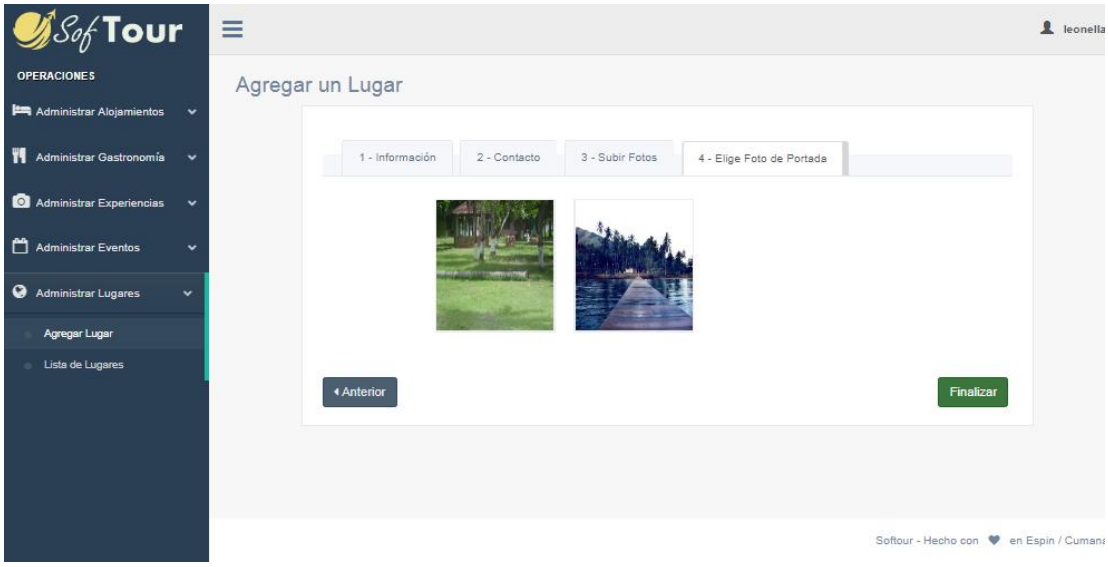


Figura HH. 11. Interfaz 4 de proceso de Ingresar Lugar por usuario Prestador de servicio

Código desarrollado para la funcionalidad Ingresar Lugar por usuario Prestador de Servicio.

```
ingresar_POST(req, res, next) {
    this._validar.campos(req.body, false, (error, resultados) => {
        if(error) next(error);
        res.locals.datos = resultados;

        if (res.locals.datos.error) {
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;

            this._respuesta.para_ingresar_POST(req, res);
        }
        else {
            res.locals.datos.value.geo_ubicacion = {
                "type": "Point",
                "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
            };
            res.locals.datos.value.ubicacion = {
                "direccion": res.locals.datos.value.direccion,
                "pais": res.locals.datos.value.pais,
                "estado": res.locals.datos.value.estado,
                "ciudad": res.locals.datos.value.ciudad,
                "codigo_postal": res.locals.datos.value.codigo_postal
            };

            this._Lugar.buscar({"url": res.locals.datos.value.url}, (error, lugar) => {
                if (lugar) {
                    res.locals.datos.error = { "details": [ {
                        "path": "nombre",
                        "message": 'Ya existe un lugar con este nombre.'
                    } ] };
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_ingresar_POST(req, res);
                }

                else {
                    let lugar = this._fabricar.lugarTipo(res.locals.datos.value.tipo_lugar);
                    if(req.session.passport.user.tipo_usr == "Administrador" && lugar.__t)
                        res.locals.datos.value.aprobado = true;

                    lugar.ingresar(res.locals.datos.value, (error, creado) => {
                        if(error) {
                            res.locals.datos.error = error;

                            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                            this._respuesta.para_ingresar_POST(req, res);
                        }

                        res.locals.datos = {"value": creado};
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_ingresar_POST(req, res);
                    });
                }
            });
        }
    });
}
```

Figura HH. 12. Código desarrollado para el proceso Ingresar Lugar por usuario Prestador de Servicio.

Prueba Ingresar Lugar

EL usuario Prestador de Servicios posee muchos privilegios en el sistema, casi igualando al usuario administrador, quien se ve encargado de validar la información que agregue el Prestador de Servicios.

A continuación, se muestra las pruebas realizadas a la función denominada Agregar Lugar, en este caso ejecutada desde el perfil Prestador de Servicios, se validó su correcto flujo y que el resultado sea satisfactorio, para validad dicha funcionalidad se utilizó la técnica denominada prueba de caja negra.

Se tomaron en cuenta las siguientes clases de equivalencia:

Tabla HH. 2. Clases de equivalencia Ingresar Lugar por Prestador de servicio

Condición de entrada	Tipo	Clases de equivalencias válidas	Clases de equivalencias no válidas
Nombre de lugar	Alfabético	1:Cadena de caracteres.	2: Campo en blanco 3:Símbolos
Descripción	Alfanumérico	4:Cadena de caracteres, números.	2: Campo en blanco 5:Cadena de caracteres menores a 150.
Etiquetas	Alfanumérico	4: Cadena de caracteres, números.	
País	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Estado/Provincia	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Ciudad	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Código postal	Numérico	6: Números	1: Cadena de caracteres 3: Símbolos

			7:Letras
Dirección	Alfanumérico	1:Cadena de caracteres, números.	2: Campo en blanco
Fotos	Imagen	8: JPG, GIF, PNG	2: Campo en blanco
Elegir foto de perfil	Selección		
Condición de entrada	Tipo	Clases de equivalencias válidas	Clases de equivalencias no válidas
Nombre de lugar	Alfabético	1:Cadena de caracteres.	2: Campo en blanco 3:Símbolos
Descripción	Alfanumérico	4:Cadena de caracteres, números.	2: Campo en blanco 5:Cadena de caracteres menores a 150.
Etiquetas	Alfanumérico	4: Cadena de caracteres, números.	
País	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Estado/Provincia	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Ciudad	Alfabético	1: Cadena de caracteres	3: Símbolos 5: Números
Código postal	Numérico	6: Números	1: Cadena de caracteres 3: Símbolos 7:Letras
Dirección	Alfanumérico	1:Cadena de caracteres, números.	2: Campo en blanco
Fotos	Imagen	8: JPG, GIF, PNG	2: Campo en blanco
Elegir foto de perfil	Selección		

Tabla HH. 3. Casos de prueba Ingresar Lugar por Prestador de servicio

Condición de entrada	A1	A2	A3
Nombre de lugar	Parque Ayacucho	¡Museo Humbolt!	Monumento
Descripción	Un parque es un terreno situado en el interior de una población, que se destina a prados, jardines y arbolado sirviendo como lugar de esparcimiento y recreación de los ciudadanos.		En el monumento alegórico a la Fundación de Cumaná el Indio representa los aborígenes que poblaron nuestra región y el Monje la Evangelización de los Frailes Dominicos y Franciscanos quienes oficiaron las primeras misas en tierra firme del continente americano.
Etiquetas	Parque; distracción; diversión	Museo; Teléfono; 123	Paseo; visitas;
País	Venezuela	México	
Estado/Provincia	Sucre		Amazonas
Ciudad	Cumaná		4568
Código postal	6101	5555	nchd
Dirección	Av. Humboldt a orillas del Río Manzanares		
Fotos	PNG,JPG	Doc, PDF	
Elegir foto de perfil	Selección	Selección	
Clases Equivalencia	1,4,6,8	1,2,3	1,2,4,5,7
Resultado	Registro exitoso	No se concretó registro	No se concretó registro

Para el caso de prueba A1 se inicia el proceso de registro de acuerdo a lo establecido en el cuadro anterior.

Figura HH. 13. Inicio caso de prueba A1 Ingresar Lugar por usuario Prestador de Servicio

Continuando con el proceso de pruebas del caso A1, se continua el llenado de datos por lo establecido en los formularios.

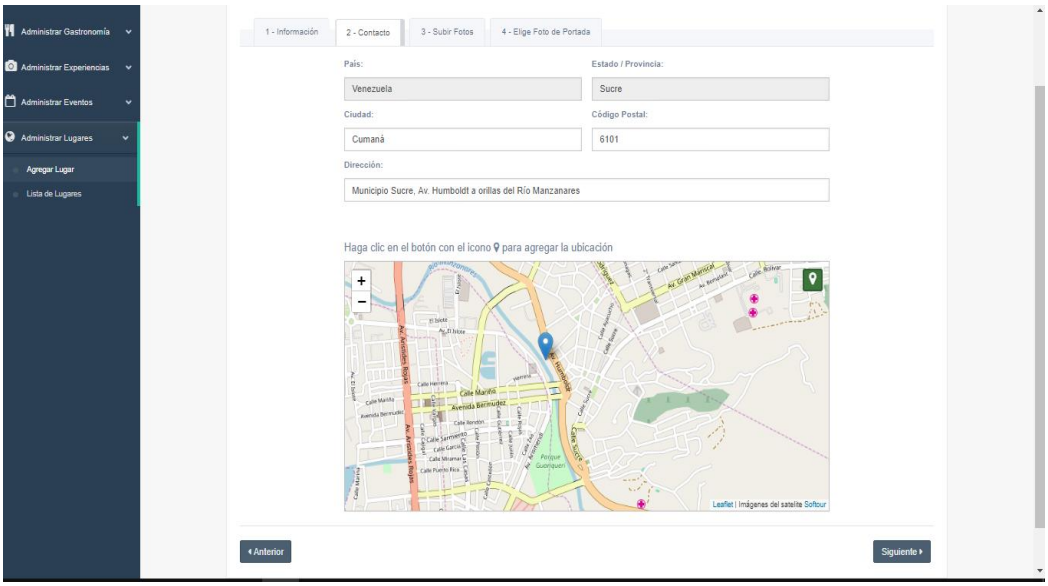


Figura HH. 14. Caso de prueba A1 Ingresar Lugar por usuario Prestador de Servicio

Continuando el llenado de datos según lo establecido para el caso de pruebas A1, se procede a ingresar imágenes del lugar ingresado.

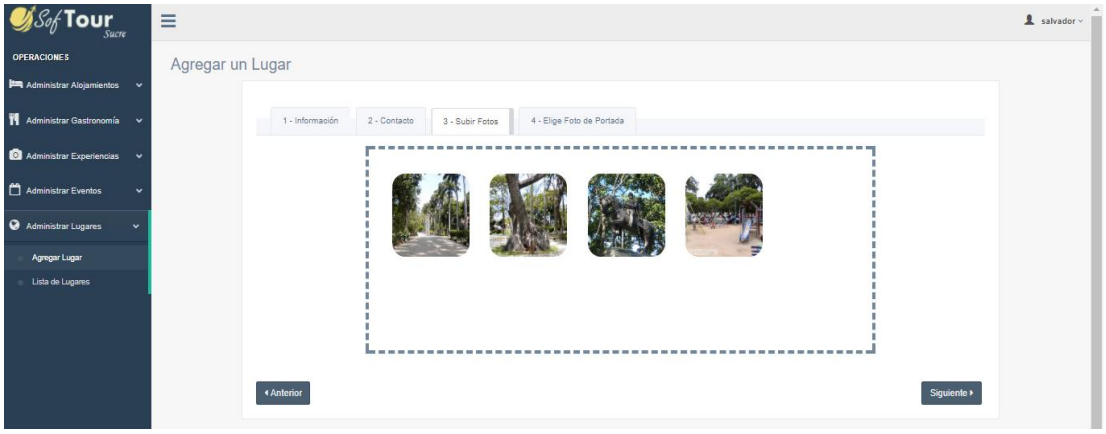


Figura HH. 15.Caso de prueba A1 Ingresar Lugar por usuario Prestador de Servicio

Para concluir el caso de pruebas A1, se procede a seleccionar imagen principal de lugar ingresado y el sistema envía el mensaje de que se ingresaron los datos correctamente.

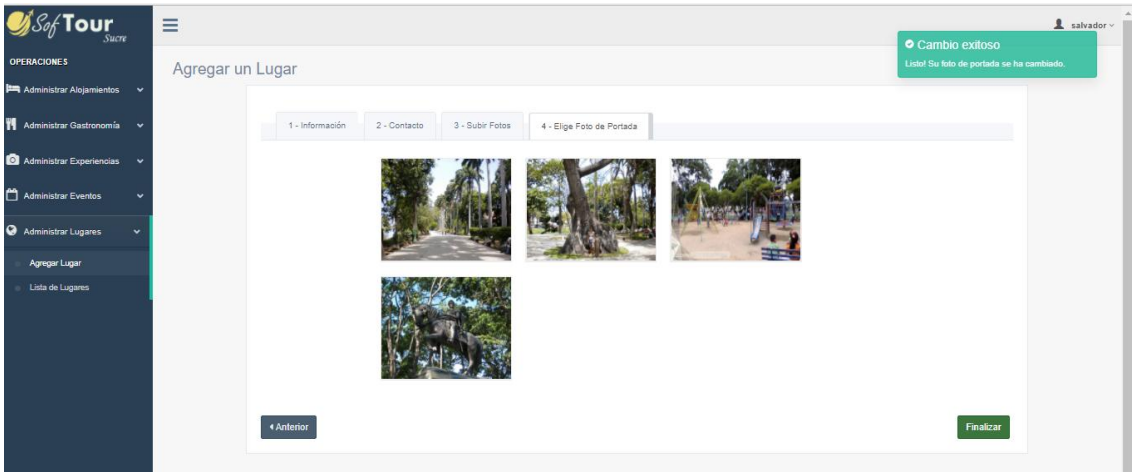


Figura HH. 16.Caso de prueba A1 Ingresar Lugar por usuario Prestador de Servicio

En el caso de pruebas A2 del registro de lugares desde el perfil prestador de servicios, no se logró concretar el proceso de registro ya que el sistema no permite símbolos y signos en el apartado de nombre.

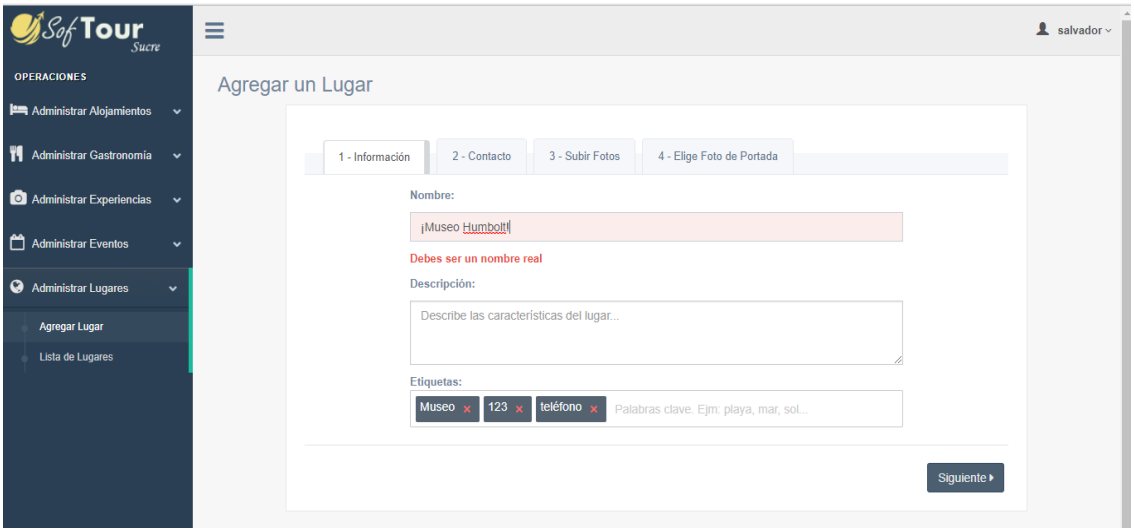


Figura HH. 17.Caso de prueba A2 Ingresar Lugar por usuario Prestador de Servicio

En el caso de pruebas A3 el sistema comprueba como válidos los datos ingresados para el primer apartado del formulario.

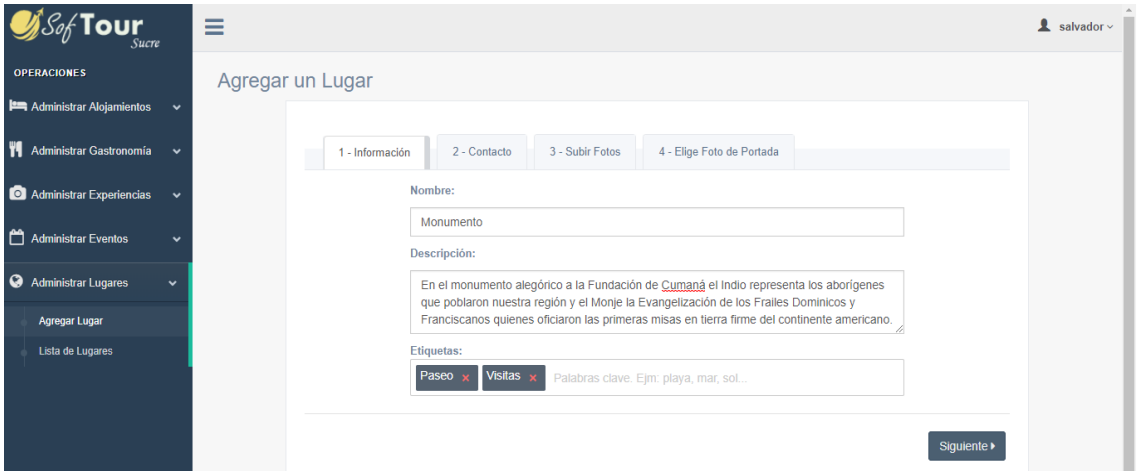


Figura HH. 18.Caso de prueba A3 Ingresar Lugar por usuario Prestador de Servicio

El sistema evita ingresar datos inválidos en el campo de ciudad y código postal,

señalándole al usuario donde se encuentra el error y evitando que prosiga, ya que el sistema esta validado para evitar ingresar información incoherente, como se muestra en la siguiente imagen.

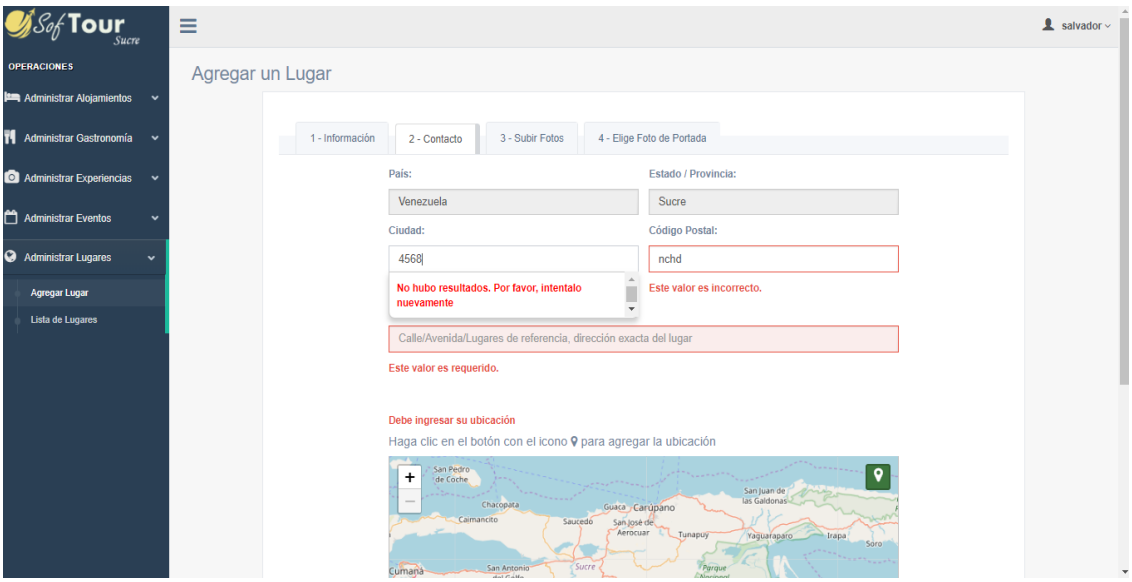


Figura HH. 19.Caso de prueba A3 Ingresar Lugar por usuario Prestador de Servicio

Conclusión de la prueba

Observando los resultados se determinó que la función supero la prueba y cumple su objetivo, el cual es ingresa un nuevo lugar al sistema. Además, se validó que sus campos estuviesen validados para evitar el ingreso de información errónea.

Prueba de interfaz y usabilidad

Uno de los requerimientos que se propuso al plantear el desarrollo del sistema fue trabajar bajos el uso de buenas prácticas de diseño de interfaz e ir de la mano con las heurísticas de usabilidad, por ende, luego de un análisis profundo se determinó que la interfaz cumple con los estándares para el beneficio del usuario prestador de servicio.

Apéndice II. Ciclo de Versión Prestador de Servicios: Iteración Modificar Experiencia.

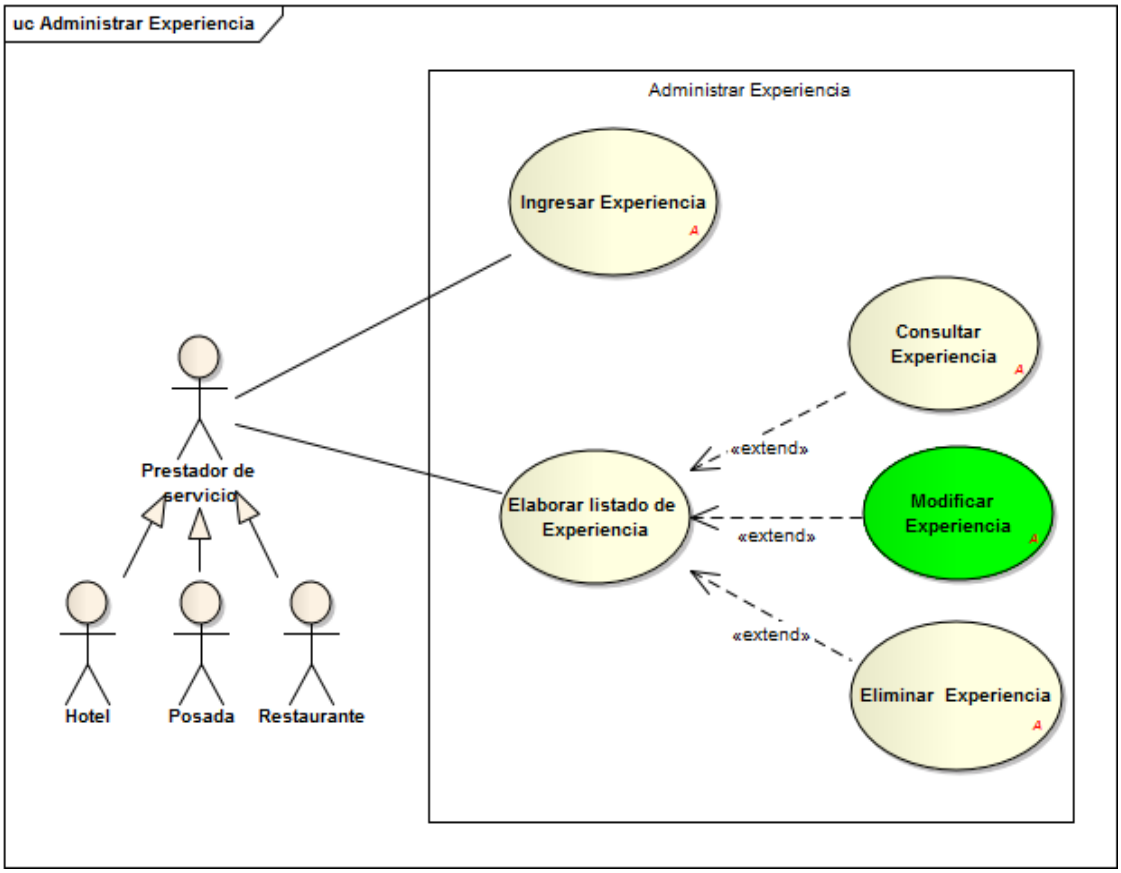


Figura II. 1. Caso de uso: Modificar Experiencia por prestador de servicio.

Tabla II. 1. Descripción Caso de uso Modificar Experiencia por usuario prestador de servicio.

Nombre del Caso de uso	Modificar Experiencia	
Versión	1.0 25/10/2017	
Actor	Prestador de Servicios.	
Descripción del Casos de Uso	El caso de uso consiste en modificar los atributos de la Experiencia de manera satisfactoria.	
Precondición	Para modificar una Experiencia, la misma debe estar registrada en el sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Prestador de Servicios selecciona la opción “Modificar Experiencia”.
	2	El sistema, muestra el formulario correspondiente para indicar la Experiencia a Modificar.
	3	El Prestador de Servicios ingresa el patrón de búsqueda de la Experiencia a modificar.
	4	El sistema ejecuta la búsqueda de la Experiencia.
	5	El Prestador de Servicios modifica los atributos de la Experiencia, selecciona aceptar.
	6	El sistema realiza las modificaciones en la BD y finaliza el caso de uso.
Postcondición	Se modifican los atributos de la Experiencia de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Prestador de Servicios decide no modificar la Experiencia y selecciona cancelar el caso de uso, seleccionando otra opción en el sistema.
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. La experiencia debe estar registrada para que pueda ser modificada.	1
Inclusiones	Ninguna.	
Extensiones	Ninguna	

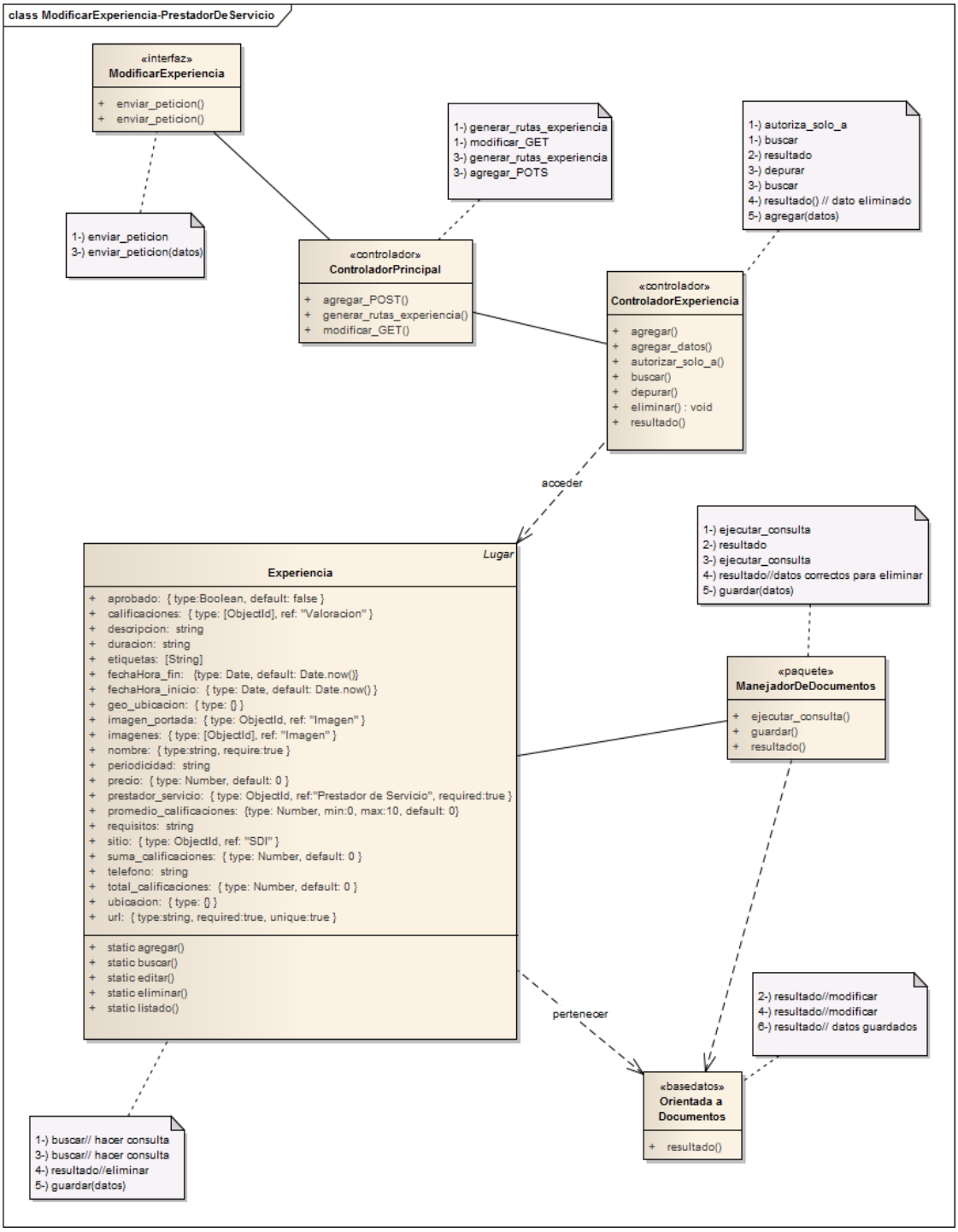


Figura II. 2.. Diagrama de Clases: Modificar Experiencia por usuario Prestador de Servicios

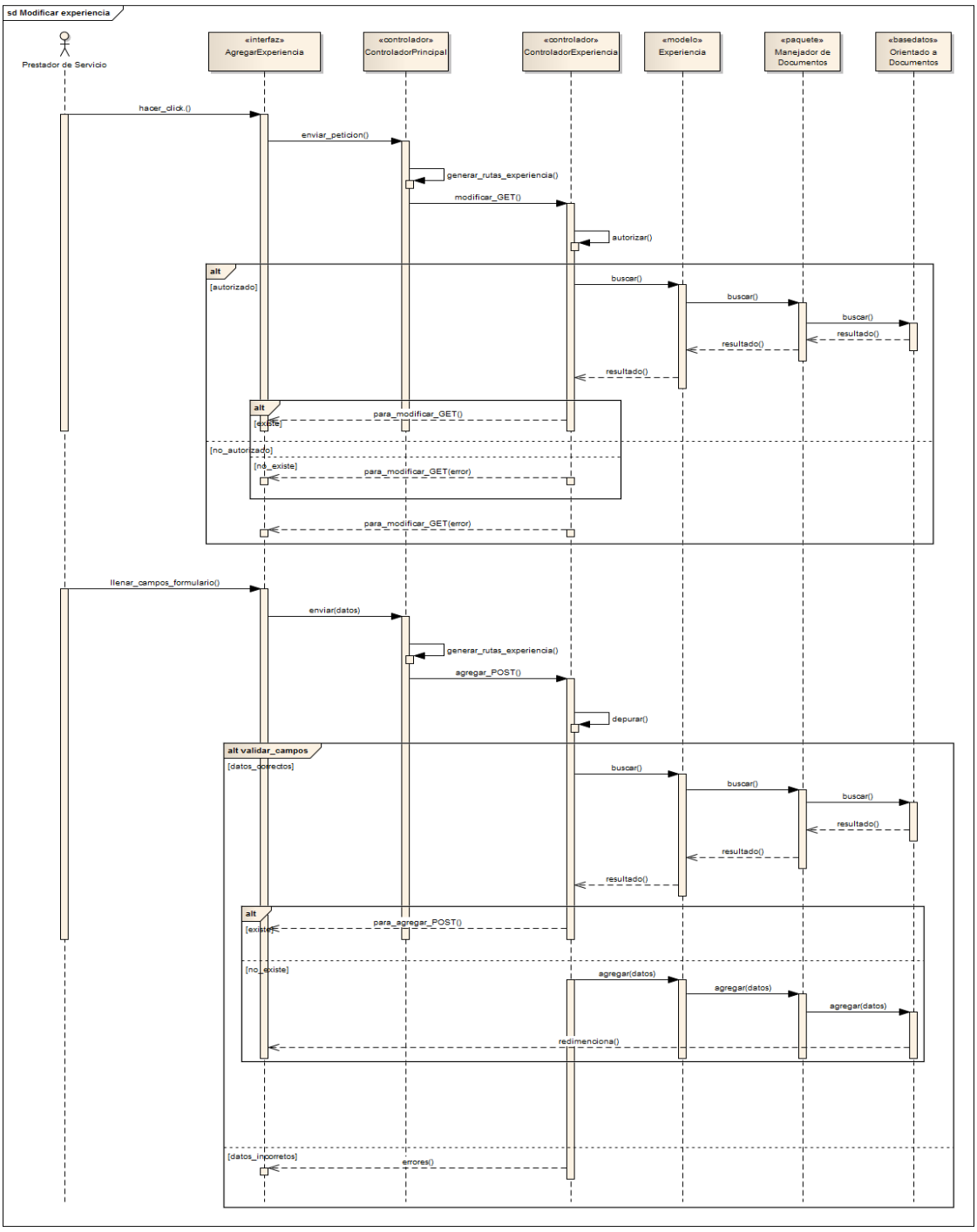


Figura II. 3. Diagrama de secuencia: Modificar Experiencia por usuario Prestador de Servicio

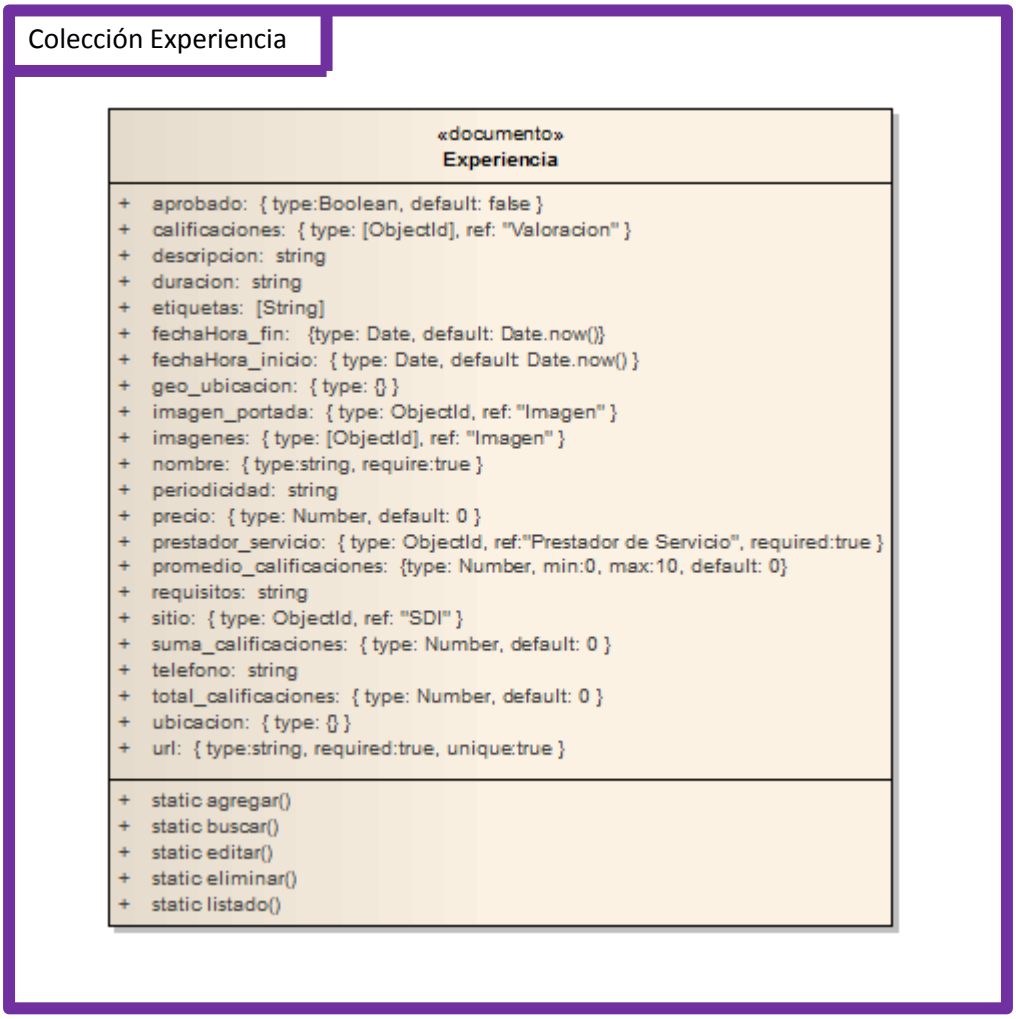


Figura II. 4. Colección de Experiencia: Modificar experiencia por usuario Prestador de Servicio

A continuación, se muestran los modelos de plantillas para el proceso de modificar experiencia.

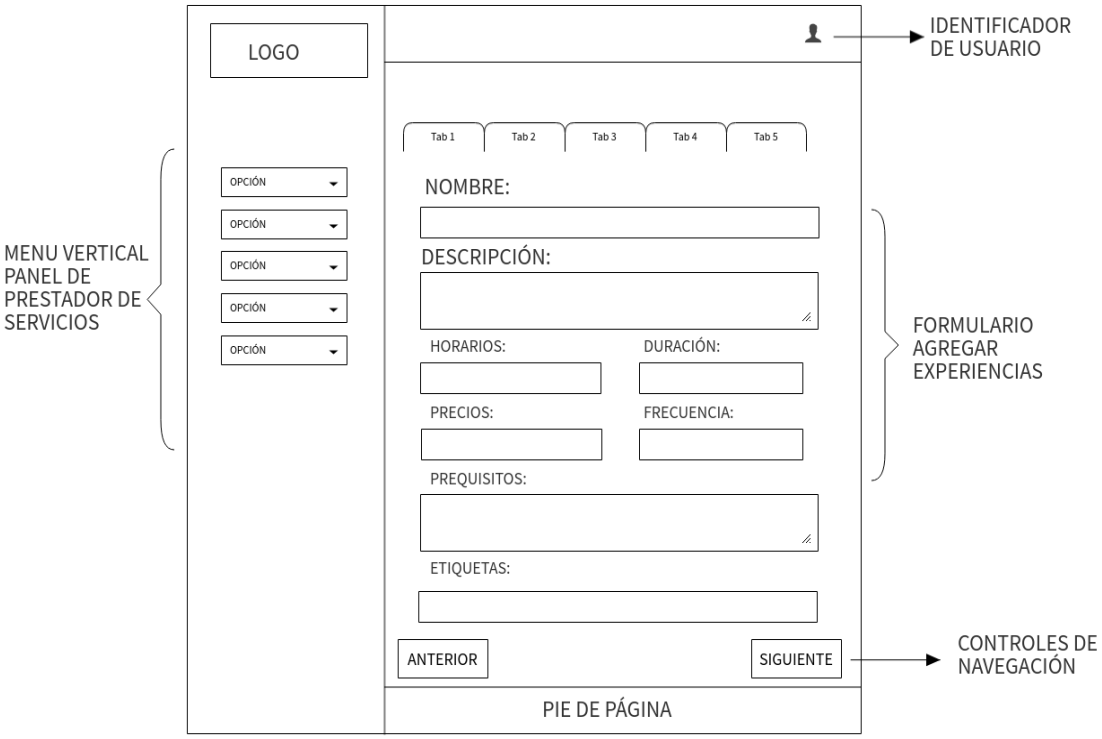


Figura II. 5. Plantilla de interfaz: Modificar Experiencia por usuario prestador de servicio.

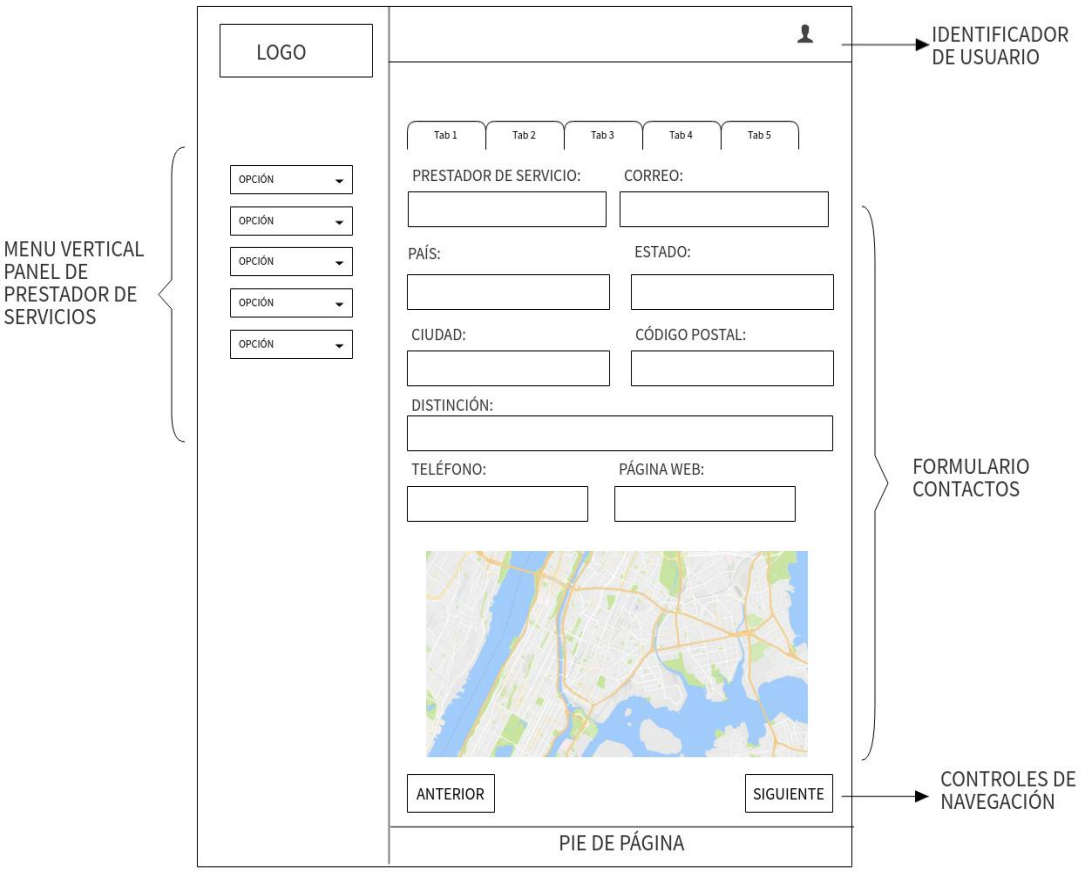


Figura II. 6. Plantilla de interfaz: Modificar Experiencia por usuario prestador de servicio.

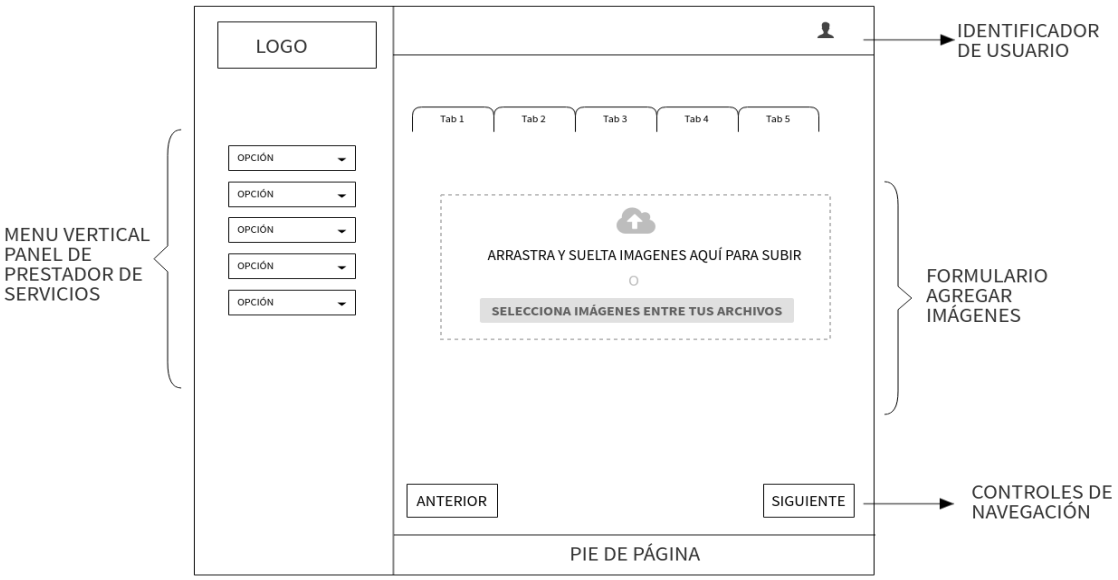


Figura II. 7. Plantilla de interfaz: Modificar Experiencia por usuario prestador de servicio.

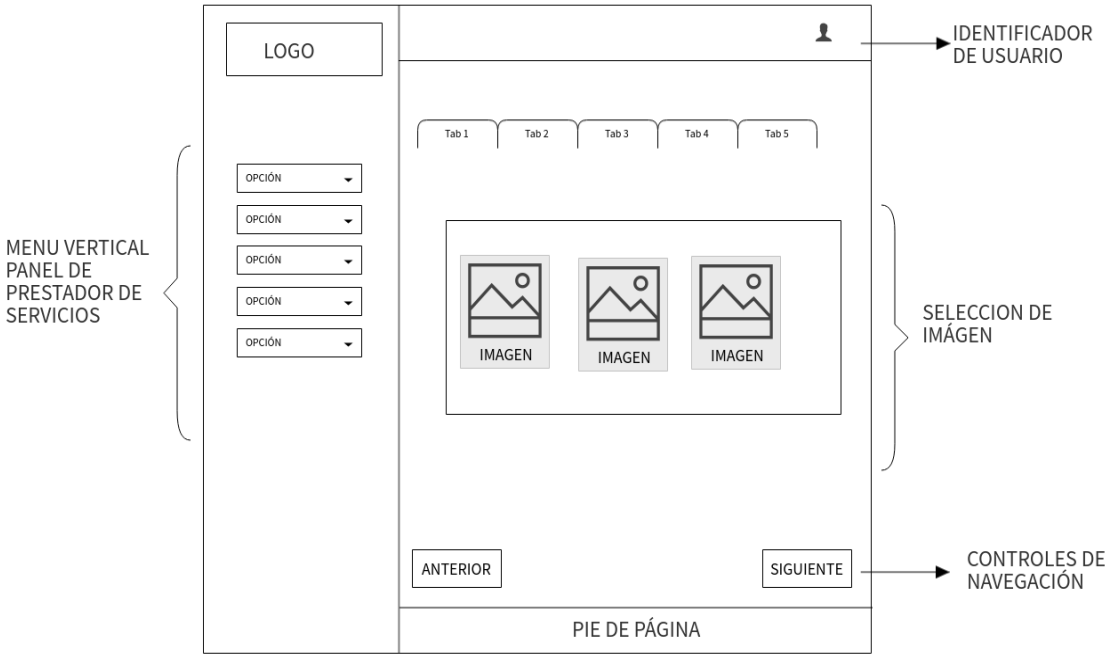


Figura II. 8. Plantilla de interfaz: Modificar Experiencia por usuario prestador de servicio.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Modificar Experiencia y el sistema responde mostrando el formulario correspondiente.

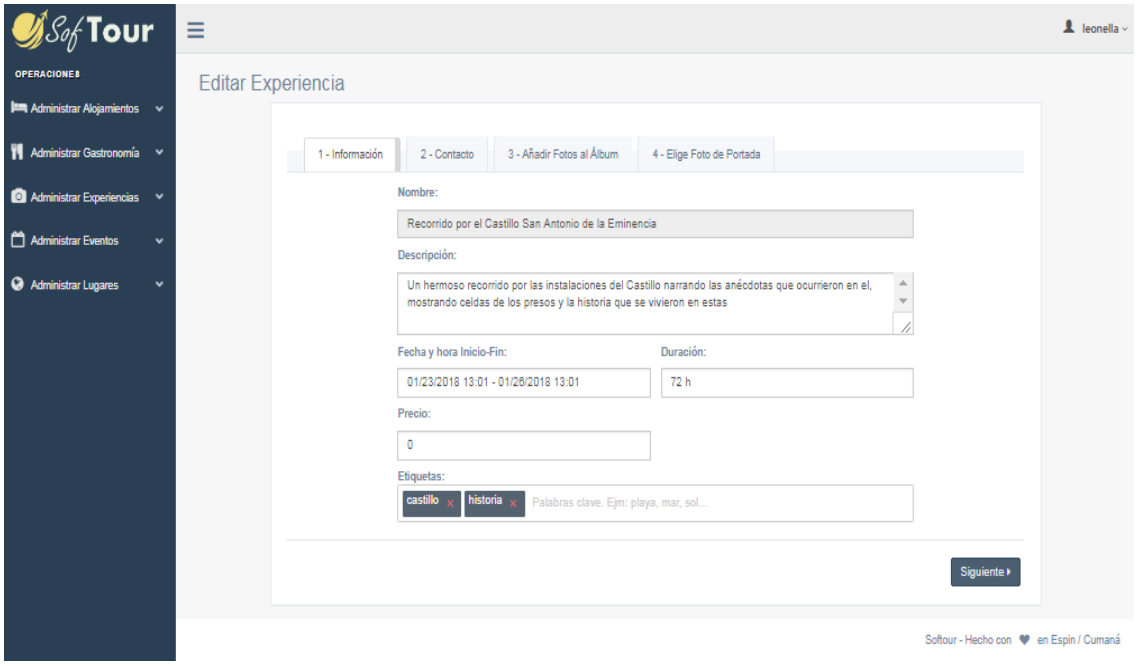


Figura II. 9.Interfaz 1 de proceso de Modificar Experiencia por usuario Prestador de servicio

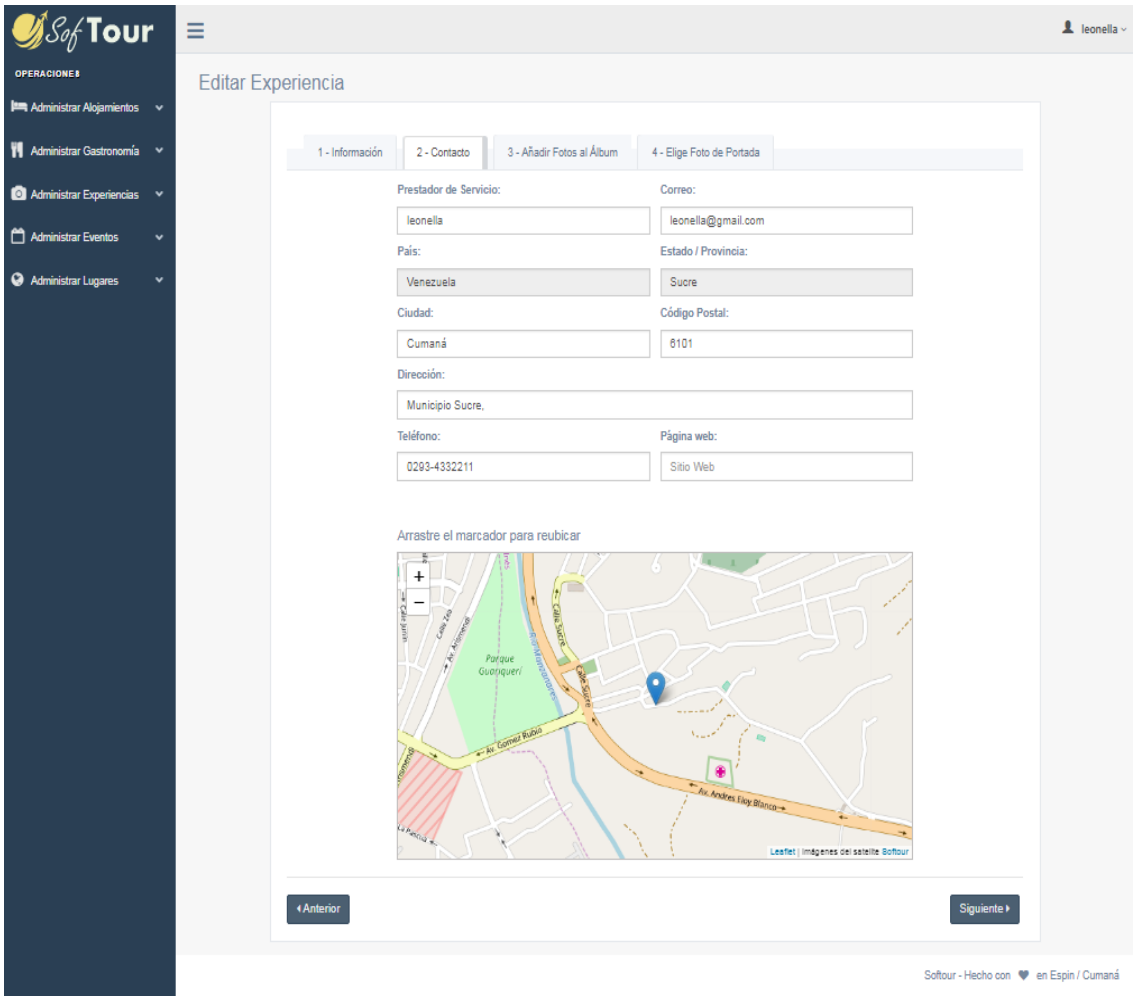


Figura II. 10. Interfaz 2 de proceso de Modificar Experiencia por usuario Prestador de servicio

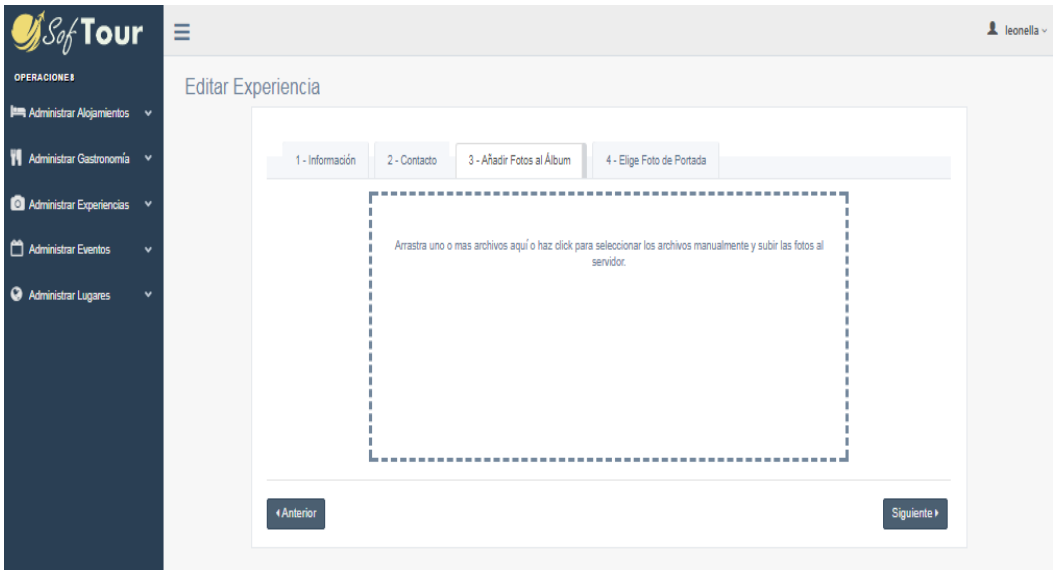


Figura II. 11. Interfaz 3 de proceso de Modificar Experiencia por usuario Prestador de servicio

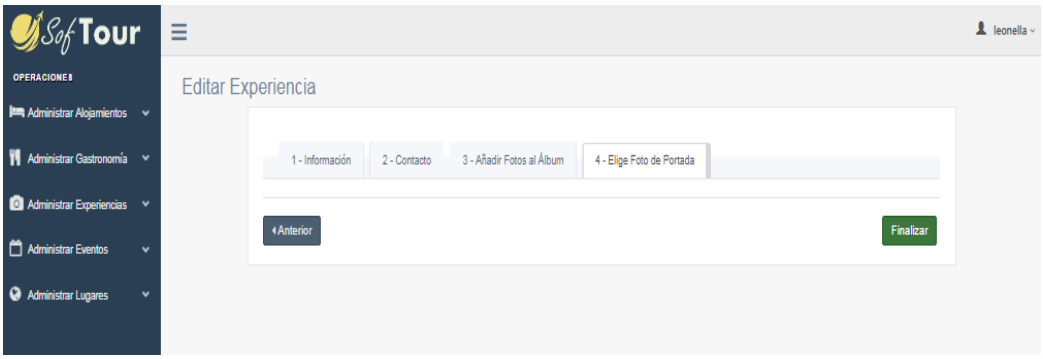


Figura II. 12 Interfaz 4 de proceso de Modificar Experiencia por usuario Prestador de servicio

Código desarrollado para la funcionalidad Modificar Alojamiento por usuario Prestador de Servicio.

```
modificar(req, res, next) {
    this._validar.campos(req.body, true, (error, resultados) => {
        if(error) return next(error);
        res.locals.datos = resultados;

        if(res.locals.datos.error){
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_modificar(req, res);
        }

        else {
            this._Experiencia.buscar({"url": req.params.url.toLowerCase()}, (error, buscado) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
                    this._respuesta.para_modificar(req, res);
                }

                if(!buscado) return next(new this._error_NoEncontrado());
                res.locals.datos.value.geo_ubicacion = {
                    "type": "Point",
                    "coordinates": [res.locals.datos.value.lng, res.locals.datos.value.lat]
                };
                res.locals.datos.value.ubicacion = {
                    "direccion": res.locals.datos.value.direccion,
                    "pais": res.locals.datos.value.pais,
                    "estado": res.locals.datos.value.estado,
                    "ciudad": res.locals.datos.value.ciudad,
                    "codigo_postal": res.locals.datos.value.codigo_postal
                };

                res.locals.datos.value.url = req.params.url.toLowerCase();
                this._Experiencia.modificar(res.locals.datos.value, (error, modificado) => {
                    if(error) {
                        res.locals.datos.error = error;
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                        this._respuesta.para_modificar(req, res);
                    }
                    if(!modificado) return next(new this._error_NoEncontrado());

                    res.locals.datos = {"value": modificado};
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_modificar(req, res);
                });
            });
        }
    });
}
```

Figura II. 13. Código desarrollado para el proceso Modificar Experiencia por usuario Prestador de Servicio.

Pruebas Modificar Experiencia

En ciertos casos los usuarios necesitan modificar la información ingresada al sistema, por tal motivo, se le ofrece al usuario prestador de servicio modificar las experiencias agregadas con anterioridad.

Para realizarle pruebas a esta función fue necesario ingresar experiencias, lo cual se realizó en pruebas anteriores y se utilizaron para esta, en este caso se modificó la siguiente información de la experiencia “bucea con tiburones”:

Información: El océano y los tiburones forman parte del territorio de los animales salvajes, imprevisibles. Aunque estas recomendaciones te puedan ser útiles y aplicables en la mayoría de los casos, no pueden constituir garantías absolutas.

En esta prueba se seleccionó una experiencia ingresada con antelación y se procedió a editar su información.

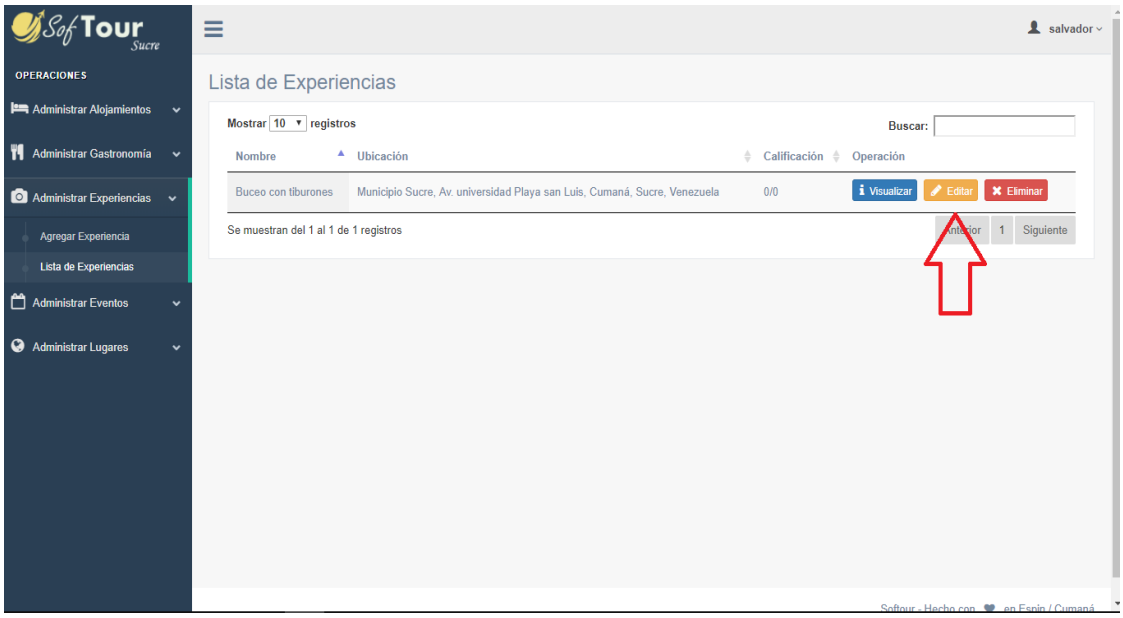


Figura II. 14.Prueba modificar experiencia por prestador de servicio

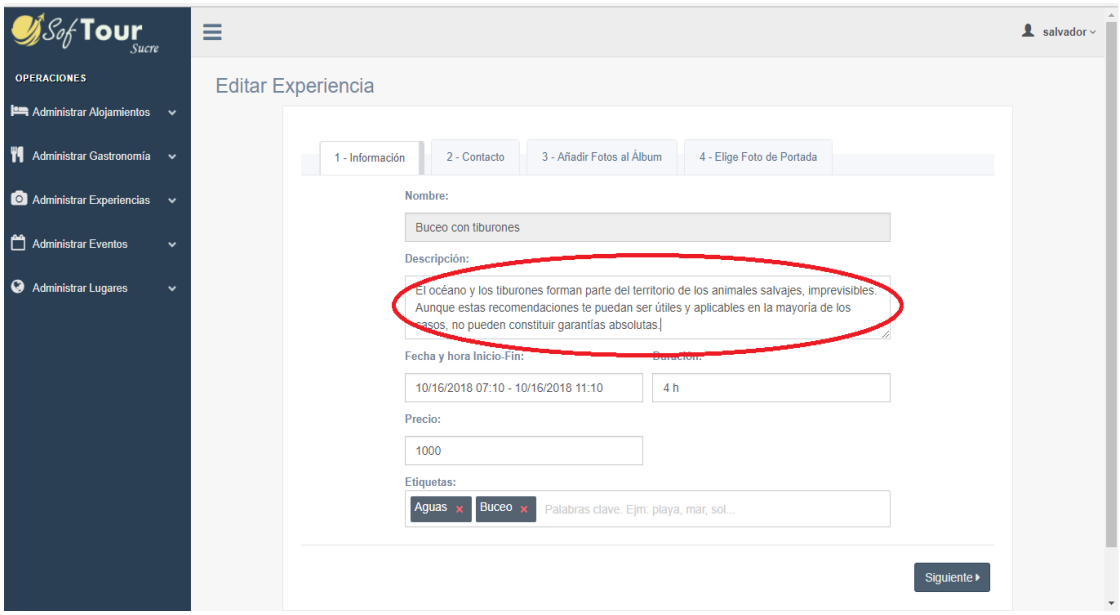


Figura II. 15.Prueba Modificar información por prestador de servicio

En la siguiente imagen se finaliza el proceso de edición guardando los cambios, evidenciando que la función cumple con su objetivo.

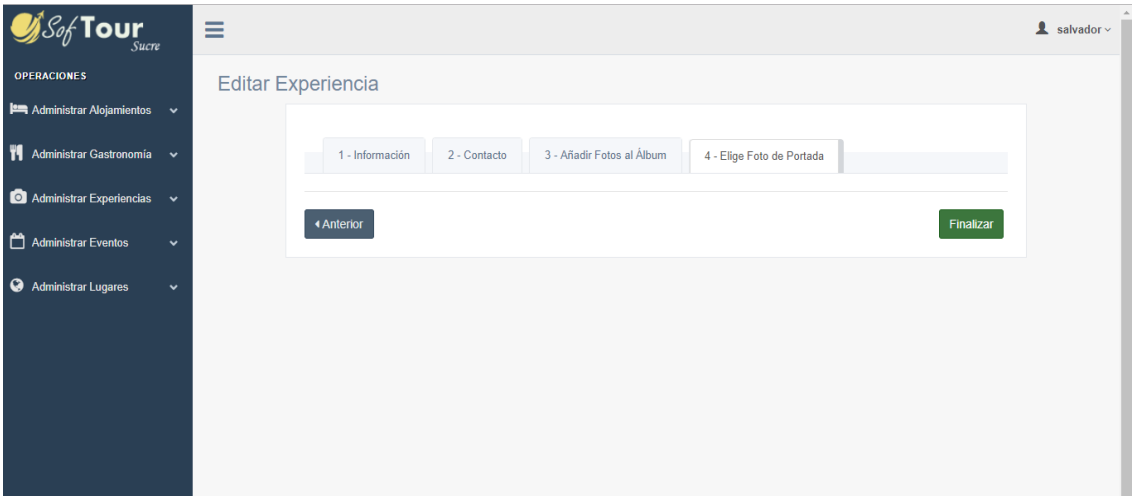


Figura II. 16.Prueba modificar experiencia por prestador de servicio

Conclusión de la prueba

Mediante la prueba se determinó que la función se ejecuta de forma correcta.

Prueba de interfaz y usabilidad

Esta interfaz pertenece al conjunto de interfaces de administrador evaluadas con antelación y donde se determinó que cuenta con las condiciones para ser usadas por el público.

Apéndice JJ. Ciclo de Versión Prestador de Servicios: Iteración Listar Gastronomía

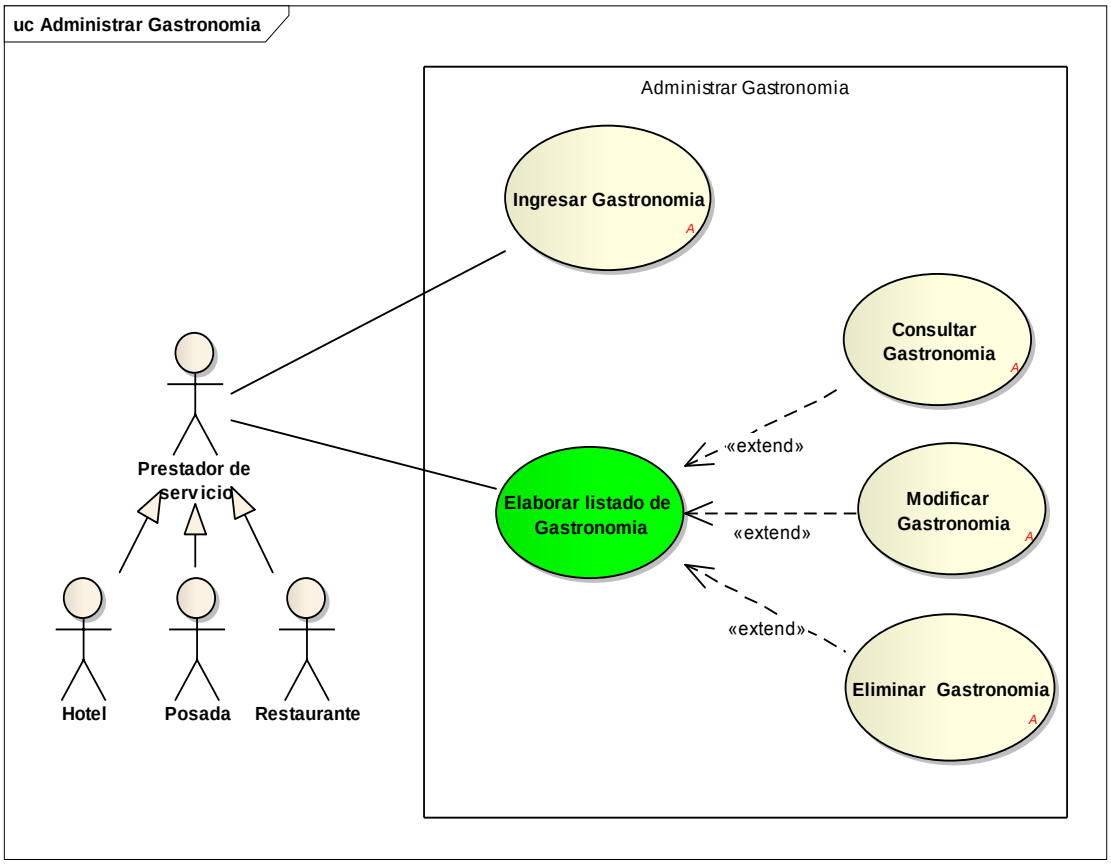


Figura JJ. 1.Caso de uso: Elaborar listado de gastronomía por prestador de servicio

Tabla JJ. 1. Descripción Caso de uso Elaborar listado de gastronomía por usuario prestador de servicio

Nombre del Caso de uso	Elaborar listado de gastronomía.	
Versión	1.0 25/10/2017	
Actor	Prestador de servicio.	
Descripción del Caso de Uso	El caso de uso consiste en elaborar un listado de las gastronomías registradas en la de base de datos del sistema, de manera satisfactoria.	
Precondición	Para elaborar el listado de las gastronomías, deben estar registradas en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Prestador de servicio selecciona la opción "Elaborar listado de gastronomías".
	2	El sistema muestra el listado de las gastronomías almacenados en la base de datos del sistema, de acuerdo a un orden predeterminado, brindando las siguientes opciones de extensión: d. "Consultar gastronomía". e. "Modificar gastronomía" f. "Eliminar gastronomía"
	3	El Prestador de servicio visualiza el listado de las gastronomías y selecciona alguna de las opciones de extensión propuestas.
	4	El sistema extiende <<extend>> hacia el caso de uso que representa la opción seleccionada.
	5	El Prestador de servicio selecciona la opción cerrar.
	6	El sistema finaliza el caso de uso.
Postcondición	Se elabora el listado de gastronomías existentes en la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Prestador de servicio decide no seleccionar ninguna opción de extensión y selecciona cancelar el caso de uso, ya sea seleccionando el botón cerrar o el botón cancelar.
	4.1.	El sistema finaliza el caso de uso.
Reglas del negocio	3. El listado de gastronomías debe ser elaborado de acuerdo a un orden preestablecido. 4. Se podrá seleccionar, desde el listado general de gastronomías, una gastronomía en específica para realizar las operaciones de consultar, modificar o eliminar.	
Inclusiones	Ninguna.	
Extensiones	4. "Consultar gastronomía". 5. "Modificar gastronomía". 6. "Eliminar gastronomía".	

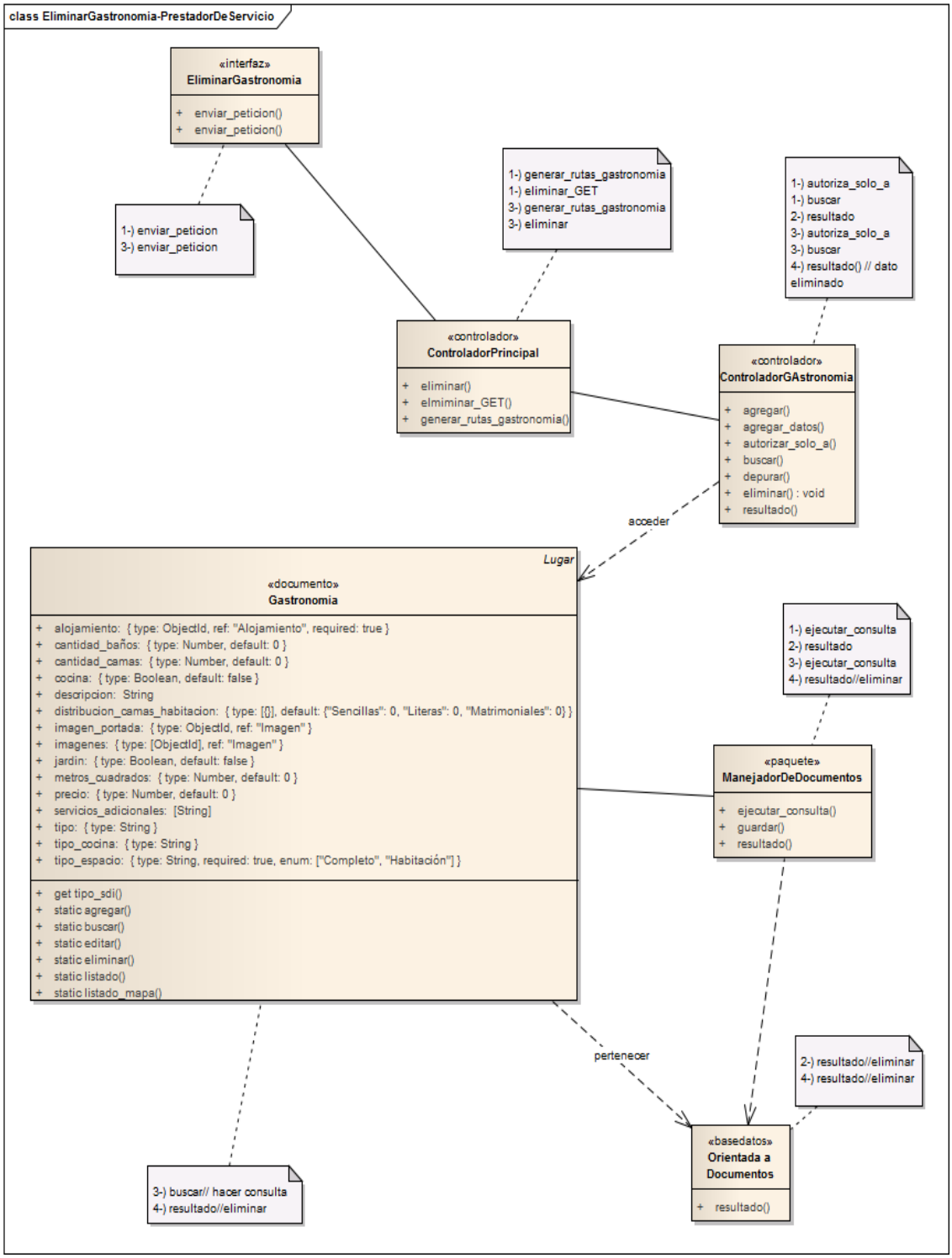


Figura JJ. 2. Diagrama de Clases: Elaborar Listado Gastronomía por usuario Prestador de Servicio.

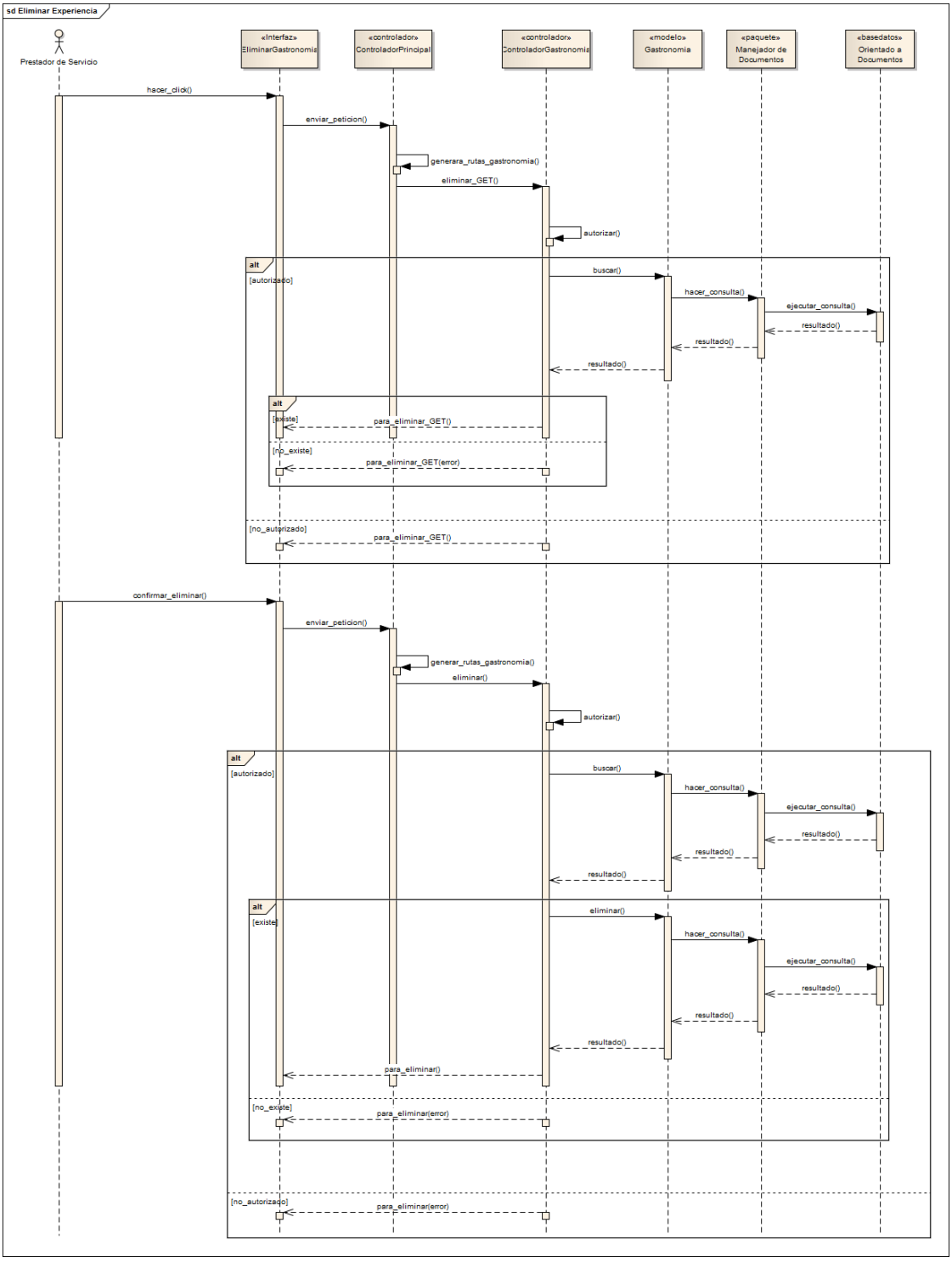


Figura JJ. 3.. Diagrama de secuencia: Elaborar Listado Gastronomía por usuario Prestador de Servicio

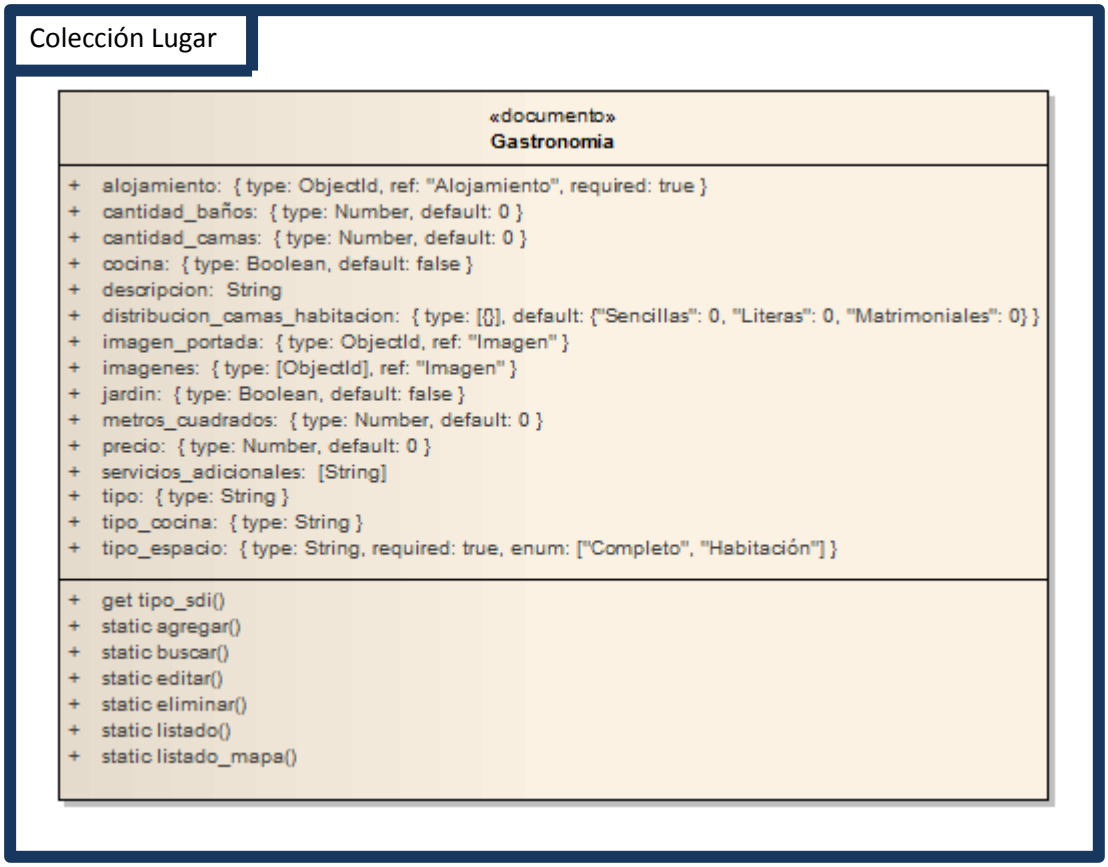


Figura JJ. 4. Colección Gastronomía: Listado de Gastronomía por usuario Prestador de Servicio

A continuación, se muestran los modelos de plantillas para el proceso de Listado de gastronomía.

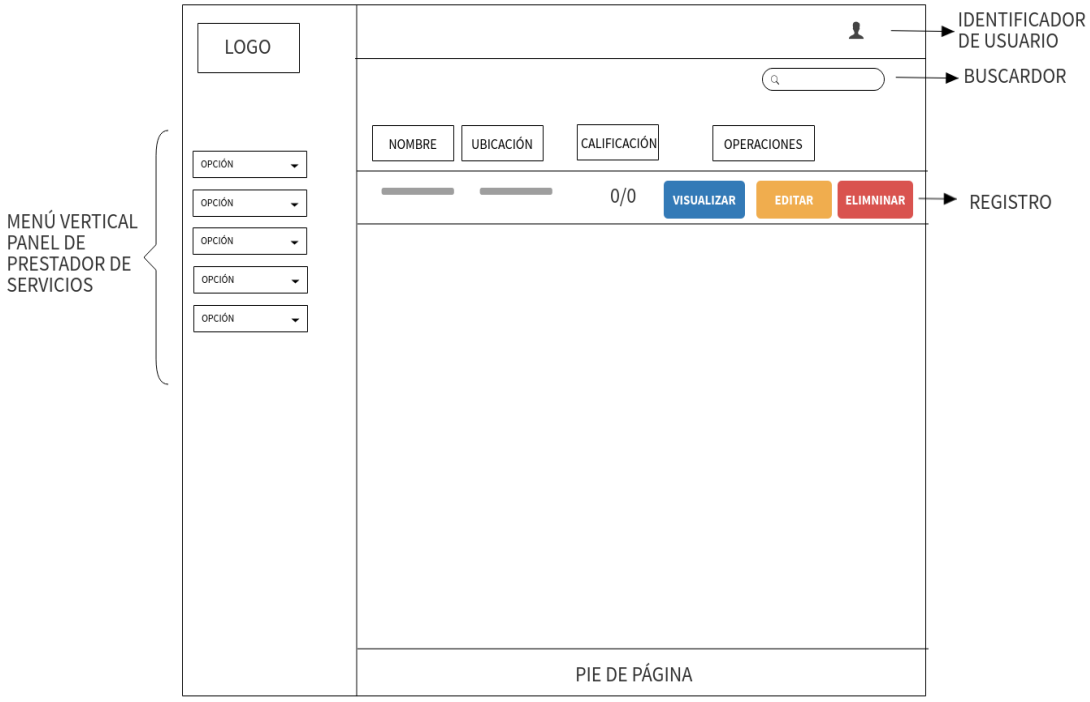


Figura JJ. 5.Plantilla de interfaz: Listado de Gastronomía por usuario prestador de servicio.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Lista de Gastronomía y el sistema responde mostrando la pantalla correspondiente.

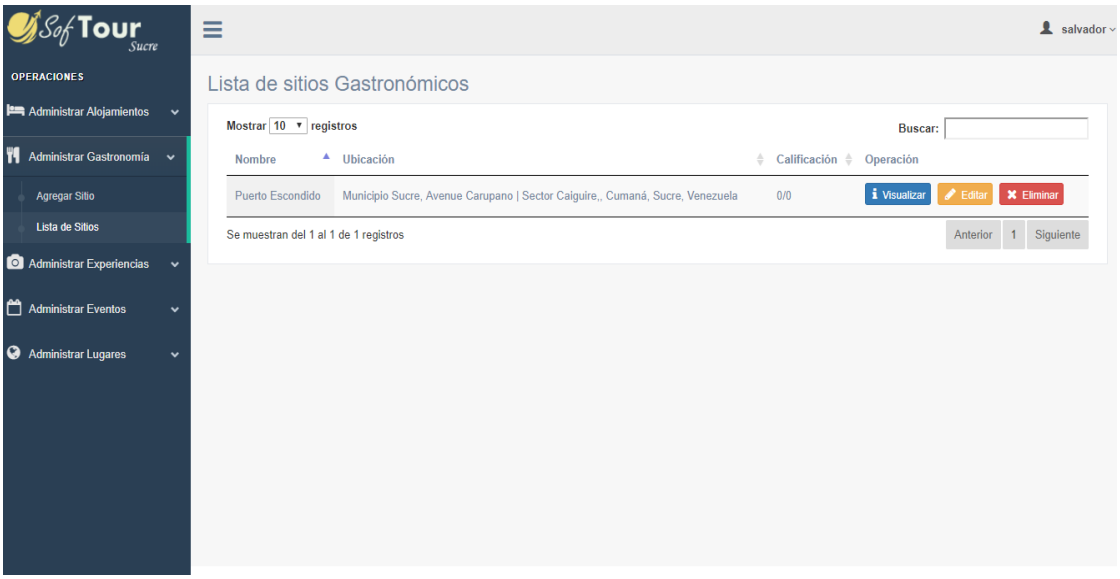


Figura JJ. 6.. Interfaz de proceso de Listar Gastronomía por usuario Administrador

Parte del código desarrollado para la funcionalidad Listado de Gastronomía por usuario Prestador de servicio.

```
listado(req, res, next) {
    let query = {};

    if(req.session.passport.user.tipo_usr == "PrestadorServicio" && req.params.tipo_entidad != "lugar")
        query.prestador_servicio = req.session.passport.user.id;

    if(req.params.tipo_entidad == "lugar")
        query.__t = null;
    else{
        query.__t = req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1);
        query.aprobado = true;
    }

    this._Lugar.listado({"consulta": query}, (error, lista) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_listado(req, res);
        }

        res.locals.datos = {"value": lista};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_listado(req, res);
    })
}
```

Figura JJ. 7. Código desarrollado para el proceso Listar gastronomía por Prestador de servicio

Apéndice KK. Ciclo de Versión Prestador de Servicios: Iteración Consultar Evento.

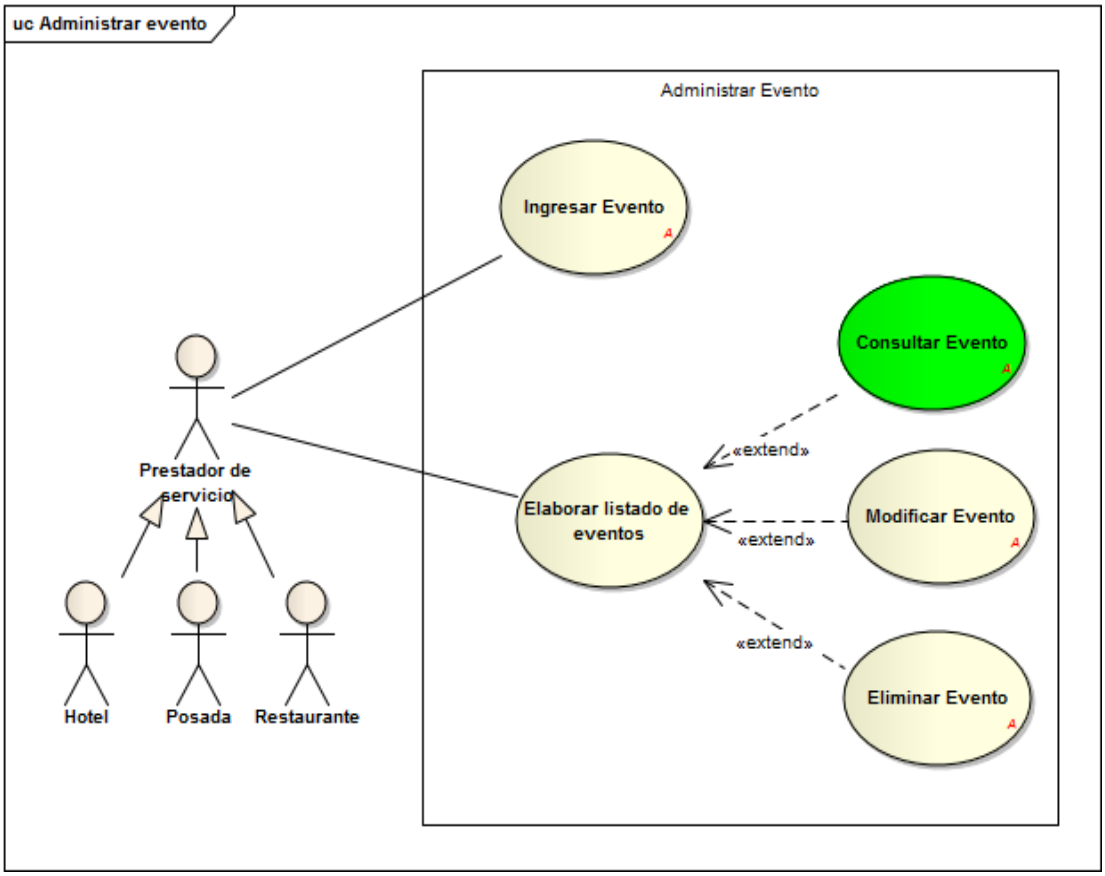


Figura KK. 1.Caso de uso: Consultar Evento por prestador de servicio.

Tabla KK. 1. Descripción Caso de uso Consultar Evento por usuario prestador de servicio

Nombre del Caso de uso	Consultar Evento	
Versión	1.0 25/10/2017	
Actor	Prestador de Servicios.	
Descripción del Casos de Uso	El caso de uso consiste en consultar los atributos del evento de manera satisfactoria.	
Precondición	Para consultar un evento, este debe estar registrado en el sistema.	
Secuencia Normal de Eventos	Paso	Acción
	1	El Prestador de Servicios selecciona la opción “Consultar”.
	2	El sistema ejecuta la búsqueda del evento.
	3	El Prestador de Servicio visualiza la consulta de los atributos del evento.
	4	El sistema finaliza el caso de uso.
Postcondición	Se consultan los atributos del evento de manera satisfactoria.	
Curso alternativo de Eventos	Paso	Acción
Reglas del negocio	1. El evento debe estar registrado para que pueda ser consultado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

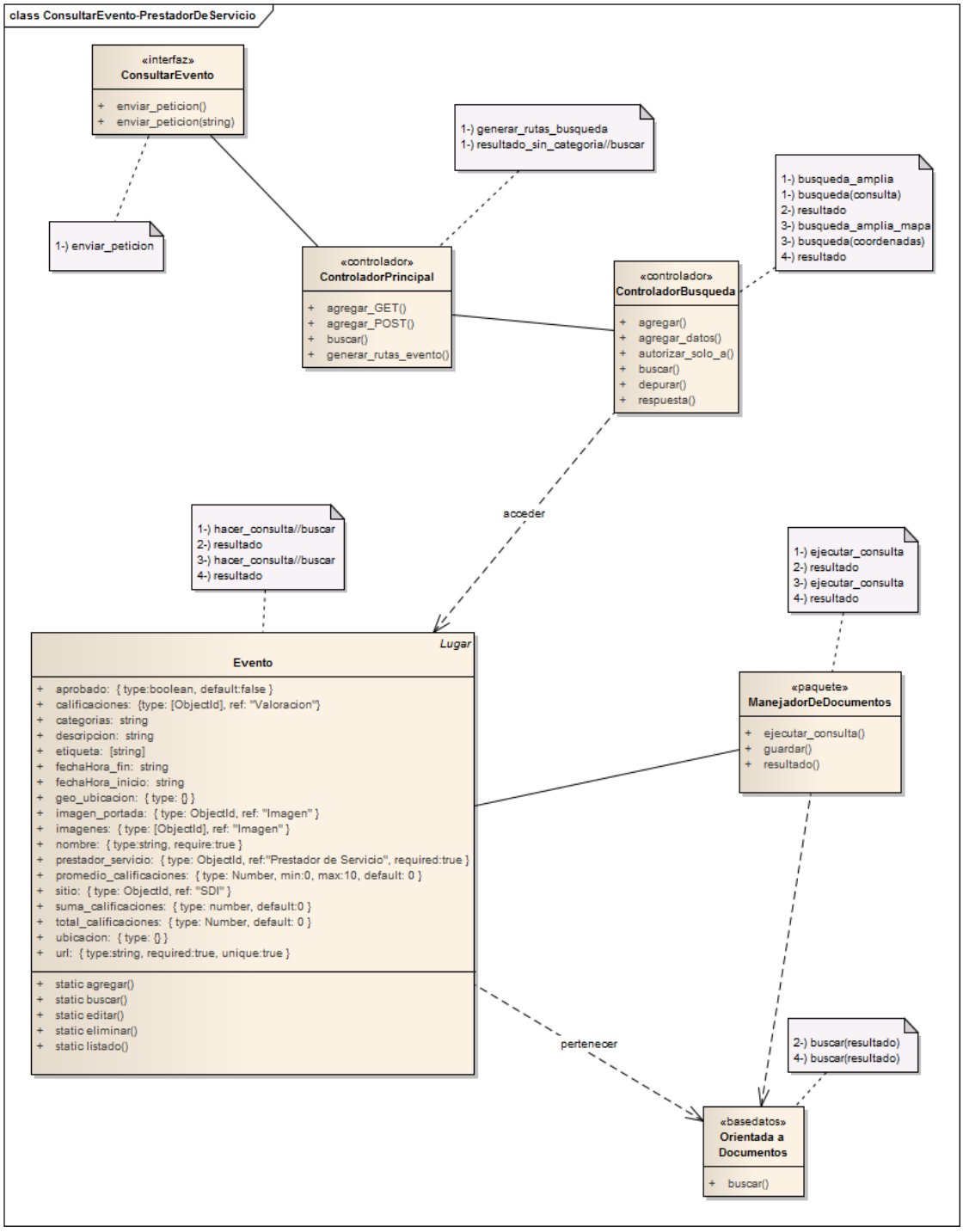


Figura KK. 2. Diagrama de Clases: Consultar Evento por usuario Prestador de Servicios

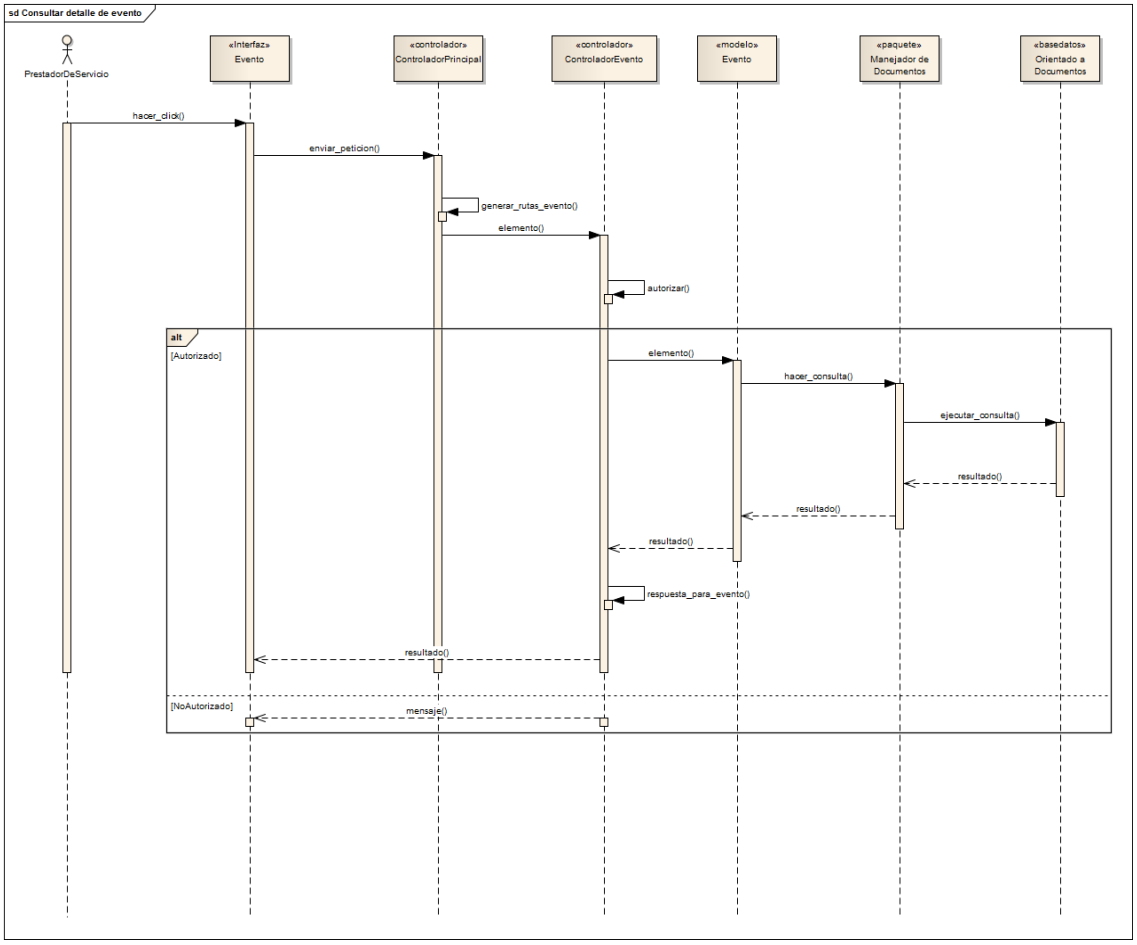


Figura KK. 3. Diagrama de secuencia: Consultar Evento por usuario Prestador de Servicio

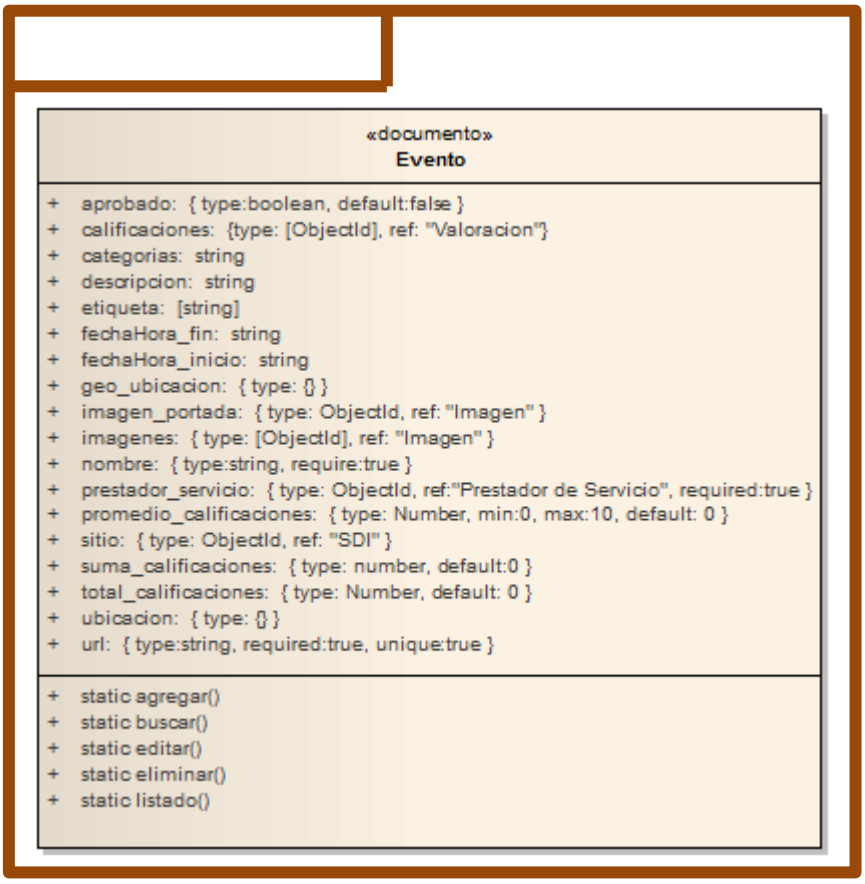


Figura KK. 4. Colección de Evento: Consultar evento por usuario Prestador de Servicio

A continuación, se muestran los modelos de plantillas para el proceso de consultar evento.

Figura KK. 5. Plantilla de interfaz: Consultar Evento por usuario prestador de servicio.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Consultar Evento y el sistema responde mostrando la interfaz con las especificaciones del Evento.

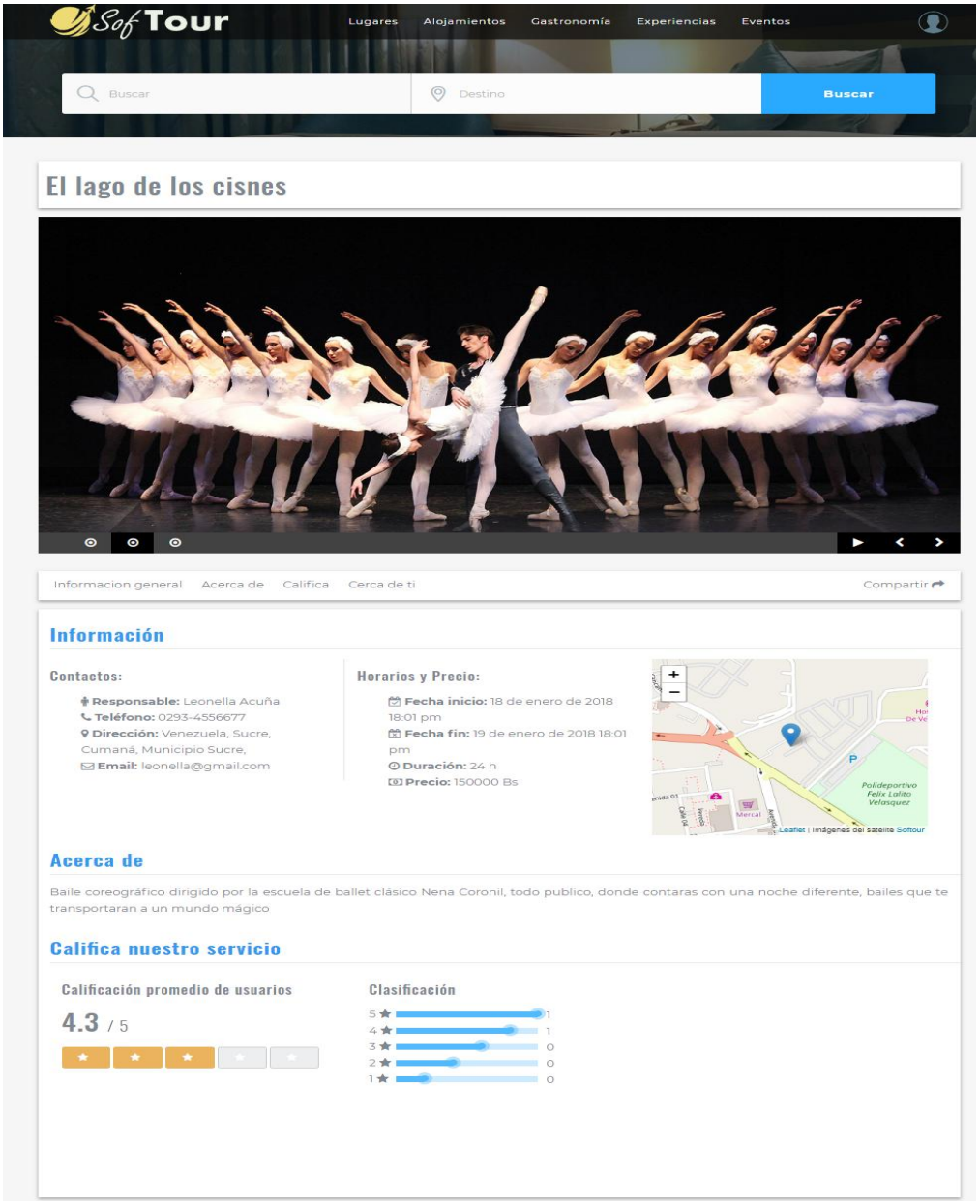


Figura KK. 6.Interfaz de proceso de Consultar Evento por usuario Prestador de servicio

Código desarrollado para la funcionalidad Consultar Evento por usuario Prestador de Servicio.

```
consultar(req, res, next) {
    this._Evento.buscar_consulta({"url": req.params.url.toLowerCase()}, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": buscado};
        if (req.isAuthenticated())
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_consultar(req, res);
    })
}
```

Figura KK. 7. Código desarrollado para el proceso Consultar Evento por usuario Prestador de servicio.

Prueba Consultar Evento

La funcionalidad de este módulo consiste en mostrarle al usuario restador de servicio las características de un evento en particular. Para poder realizar esta prueba se necesitó tener en la base de datos del sistema eventos cargados con antelación, por tal razón se utilizaron los elementos ingresados en pruebas anteriores.

Esta prueba se realizó de manera manual, seleccionando el evento a visualizar en la lista de eventos.

Desde el panel principal de prestador de servicio se procede a seleccionar el listado de eventos agregados en sistema.

Figura KK. 8.Prueba visualizar Evento por Prestador de Servicio

A continuación, se seleccionó el evento a visualizar y se procede a pulsar el botón de visualizar contemplado en sistema.

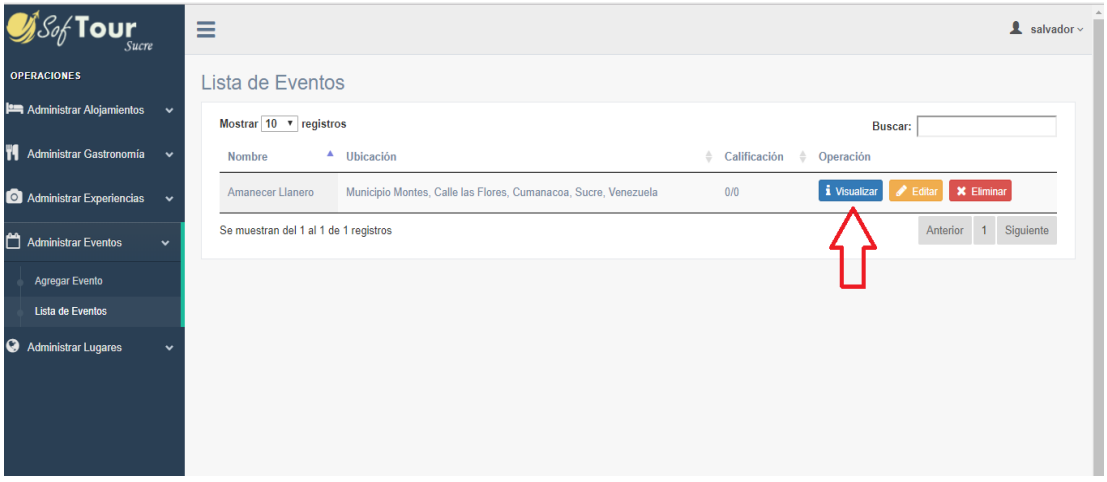


Figura KK. 9. Prueba visualizar Evento por Prestador de Servicio

En la siguiente imagen se determina que la prueba obtuvo resultados satisfactorios ya que el sistema realizó la función solicitada.

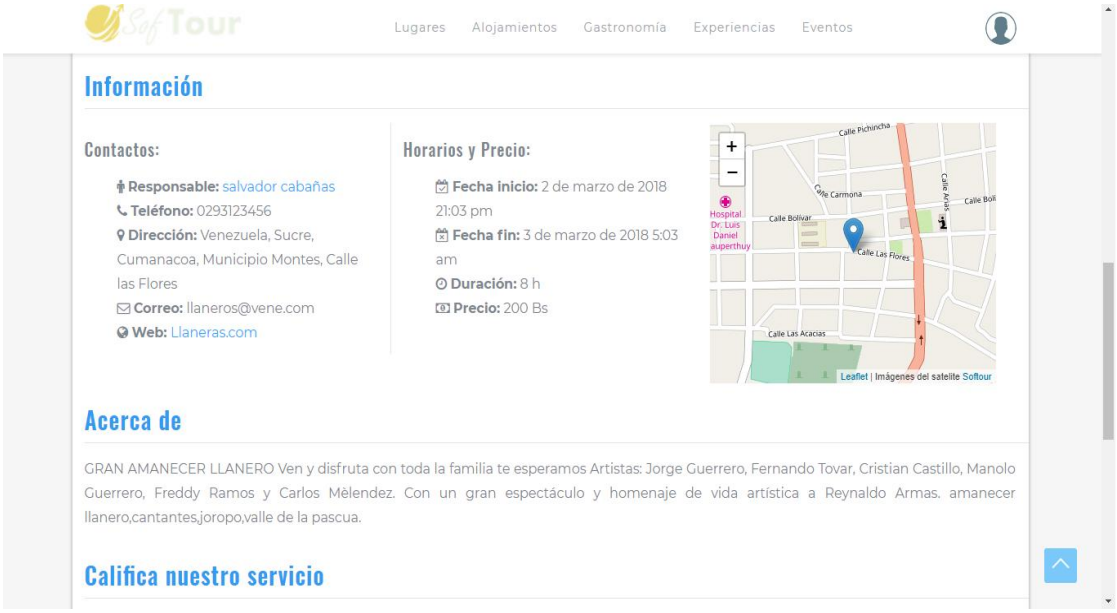


Figura KK. 10. Prueba visualizar Evento por Prestador de Servicio

Conclusión de la prueba

La prueba se realizó de forma satisfactoria obteniendo resultados positivos.

Prueba de interfaz y usabilidad

Dado que para el diseño de interfaz el equipo se basó en usar el esquema de interfaz que se encuentran en la vanguardia y siempre mantuvo la premisa de utilizar normas de usabilidad, en análisis se determinó que la interfaz es apta para la interacción con los usuarios.

Apéndice LL. Ciclo de Versión Prestador de Servicios: Iteración Eliminar Alojamiento.

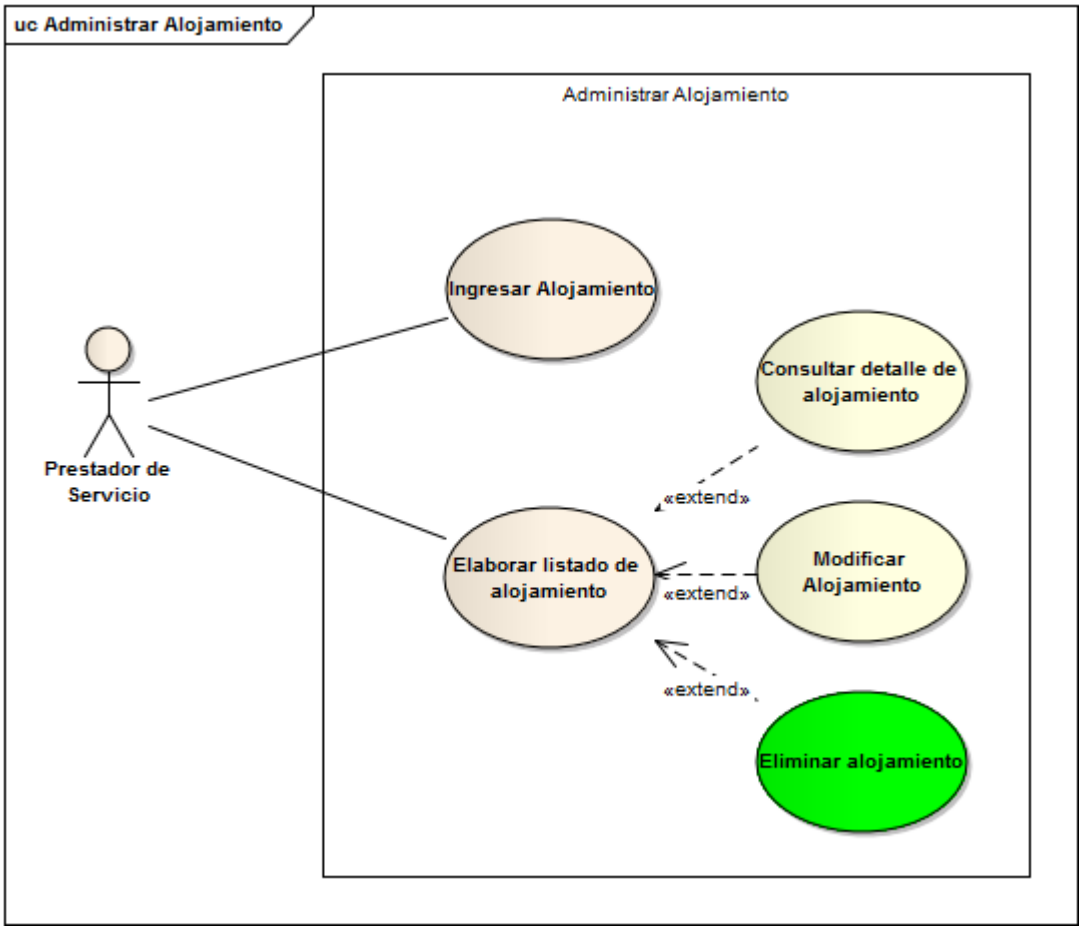


Figura LL. 1. Caso de uso: Eliminar Alojamiento por prestador de servicio.

Tabla LL. 1. Descripción Caso de uso Eliminar Alojamiento por usuario prestador de servicio

Nombre del Caso de uso	Eliminar alojamiento.	
Versión	1.0 25/10/2017	
Actor	Prestador de Servicio.	
Descripción del Caso de Uso	El caso de uso consiste en eliminar un alojamiento de manera satisfactoria.	
Precondición	Para eliminar un alojamiento, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Prestador de Servicio selecciona la opción “Eliminar alojamiento”.
	2	El sistema ejecuta la búsqueda del alojamiento.
	3	El Prestador de Servicio visualiza el alojamiento que desea eliminar con un mensaje de confirmación y el usuario selecciona la opción “Si, estoy seguro”.
	4	El sistema realiza la eliminación del alojamiento en la base de datos del sistema y finaliza el caso de uso.
Postcondición	Se elimina el alojamiento de la base de datos del sistema de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Prestador de Servicio decide no continuar con el caso de uso y selecciona cancelar el caso de uso, seleccionando otra opción del sistema.
	4.1.	El sistema finaliza el caso de uso.
Reglas del negocio	1. El alojamiento debe estar registrado para que pueda ser eliminado.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

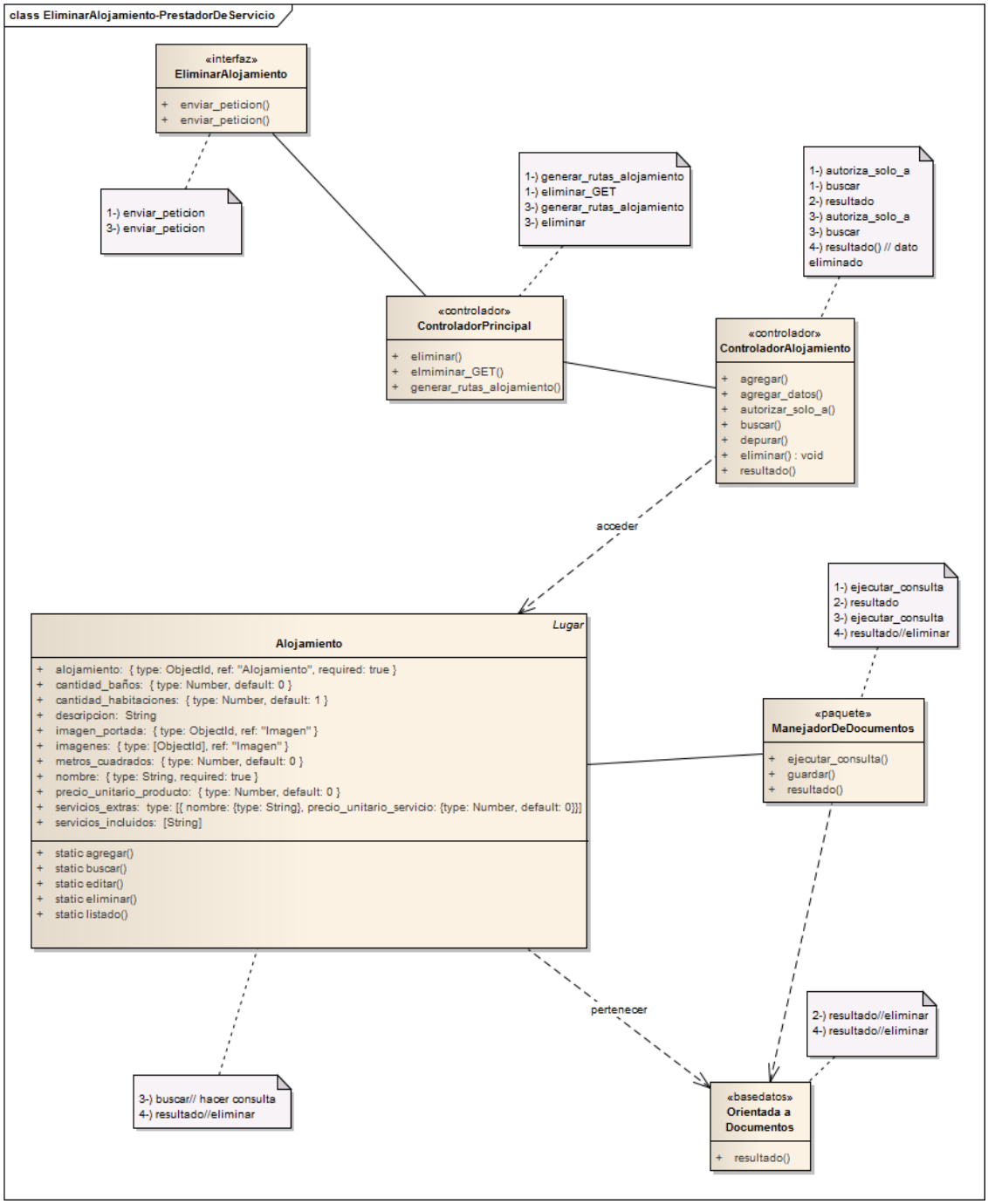


Figura LL. 2. Diagrama de Clases: Eliminar Alojamiento por usuario Prestador de Servicios

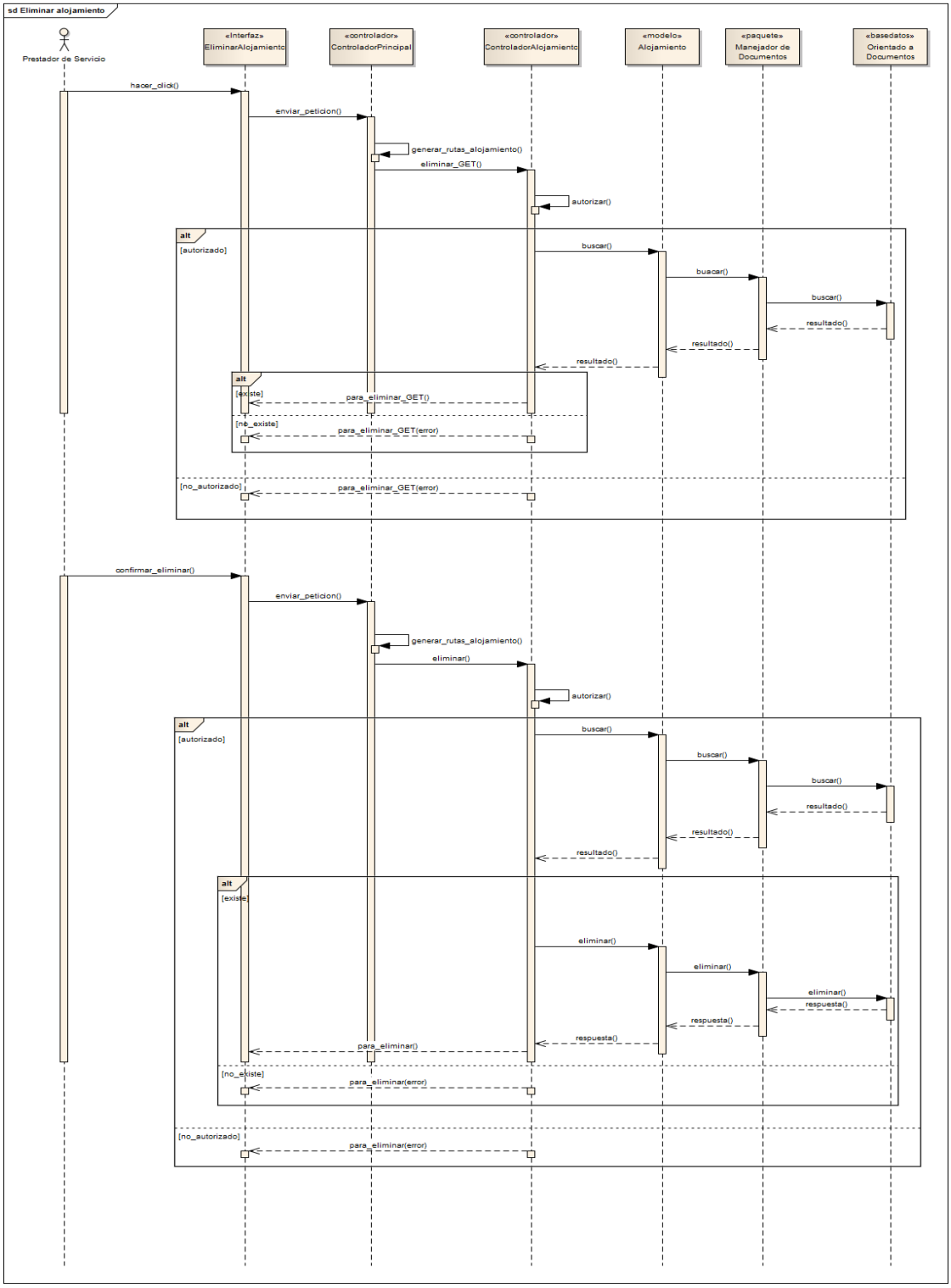


Figura LL. 3. Diagrama de secuencia: Eliminar alojamiento por usuario Prestador de Servicio

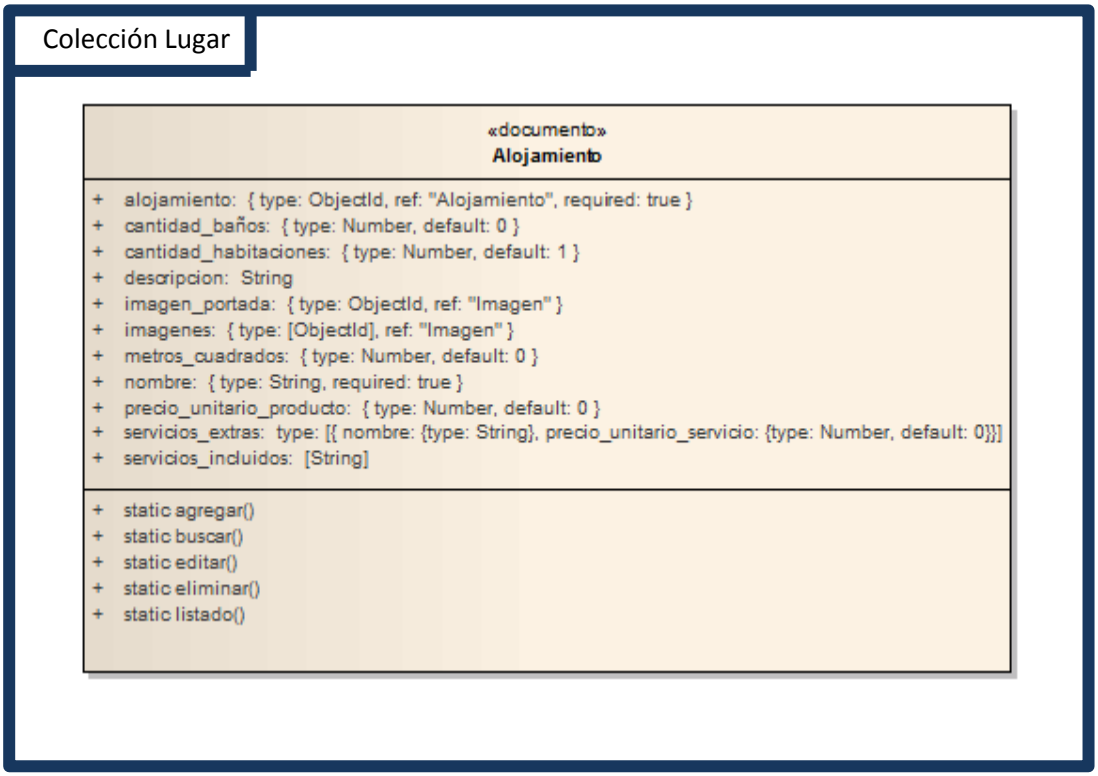


Figura LL. 4.Colección de Alojamiento: Eliminar alojamiento por usuario Prestador de Servicio

A continuación, se muestran los modelos de plantillas para el proceso de eliminar alojamiento.

Figura LL. 5. Plantilla de interfaz 1: Eliminar Alojamiento por usuario prestador de servicio.

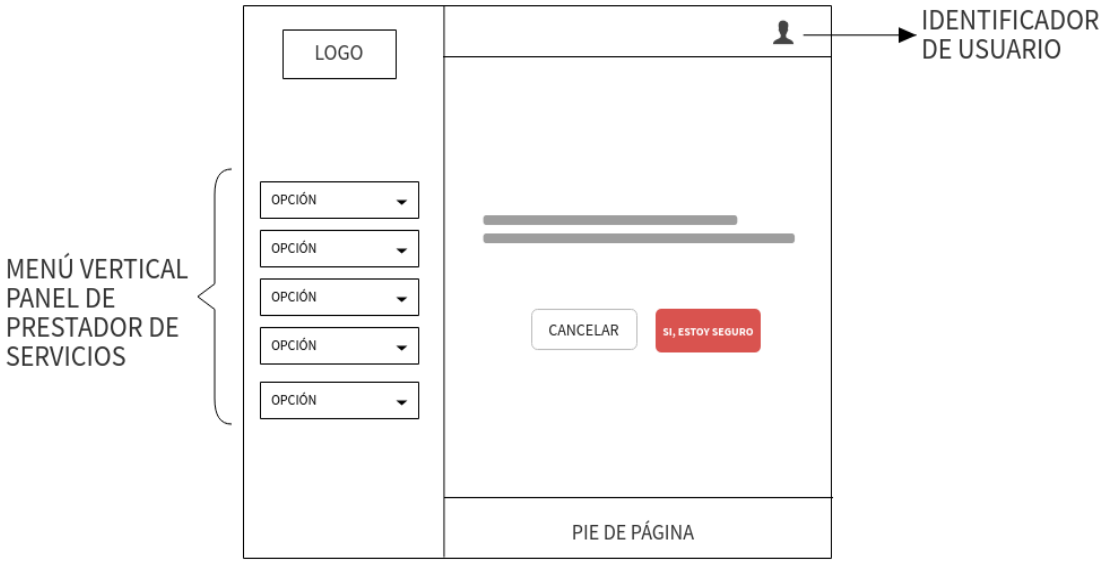


Figura LL. 6. Plantilla de interfaz 2: Eliminar Alojamiento por usuario prestador de servicio.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Eliminar Alojamiento y el sistema responde mostrando la interfaz de confirmación de acción.

Figura LL. 7. Interfaz 1 de proceso de Eliminar Alojamiento por usuario Prestador de servicio

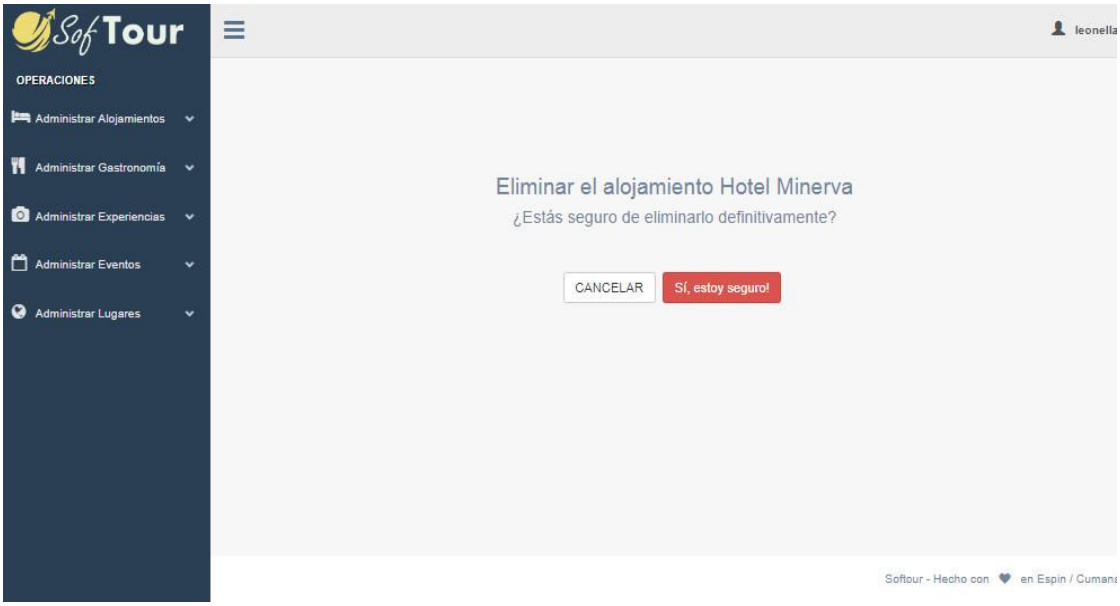


Figura LL. 8. Interfaz 2 de proceso de Eliminar Alojamiento por usuario Prestador de servicio

Parte del código desarrollado para la funcionalidad Eliminar Alojamiento por usuario Prestador de Servicio

```
eliminar(req, res, next) {
    let tipo_entidad = req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.
        substring(1);

    this._Lugar.eliminar({"url": req.params.url.toLowerCase(), "__t": tipo_entidad}, (error, eliminado) => {
        if(error) {
            res.locals.datos = {"error": error};
            res.locals.datos.value = {"usuario_sesionado": req.session.passport.user};
            this._respuesta.para_eliminar(req, res);
        }
        if(!eliminado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": {"url": "listado"}};
        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_eliminar(req, res);
    });
}
```

Figura LL. 9. Código desarrollado para el proceso Eliminar Alojamiento por usuario Prestador de servicio.

Prueba Eliminar Alojamiento

La siguiente función a evaluar consiste en borrar del sistema alojamientos agregados con anterioridad por el prestador de servicio.

Para llevar a cabo esta prueba se tomó en cuenta elementos agregados en pruebas anteriores, a continuación, se muestra secuencia de imágenes de una de las pruebas realizadas.

Se selecciona el alojamiento que será eliminado para llevar a cabo la prueba, y se procede a seleccionar le botón eliminar.

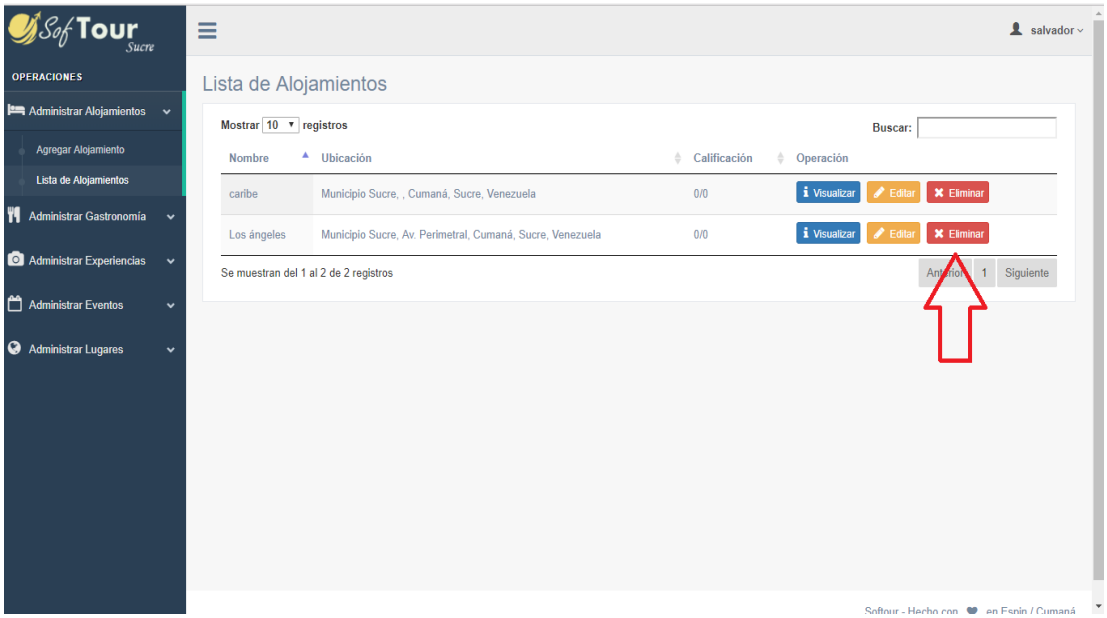


Figura LL. 10.Prueba eliminar alojamiento por prestador de servicio

El sistema consulta al usuario si quiere seguir con la acción mediante una interfaz única para esto, como se muestra en la siguiente imagen:

Figura LL. 11.Prueba eliminar alojamiento por prestador de servicio.

A continuación, se muestra la lista de alojamientos agregados en sistema donde se valida que fue eliminado el alojamiento seleccionado.

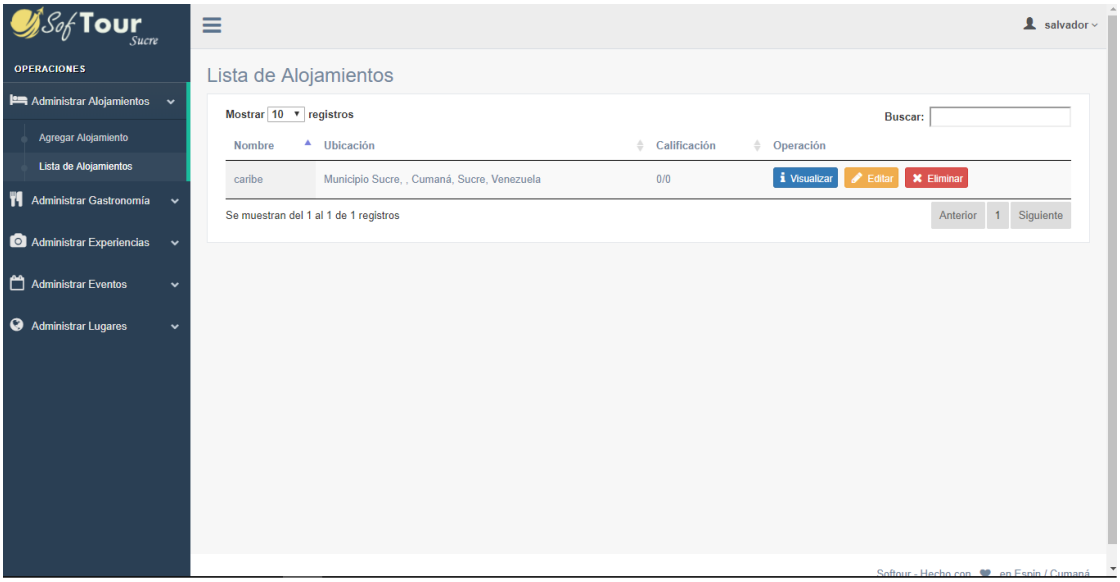


Figura LL. 12.Prueba eliminar alojamiento por prestador de servicio.

Conclusión de la prueba

Como se notó en la secuencia de imágenes se puede determinar que la función se ejecuta de forma correcta.

Prueba de interfaz y usabilidad

Este interfaz perteneces al conjunto de interfaces de Administrador que se evaluó con anterioridad y supero el análisis realizado.

Apéndice MM. Consultar Alojamiento por Turista

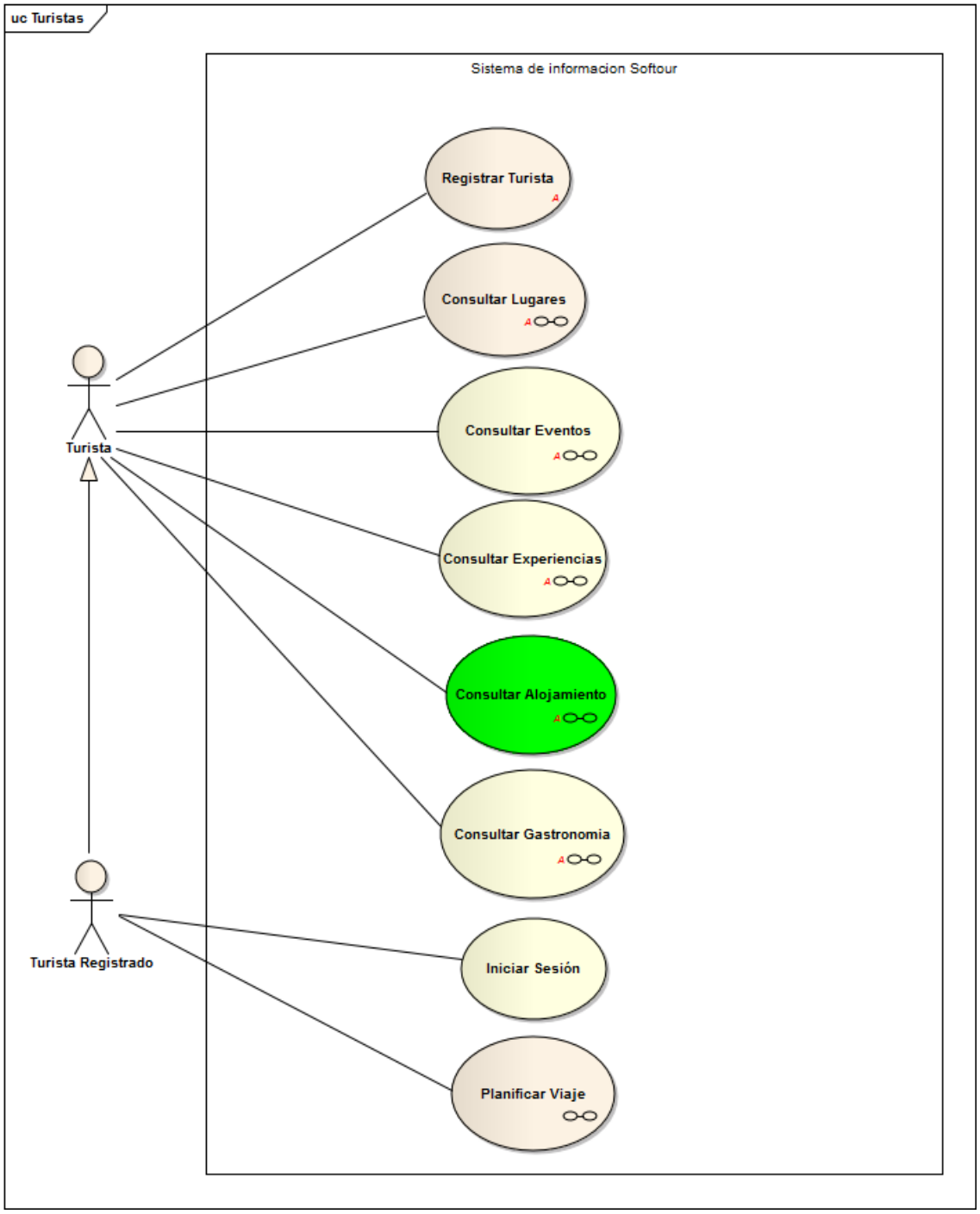


Figura MM. 1. Caso de uso: Consultar Alojamiento por usuario Turista.

Tabla MM. 1. Descripción Caso de uso Consultar Alojamiento por usuario Turista.

Nombre del Caso de uso	Consultar Alojamiento	
Versión	1.0 03/11/2017	
Actor	Turista.	
Descripción del Caso de Uso	El caso de uso consiste en consultar los alojamientos disponibles desde una perspectiva visual, organizada y llamativa. Permitiendo enlazar el caso de uso con patrones de búsqueda específicos relacionados con lo que desea el turista.	
Precondición	Ninguna	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El caso de uso comienza cuando el Turista hace clic en el menú en la categoría "Alojamientos" o alguna subcategoría.
	2	El sistema despliega los alojamientos disponibles y muestra los posibles filtros que aplican para refinar aún más la búsqueda, acordes a lo que desea el turista. Esto se realiza extendiendo <<extend>> la funcionalidad de la búsqueda con el caso de uso: a.- Administrar patrones de búsqueda de alojamientos.
	3	El Turista visualiza los alojamientos disponibles que se muestran y selecciona uno de ellos para ver el detalle del mismo.
	4	El sistema muestra el detalle del alojamiento seleccionado.
	5	El Turista visualiza los atributos del alojamiento.
	6	El sistema cierra el caso de uso y retorna el control.
Postcondición	Se consulta el alojamiento disponible desde la perspectiva del turista, satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
Reglas del negocio	1. Las consultas de los alojamientos en los que se incluyan filtros o patrones de búsquedas deben considerarse extensiones de este caso de uso. 2. Al seleccionar un alojamiento, el turista podrá visualizar mayor detalle del mismo.	
Inclusiones	Ninguna	
Extensiones	1. Administrar patrones de búsqueda de alojamientos.	

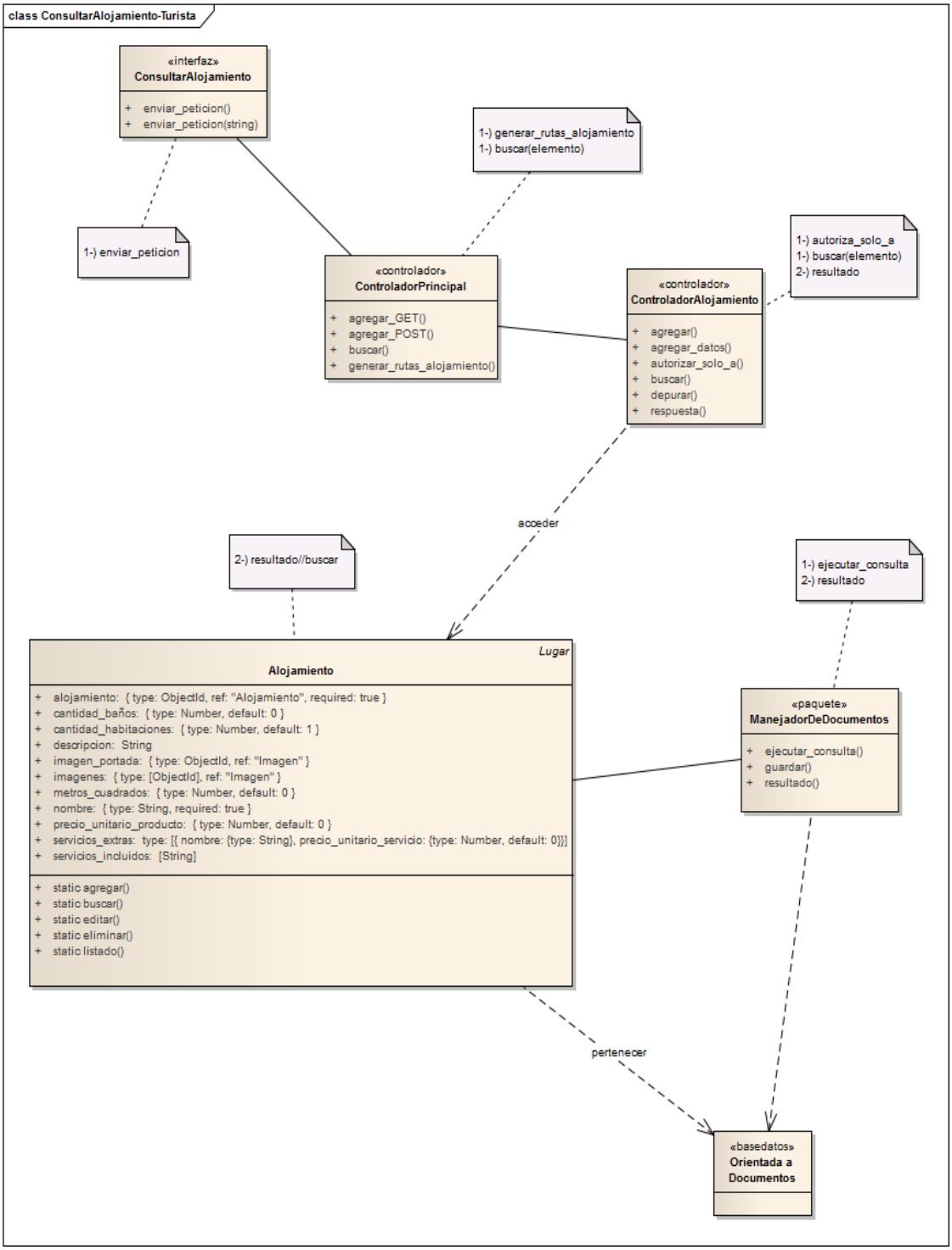


Figura MM. 2. Diagrama de Clases: Consultar Alojamiento por Usuario Turista.

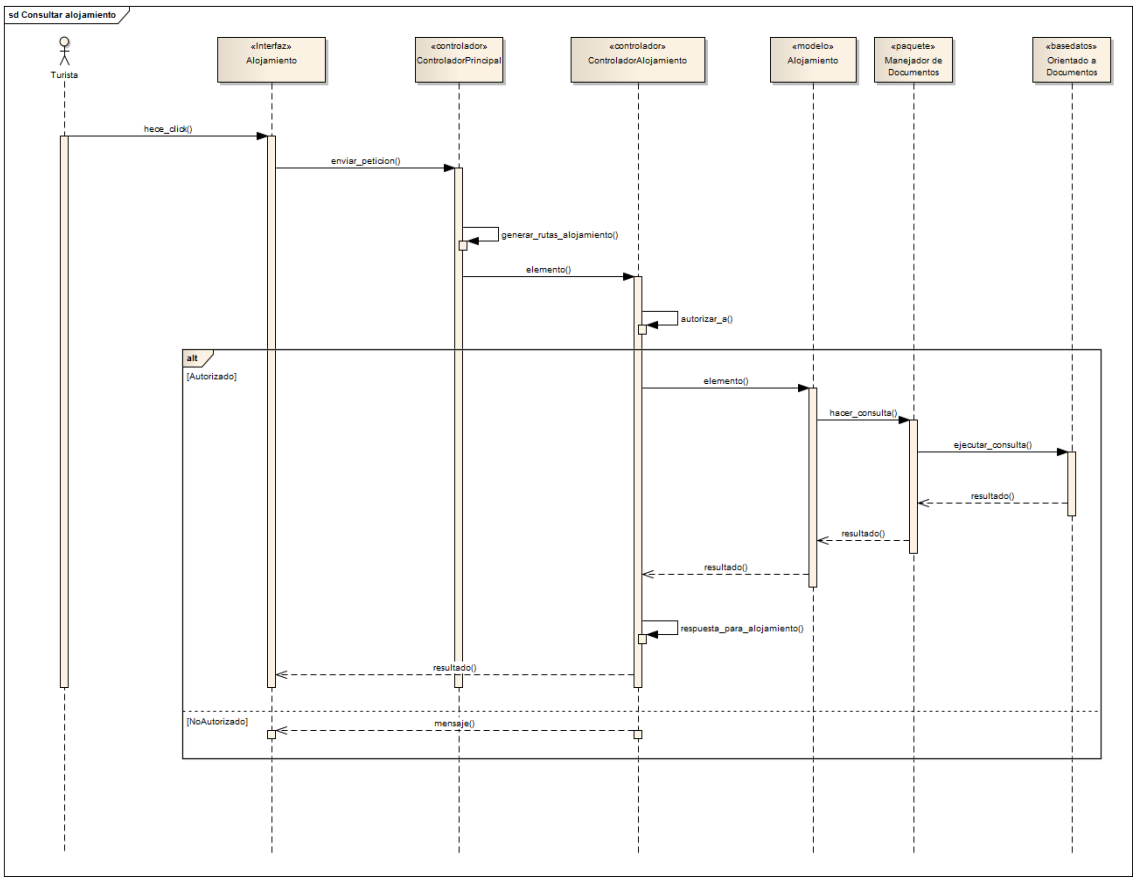


Figura MM. 3. Diagrama de secuencia: Consultar Alojamiento por usuario Turista

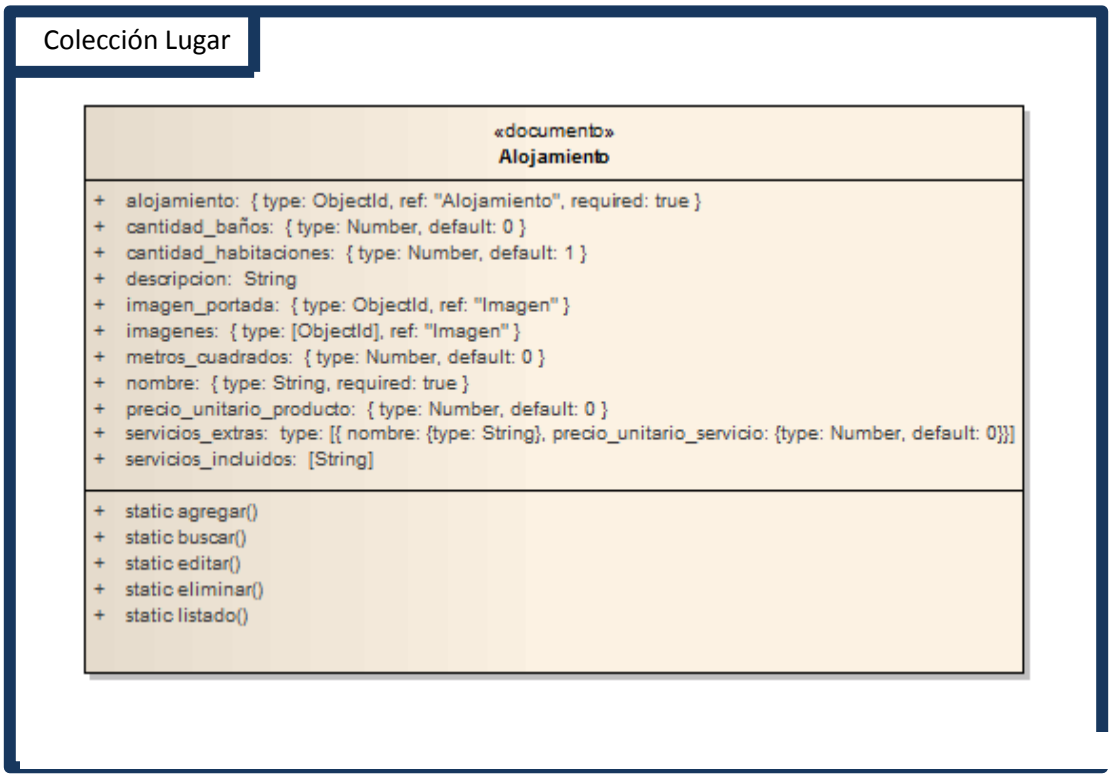


Figura MM. 4. Colección de Alojamiento: Consultar alojamiento por usuario Turista.

A continuación, se muestran los bosquejos utilizados para el diseño de interfaz.

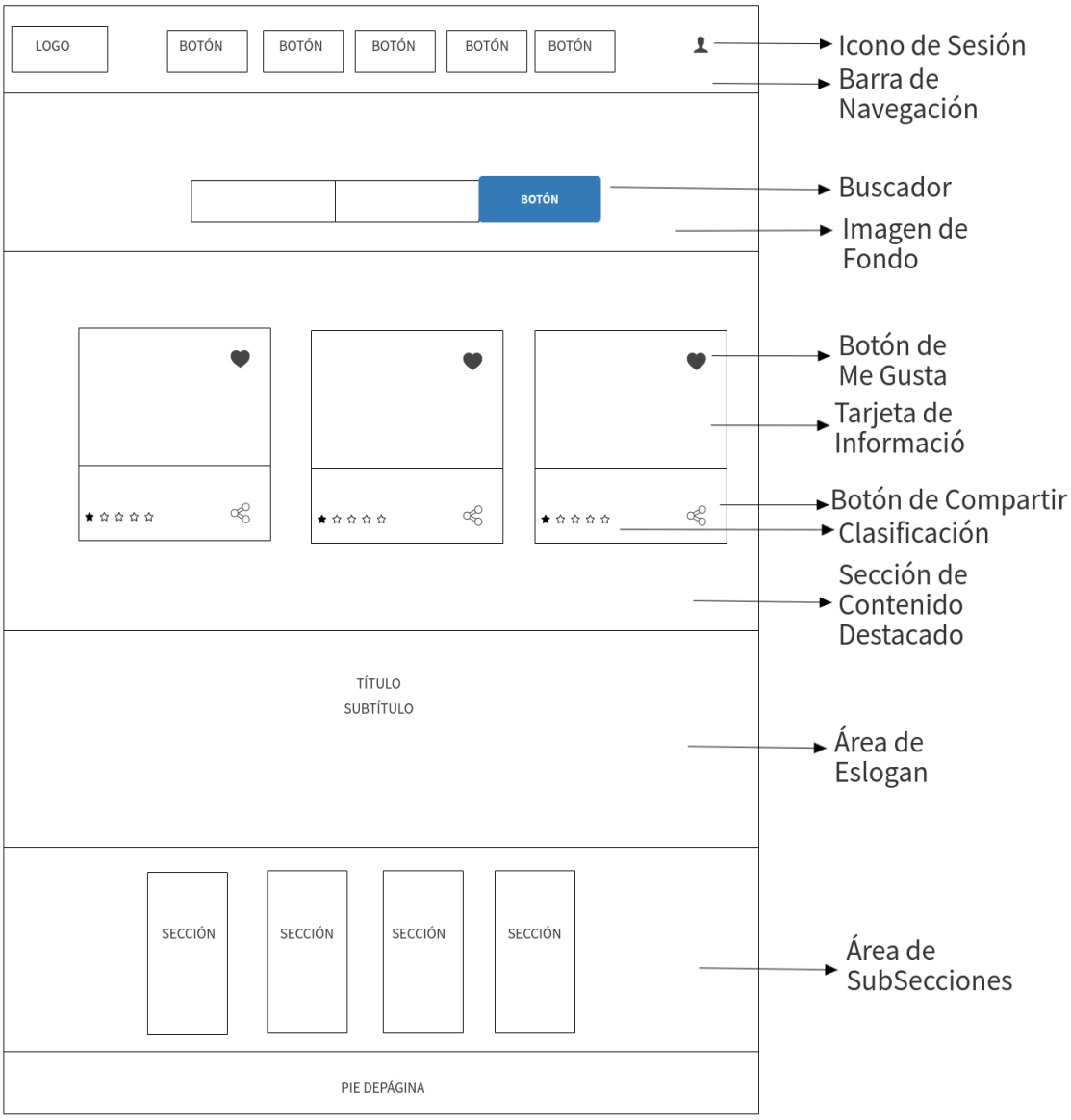


Figura MM. 5. Plantilla de interfaz: Consultar Alojamiento por usuario Turista.

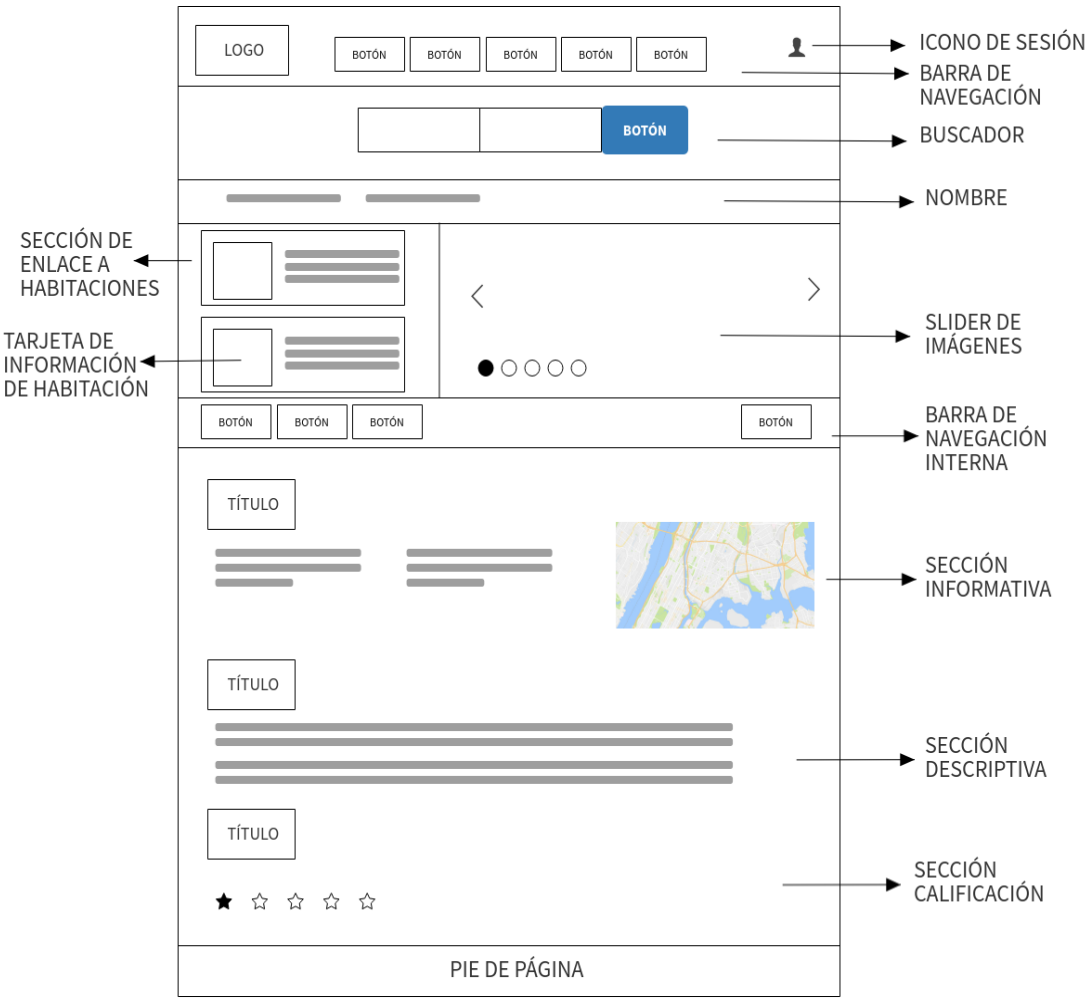


Figura MM. 6. Plantilla de interfaz: Consultar Alojamiento por usuario Turista.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Alojamiento, y luego selecciona un alojamiento en específico y el sistema responde mostrando la interfaz con las especificaciones.

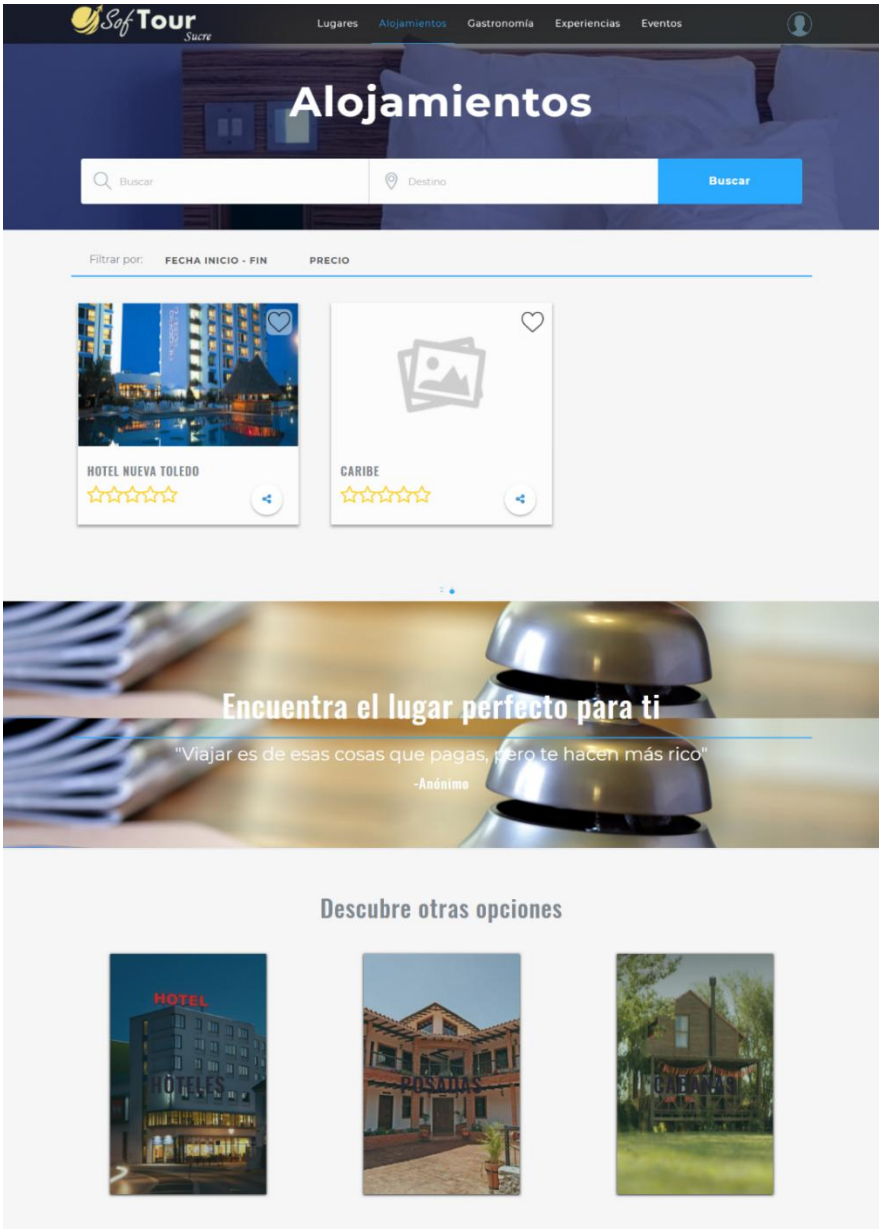


Figura MM. 7. Interfaz de proceso de Consultar Alojamiento por usuario Turista

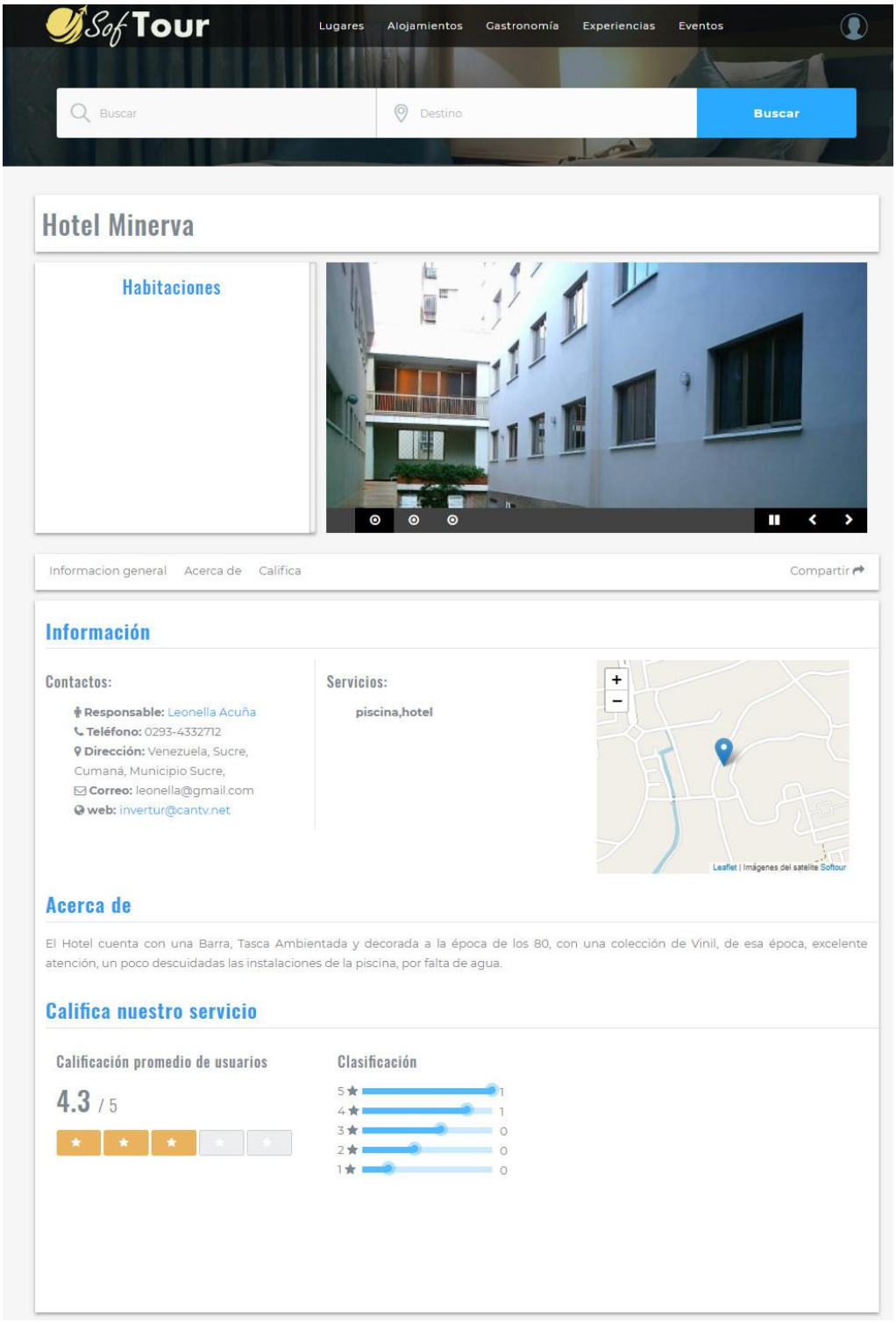


Figura MM. 8. Interfaz de proceso de Consultar Alojamiento por usuario Turista.

Código desarrollado para la funcionalidad Consultar Alojamiento por usuario Turista.

```
consultar(req, res, next) {
    let parametros = {
        "__t": req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1),
        "url": req.params.url.toLowerCase()
    };
    if(parametros.__t)
        parametros.aprobado = true;

    this._Lugar.buscar_consulta(parametros, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        if(buscado.__t){
            this.CategoriaAlojamiento.find({"alojamiento": buscado._id}, (error, lista_categorias) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    if (req.isAuthenticated())
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_consultar_producto(req, res);
                }

                res.locals.datos = {
                    "value": buscado,
                    "lista_categorias": lista_categorias
                };

                if (req.isAuthenticated())
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                this._respuesta.para_consultar(req, res);
            });
        }
        else {
            res.locals.datos = { "value": buscado };

            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
    })
}
```

Figura MM. 9. Código desarrollado para el proceso Consultar Alojamiento por usuario Turista.

.

Prueba Consultar Alojamiento

Consultar alojamiento- turista

Para validar esta función se hará selección de la sección y se validará que se encuentre la información seleccionada.

A continuación, secuencia de imágenes de algunas de las pruebas realizadas. Desde la interfaz principal de SofTour se selecciona el módulo de alojamientos.

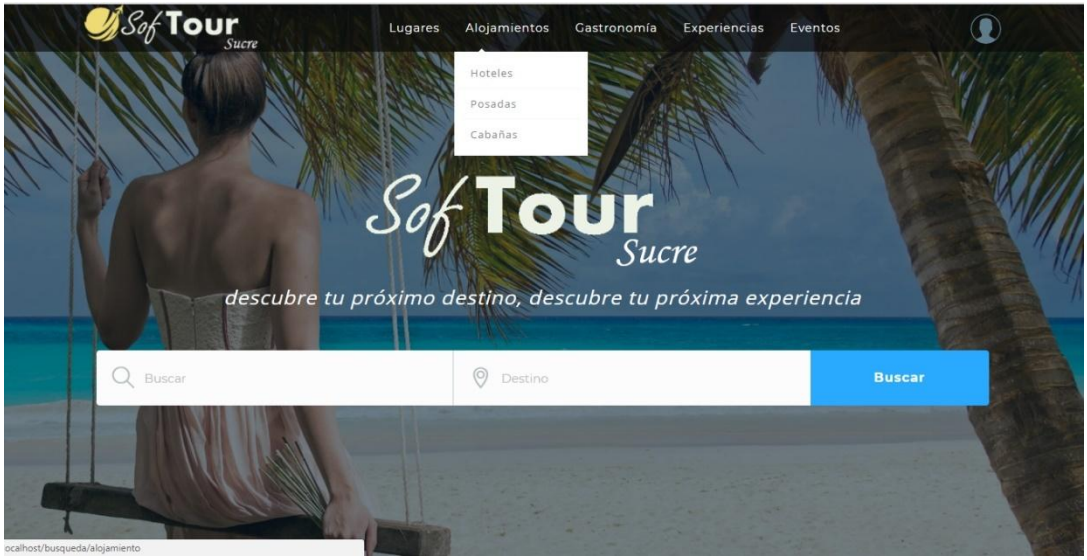


Figura MM. 10. Prueba consultar alojamiento por turista

En la siguiente imagen se muestra que la función se ejecutó de forma exitosa.

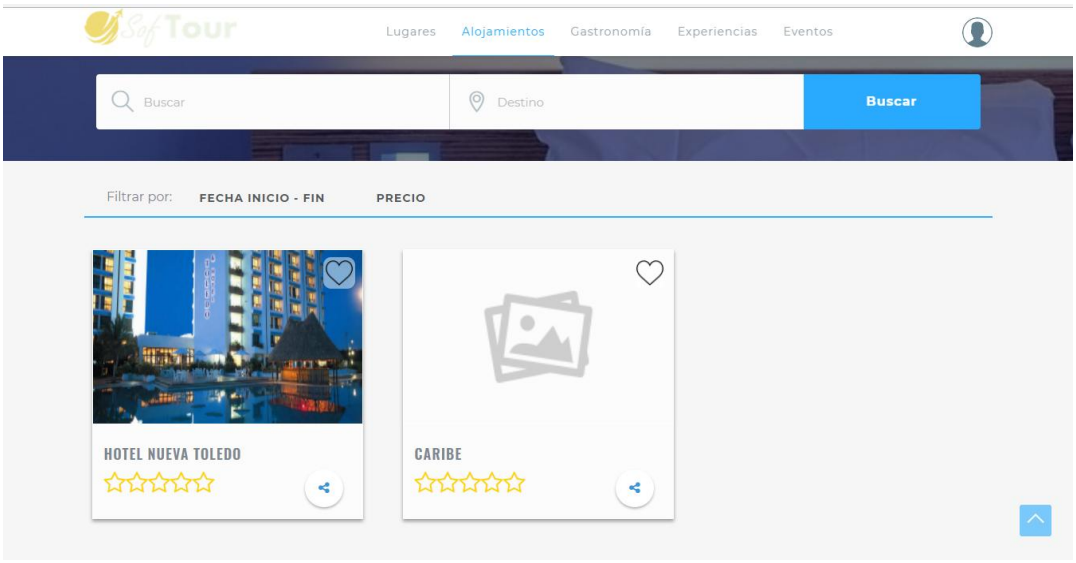


Figura MM. 11. Prueba consultar alojamiento por turista

Conclusión de la prueba

Se visualizo correctamente la interfaz de Alojamiento, lo que indica que la función se cumple correctamente.

Prueba de interfaz y usabilidad

Esta interfaz pertenece al conjunto de interfaces que se muestran al usuario Turista, evaluadas mediante la encuesta previamente realizada.

Apéndice NN. Consultar Evento por Turista

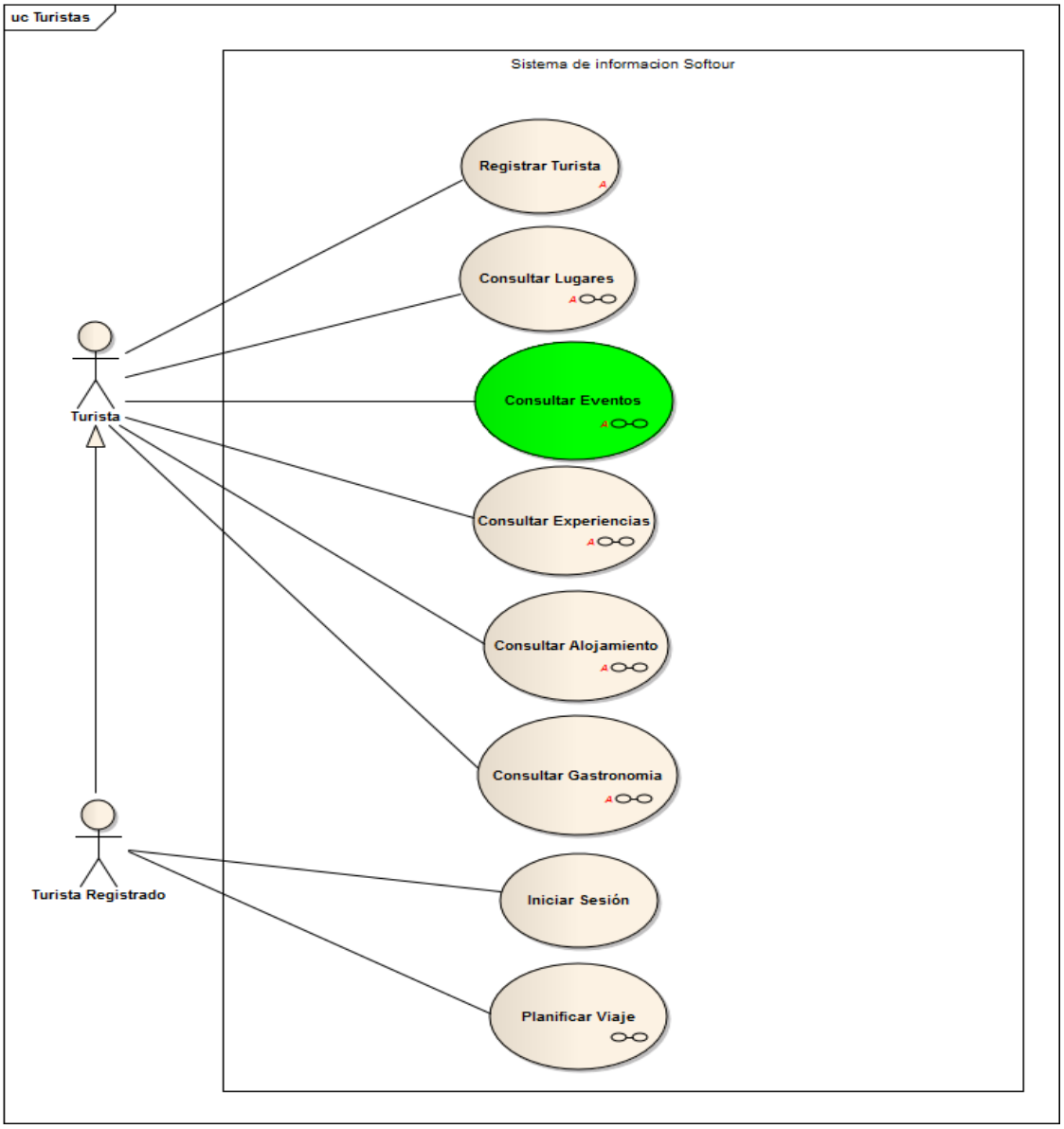


Figura NN. 1. Caso de uso: Consultar Evento por usuario Turista.

Tabla NN. 1. Descripción Caso de uso Consultar Evento por usuario Turista

Nombre del Caso de uso	Consultar Evento	
Versión	1.0 03/11/2017	
Actor	Turista.	
Descripción del Caso de Uso	El caso de uso consiste en consultar los eventos desde una perspectiva visual, organizada y llamativa. Permitiendo enlazar el caso de uso con patrones de búsqueda específicos relacionados con lo que desea el turista.	
Precondición	Ninguna	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El caso de uso comienza cuando el Turista hace clic en el menú en la categoría “Eventos” o alguna subcategoría.
	2	El sistema despliega los eventos disponibles y muestra los posibles filtros que aplican para refinar aún más la búsqueda, acordes a lo que desea el turista. Esto se realiza extendiendo <<extend>> la funcionalidad de la búsqueda con el caso de uso: a.- Administrar patrones de búsqueda de eventos.
	3	El Turista visualiza los eventos disponibles que se muestran y selecciona uno de ellos para ver el detalle del mismo.
	4	El sistema muestra el detalle del evento seleccionado.
	5	El Turista visualiza los atributos del evento.
	6	El sistema cierra el caso de uso y retorna el control.
Postcondición	Se consulta el evento desde la perspectiva del turista, satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
Reglas del negocio	1. Las consultas de los eventos en los que se incluyan filtros o patrones de búsquedas deben considerarse extensiones de este caso de uso. 2. Al seleccionar un evento, el turista podrá visualizar mayor detalle del mismo.	
Inclusiones	Ninguna	
Extensiones	1. Administrar patrones de búsqueda de eventos.	

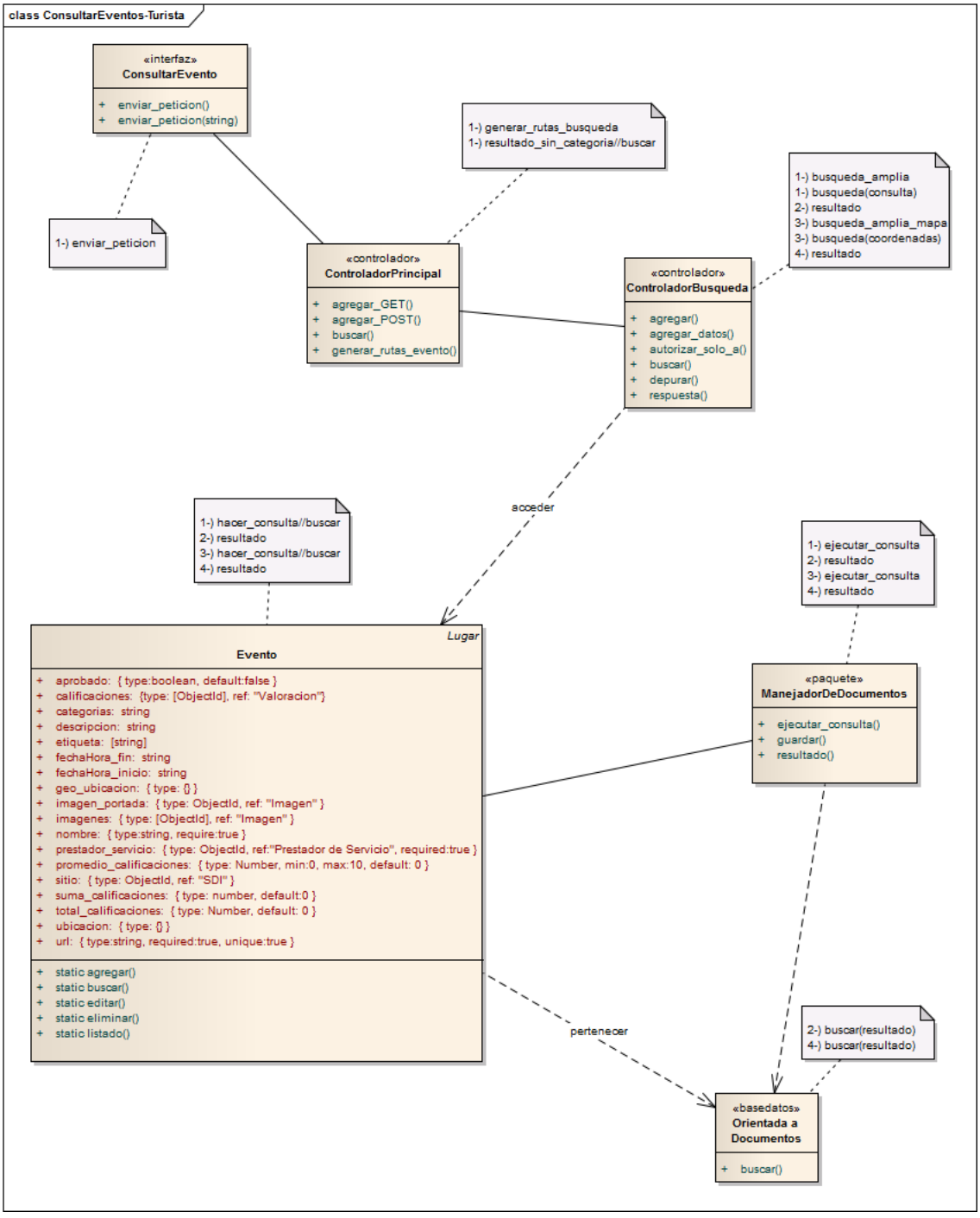


Figura NN. 2. Diagrama de Clases: Consultar Evento por Usuario Turista.

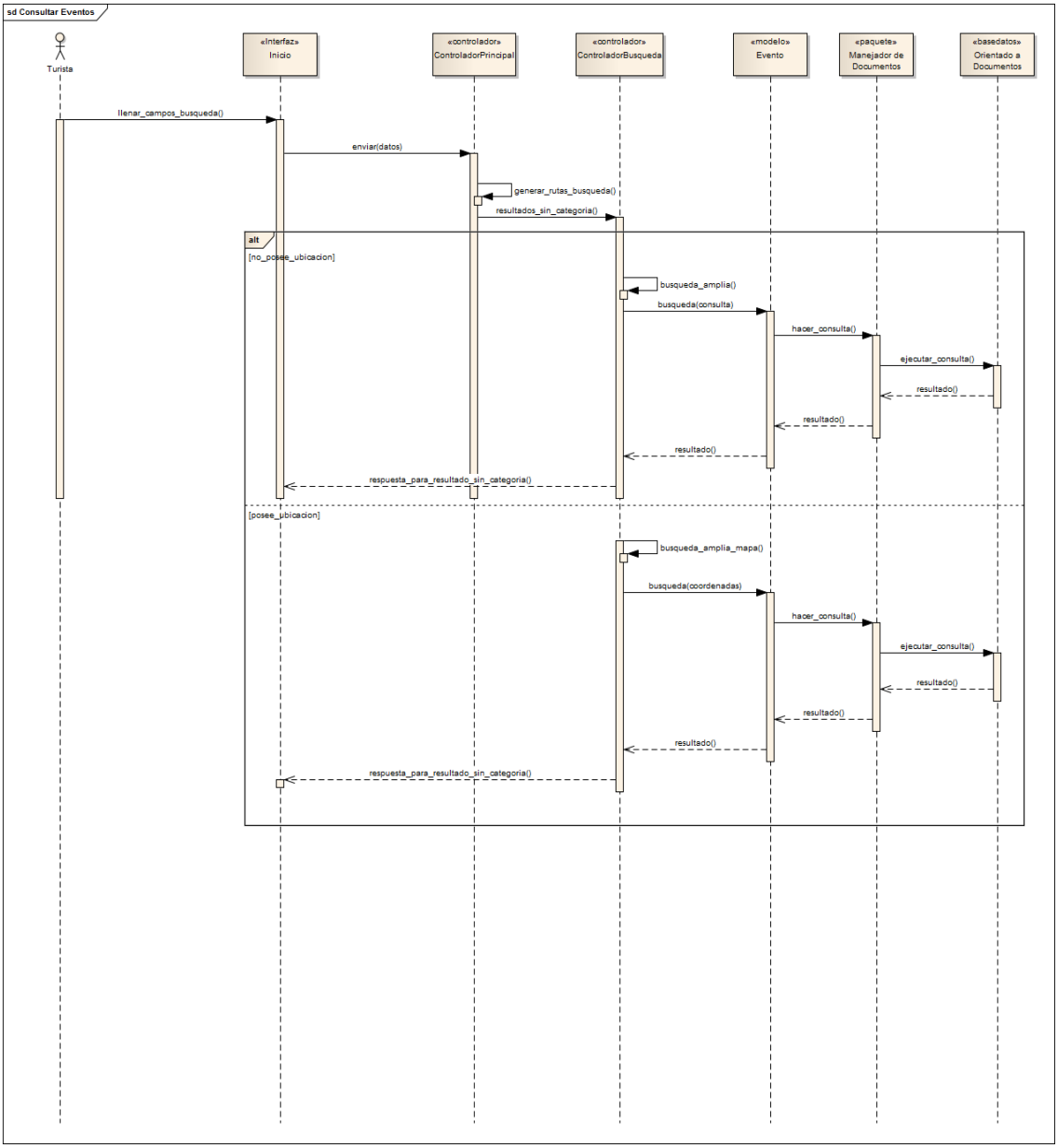


Figura NN. 3. Diagrama de secuencia: Consultar Evento por usuario Turista

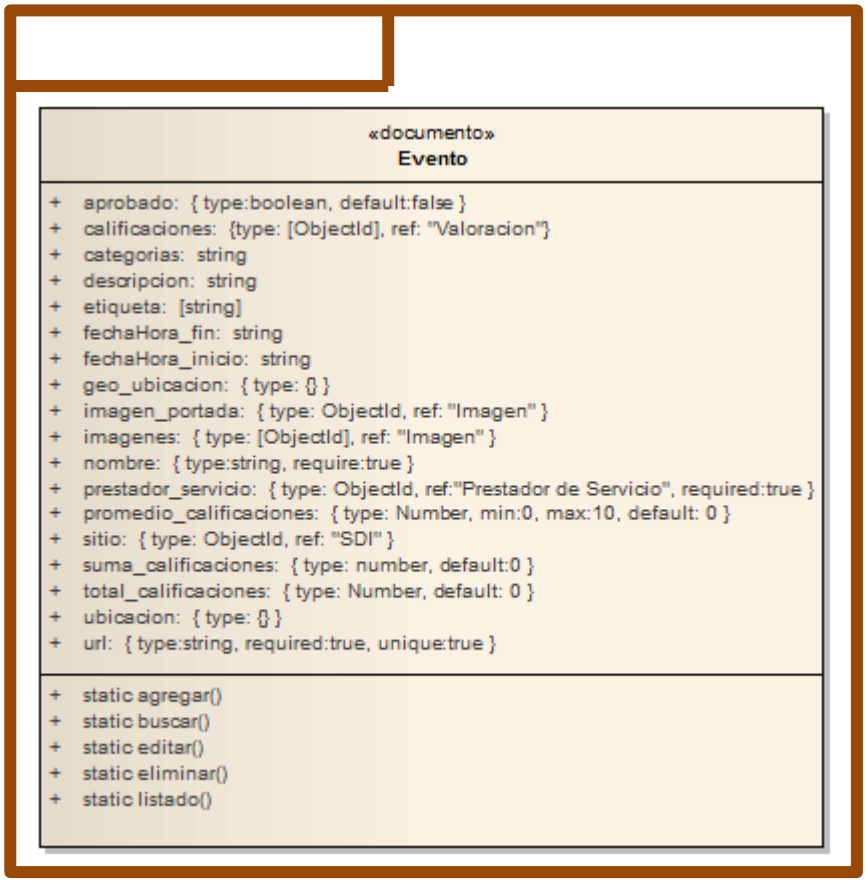


Figura NN. 4. Colección de Evento: Consultar Evento por usuario Turista.

A continuación, se muestran los modelos de plantillas para el proceso de consultar evento por usuario turista.

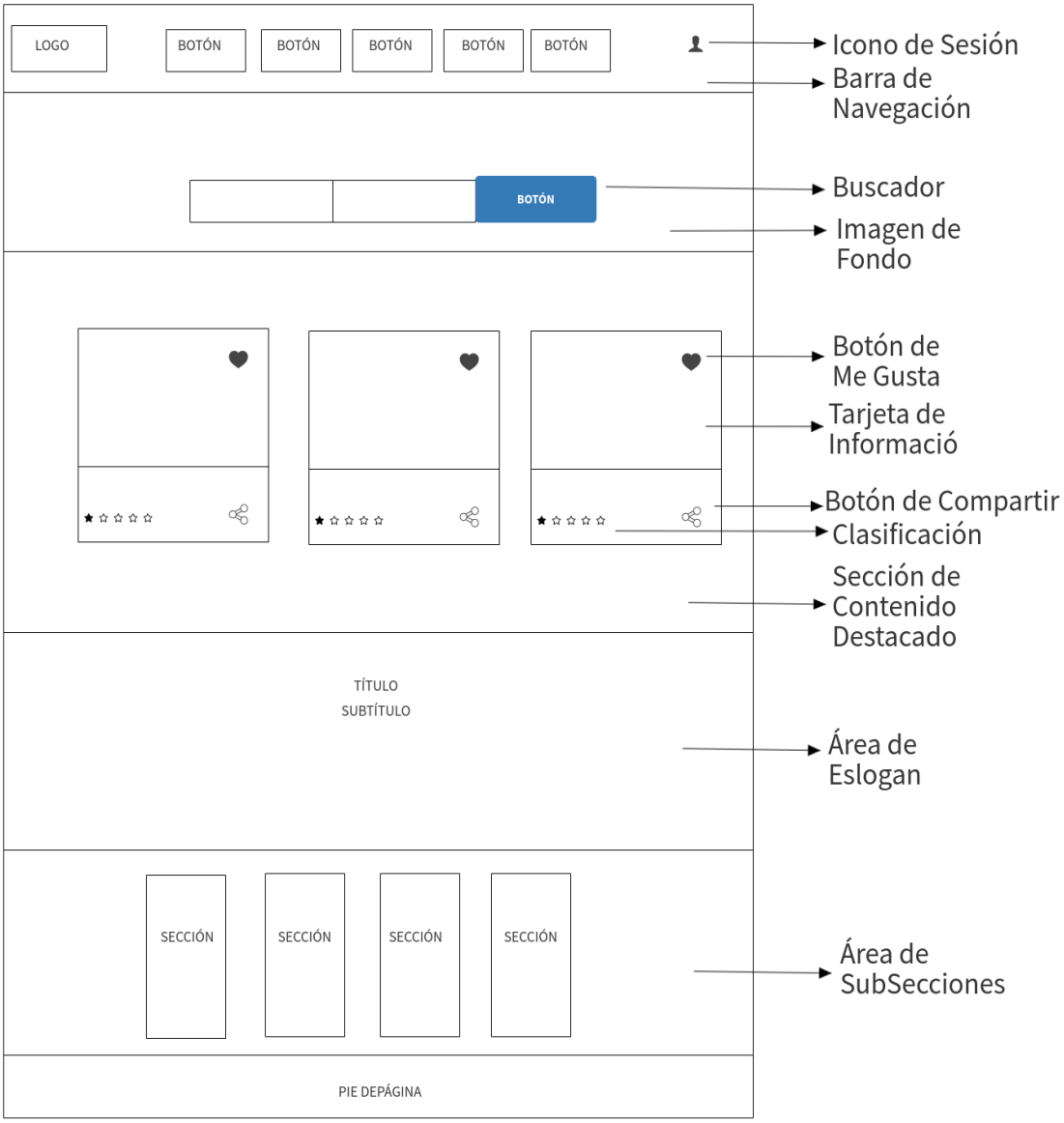


Figura NN. 5. Plantilla de interfaz general: Consultar Evento Turista.

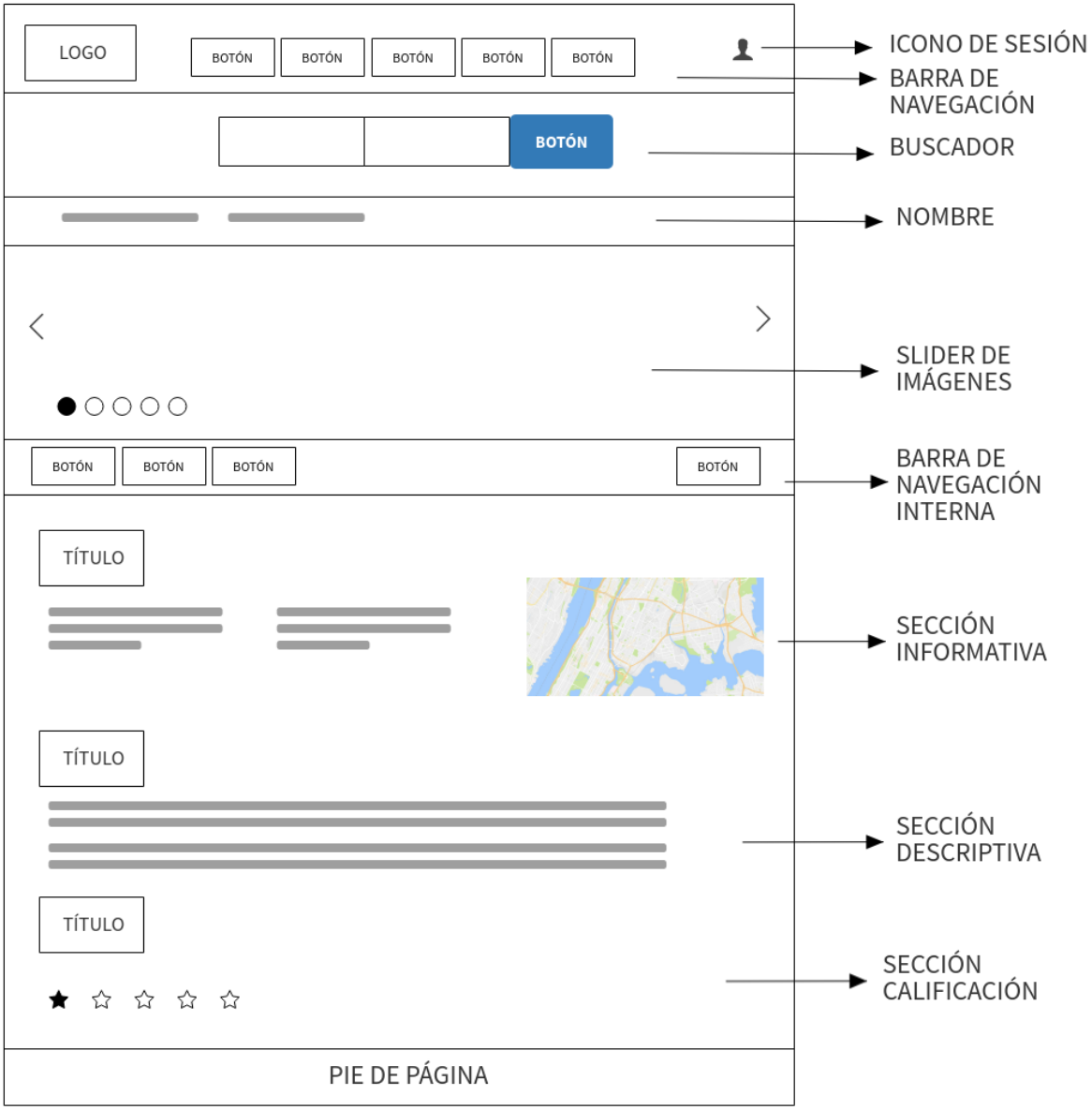


Figura NN. 6. Plantilla de interfaz especifica: Consultar Evento por usuario Turista.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Evento, y luego selecciona un evento en específico y el sistema responde mostrando la interfaz con las especificaciones.

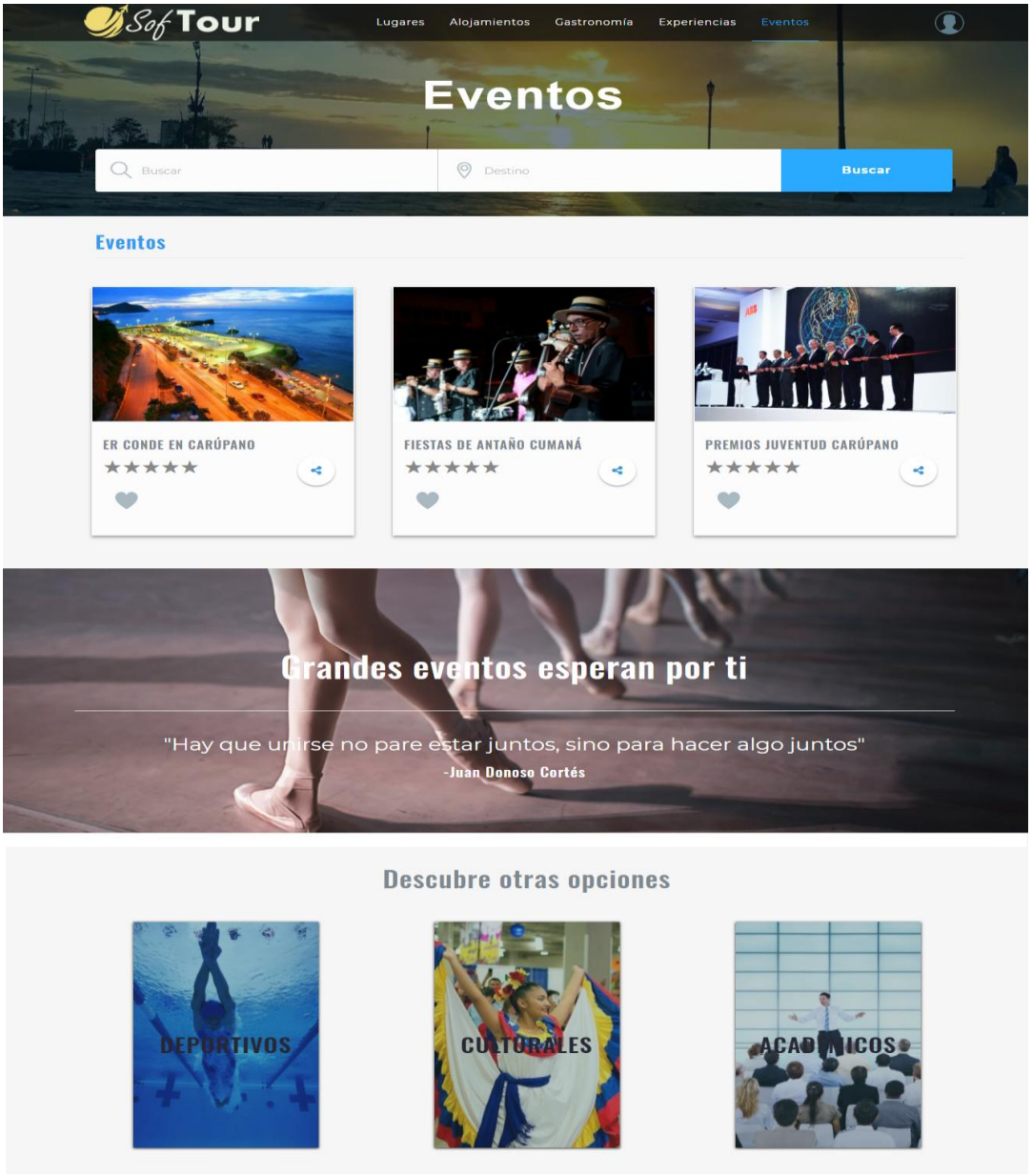


Figura NN. 7. Interfaz 1 de proceso de Consultar Evento por usuario Turista

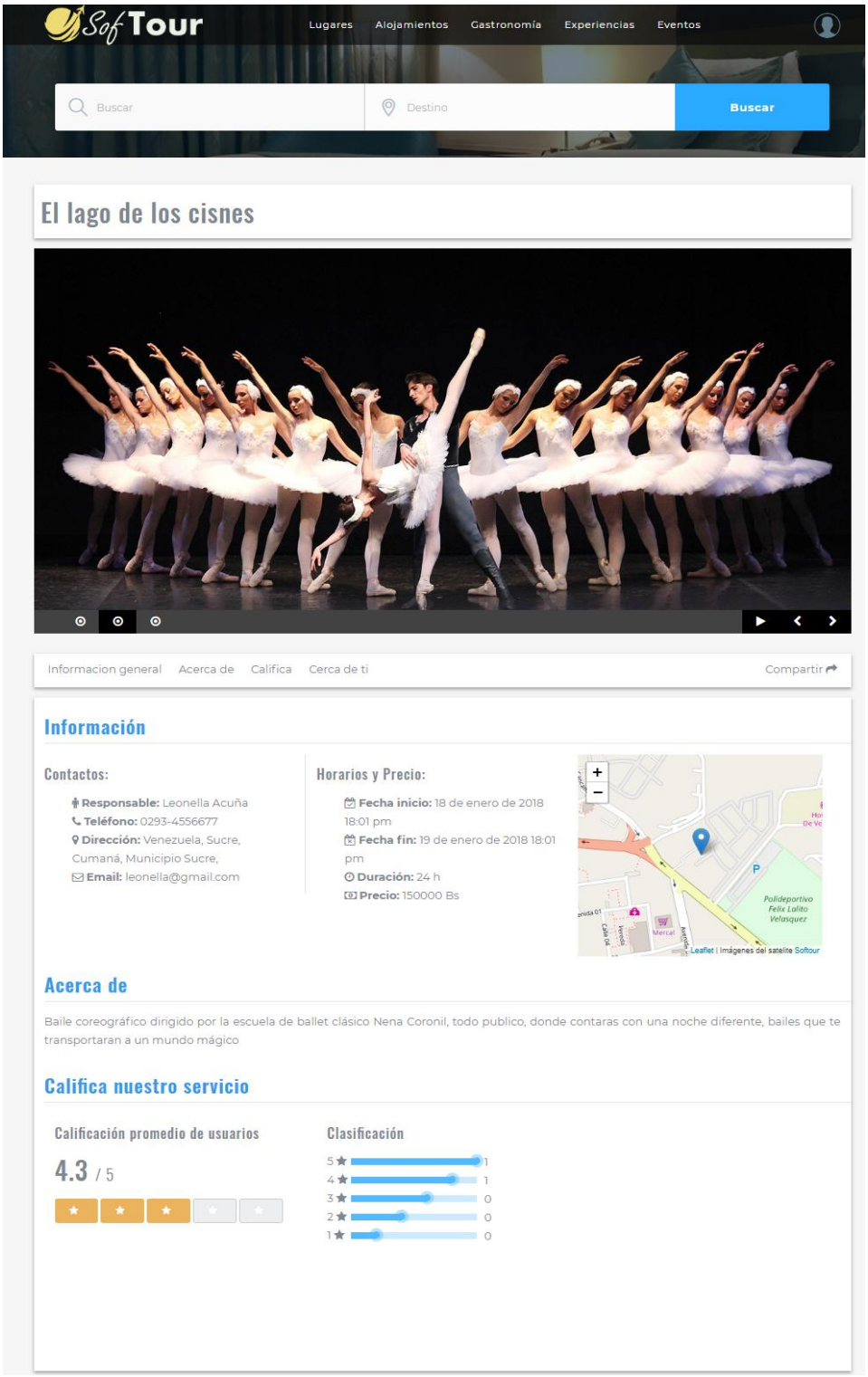


Figura NN. 8. Interfaz 2 de proceso de Consultar Evento por usuario Turista

Código desarrollado para la funcionalidad Consultar Evento por usuario Turista.

```
consultar(req, res, next) {
    this._Evento.buscar_consulta({"url": req.params.url.toLowerCase()}, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": buscado};
        if (req.isAuthenticated())
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_consultar(req, res);
    })
}
```

Figura NN. 9. Código desarrollado para el proceso Consultar Alojamiento Por usuario Turista.

Prueba Consultar Evento

La función a validar es la visualización por parte del usuario turista al momento de seleccionar el apartado de eventos.

Para documentación de la prueba se muestran imágenes realizadas durante la prueba:

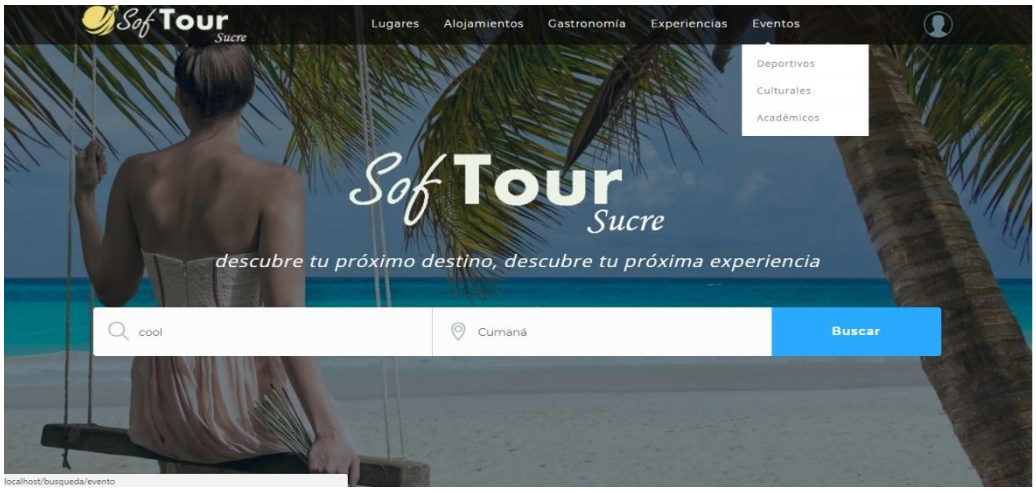


Figura NN. 10. Prueba Consultar Eventos por Turista. Paso 1

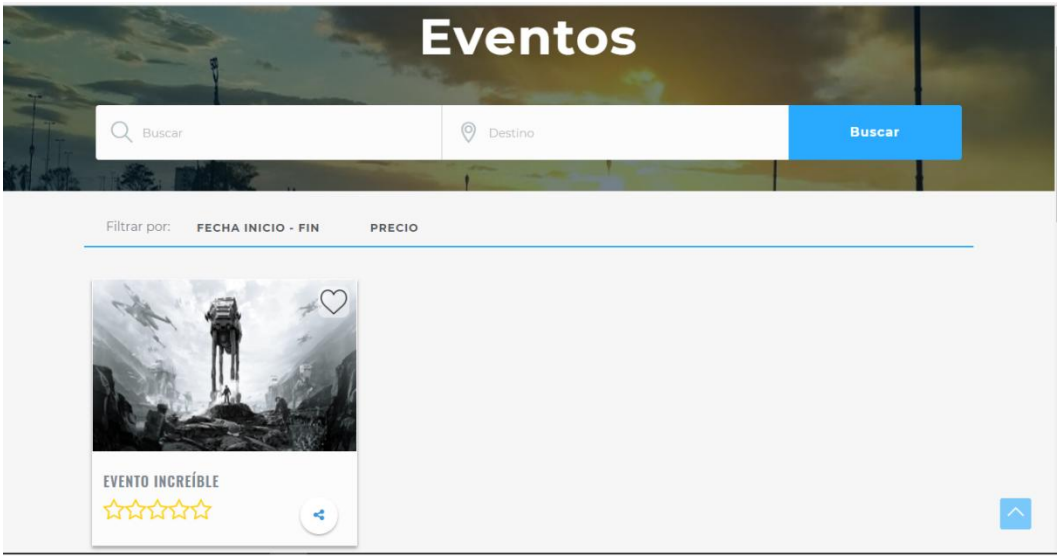


Figura NN. 11. Prueba Consultar Eventos por Turista. Paso 2.

Conclusión de la prueba

La función cumple con el objetivo para el cual fue diseñada.

Prueba de interfaz y usabilidad

Mediante encuesta previa se determinó que la interfaz se encuentra optima ara su interacción.

Apéndice OO. Consultar Experiencia por Turista.

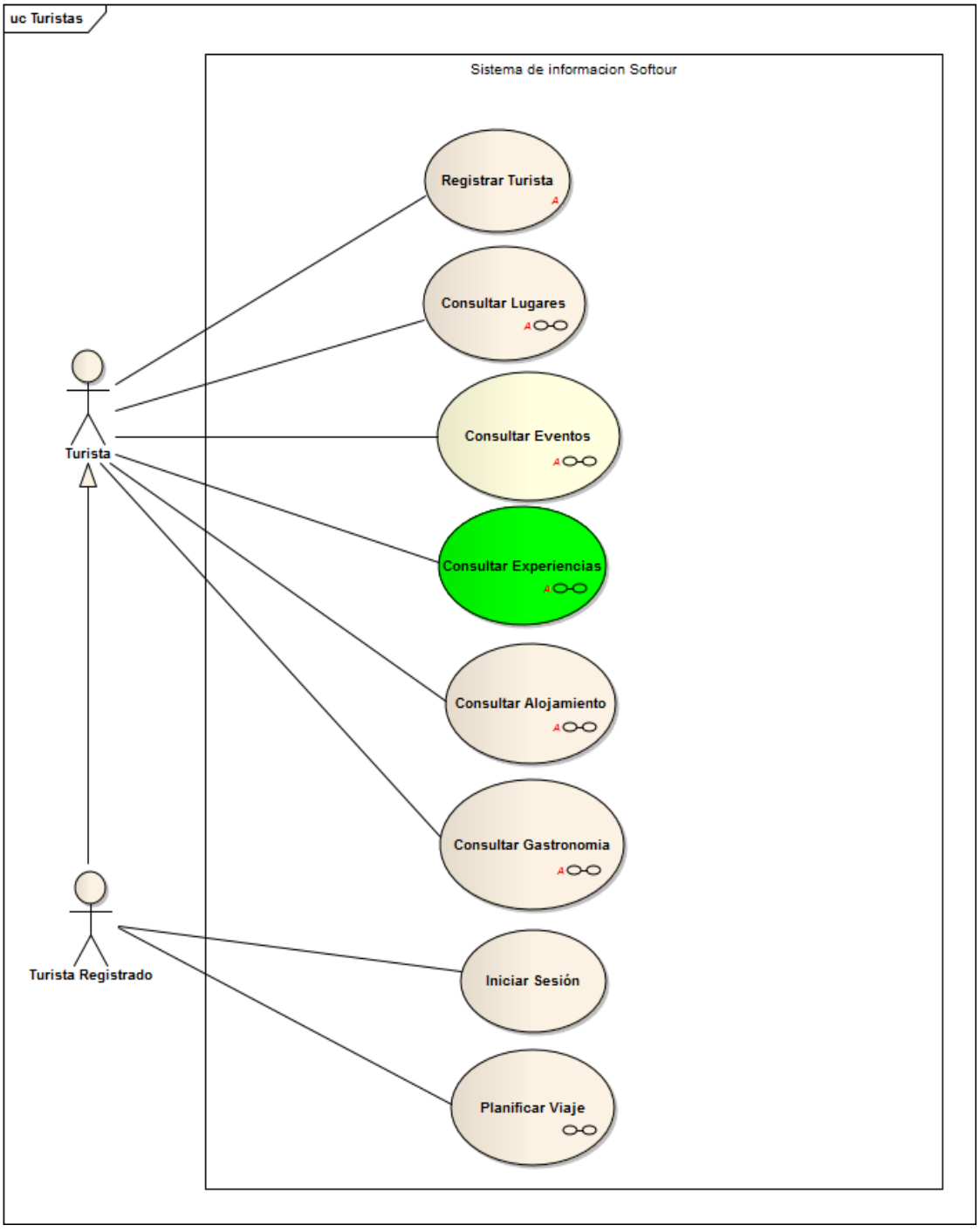


Figura OO. 1. Caso de uso: Consultar Alojamiento por usuario Turista.

Tabla OO. 1. Descripción Caso de uso Consultar Alojamiento por usuario Turista.

Nombre del Caso de uso	Consultar Experiencias	
Versión	1.0 03/11/2017	
Actor	Turista.	
Descripción del Caso de Uso	El caso de uso consiste en consultar las experiencias desde una perspectiva visual, organizada y llamativa. Permitiendo enlazar el caso de uso con patrones de búsqueda específicos relacionados con lo que desea el turista.	
Precondición	Ninguna	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El caso de uso comienza cuando el Turista hace clic en el menú en la categoría "Experiencia" o alguna subcategoría.
	2	El sistema despliega las experiencias disponibles y muestra los posibles filtros que aplican para refinar aún más la búsqueda, acordes a lo que desea el turista. Esto se realiza extendiendo <<extend>> la funcionalidad de la búsqueda con el caso de uso: a.- Administrar patrones de búsqueda de experiencias.
	3	El Turista visualiza los alojamientos disponibles que se muestran y selecciona uno de ellos para ver el detalle del mismo.
	4	El sistema muestra el detalle de la experiencia seleccionada.
	5	El Turista visualiza los atributos de la experiencia.
	6	El sistema cierra el caso de uso y retorna el control.
Postcondición	Se consulta la experiencia desde la perspectiva del turista, satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Turista decide no consultar la experiencia y selecciona cancelar el caso de uso (selecciona el botón cerrar o el botón cancelar).
	4.1.	El sistema finaliza el caso de uso.
Reglas del negocio	3. Las consultas de las experiencias que incluyan filtros o patrones de búsquedas deben considerarse extensiones de este caso de uso. 4. Al seleccionar una experiencia, el turista podrá visualizar mayor detalle del mismo.	
Inclusiones	Ninguna	
Extensiones	2. Administrar patrones de búsqueda de experiencias	

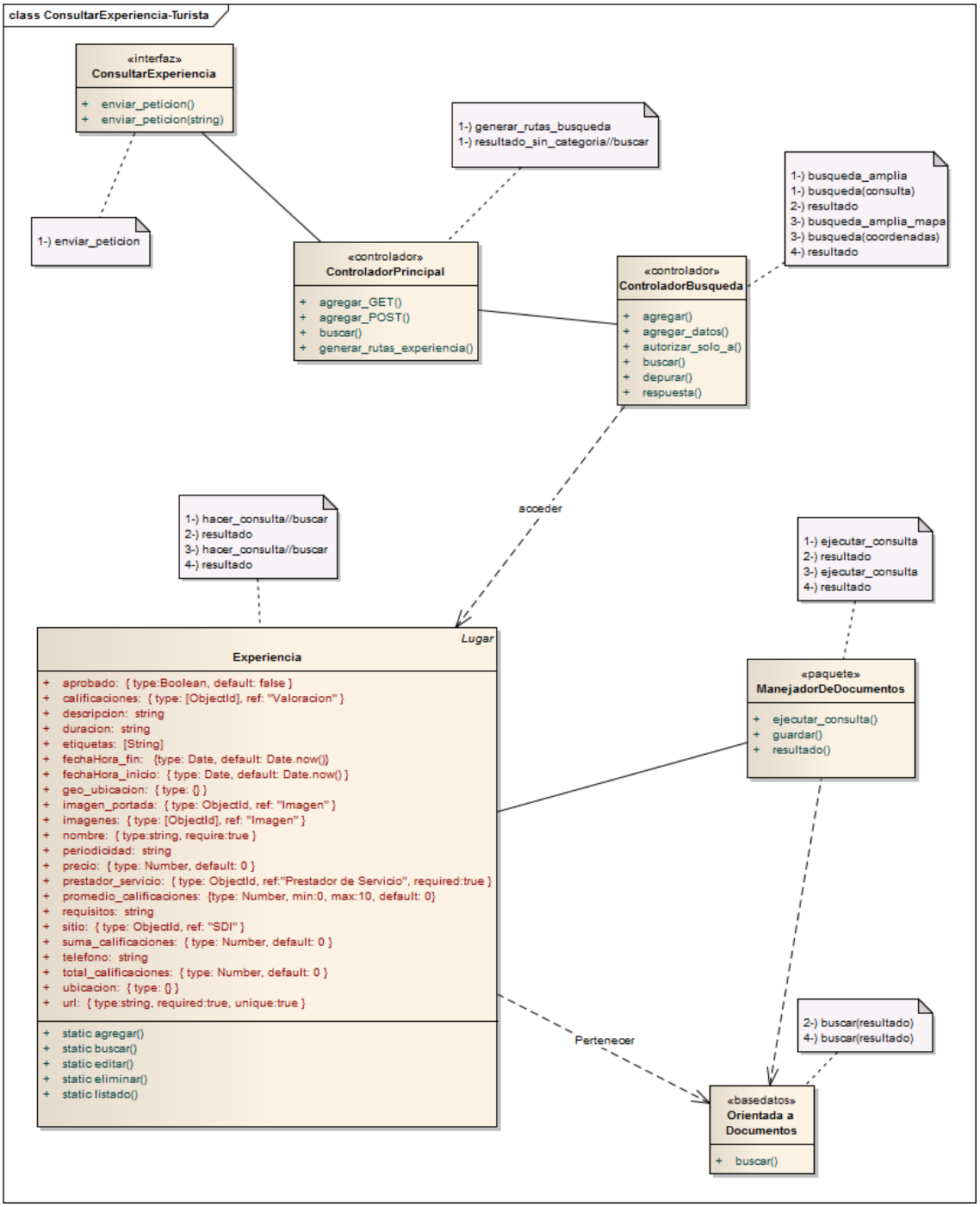


Figura OO. 2. Diagrama de Clases: Consultar Experiencia por Usuario Turista.

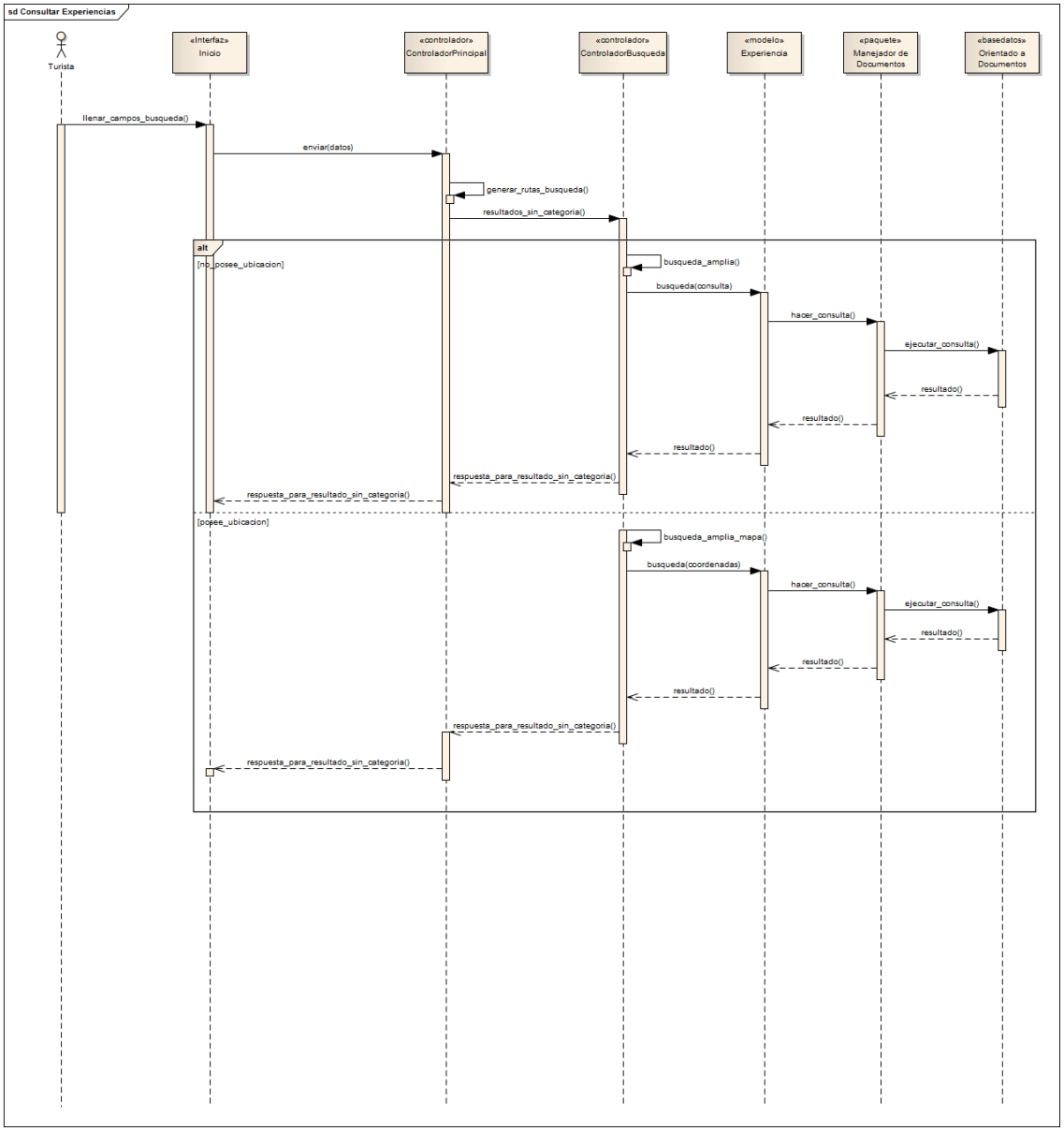


Figura OO. 3. Diagrama de secuencia: Consultar Experiencia por usuario Turista.

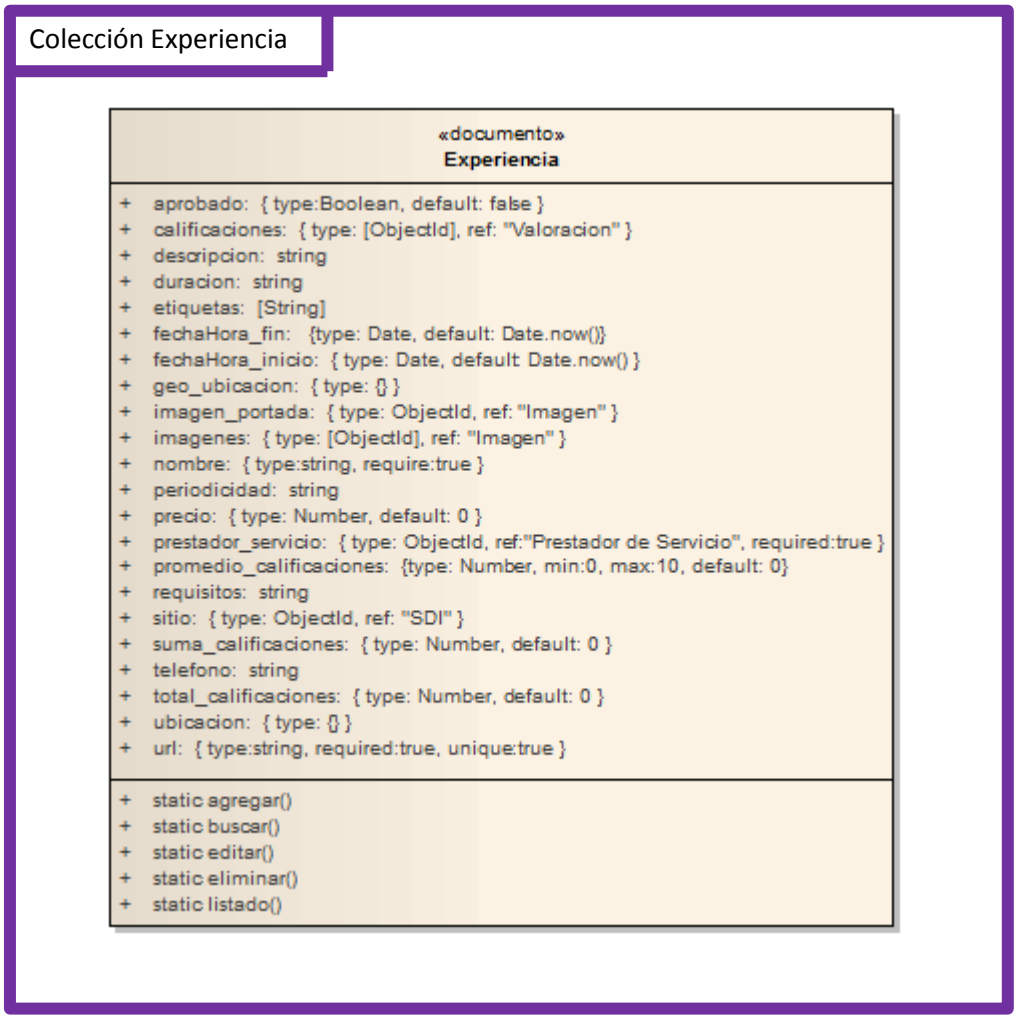


Figura OO. 4. Colección de Experiencia: Consultar experiencia por usuario Turista.

A continuación, se muestran los modelos de plantillas para el proceso de consultar experiencia por usuario turista.

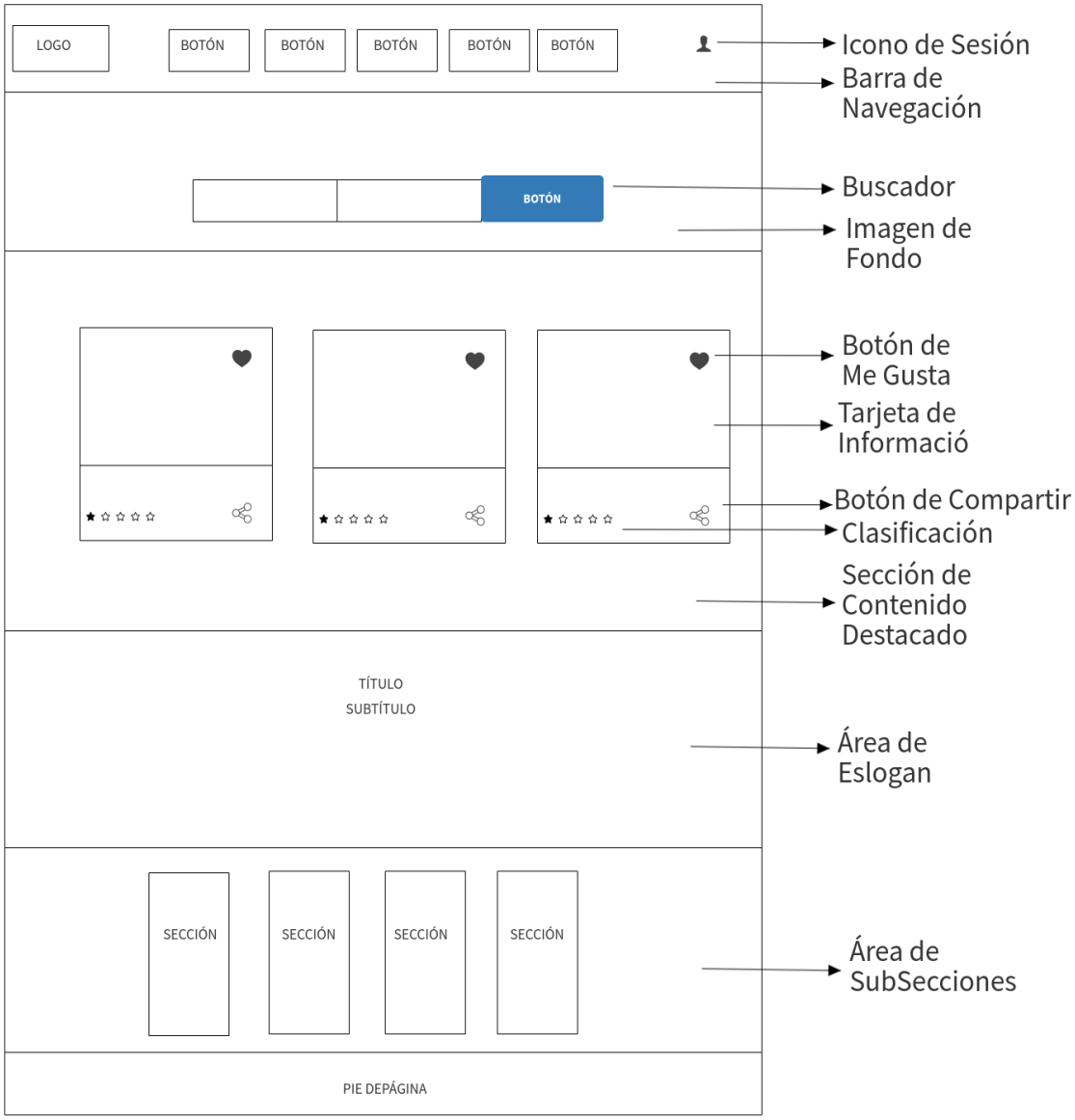


Figura OO. 5. Plantilla de interfaz general: Consultar Experiencia por usuario Turista.

Figura OO. 6. Plantilla de interfaz especifica: Consultar Experiencia por usuario Turista.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Experiencia, y luego selecciona una experiencia en específica y el sistema responde mostrando la interfaz con las especificaciones.

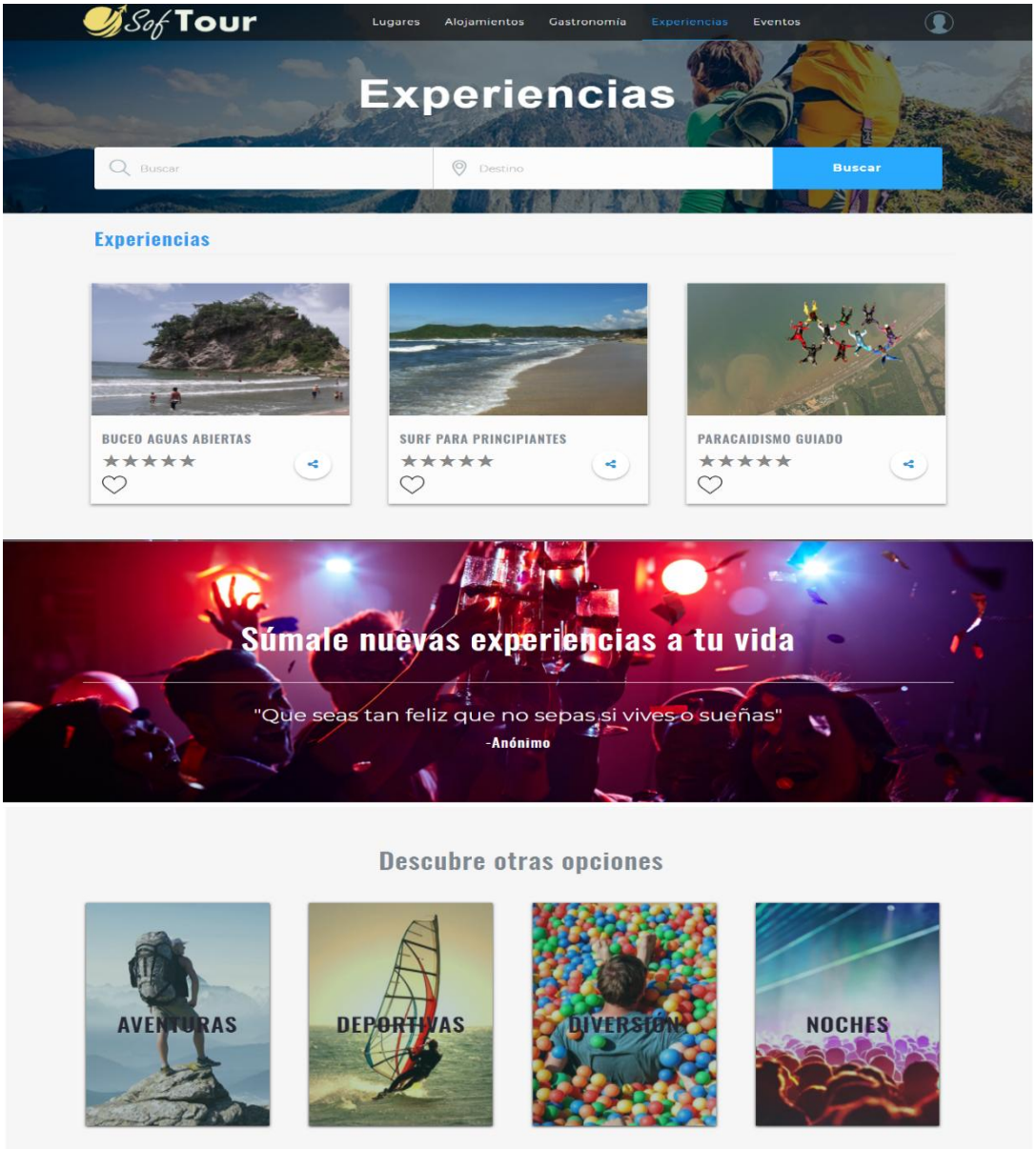


Figura OO. 7. Interfaz 1 de proceso de Consultar Experiencia por usuario Turista

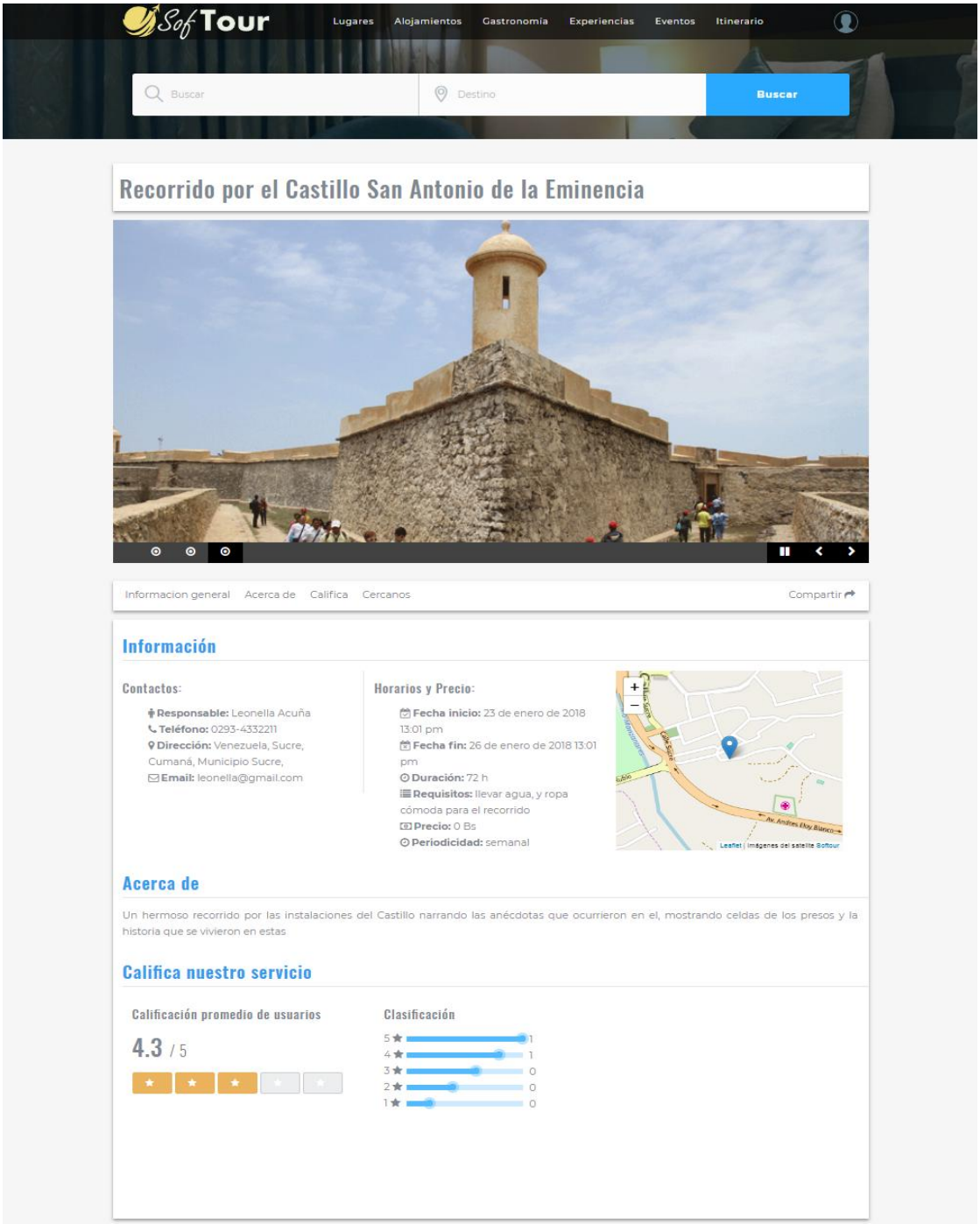


Figura OO. 8. Interfaz 2 de proceso de Consultar Experiencia por usuario Turista

Código desarrollado para la funcionalidad Consultar Experiencia por usuario Turista.

```
consultar(req, res, next) {
    this._Experiencia.buscar_consulta({"url": req.params.url.toLowerCase()}, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        res.locals.datos = {"value": buscado};
        if (req.isAuthenticated())
            res.locals.datos.value.usuario_sesionado = req.session.passport.user;
        this._respuesta.para_consultar(req, res);
    })
}
```

Figura OO. 9. Código desarrollado para el proceso Consultar Experiencia Por usuario Turista.

Prueba Consultar Experiencia

La función a validar es la selección y muestra de la sección experiencias desde el perfil turista.

A continuación, se muestra secuencias de imágenes durante la prueba:

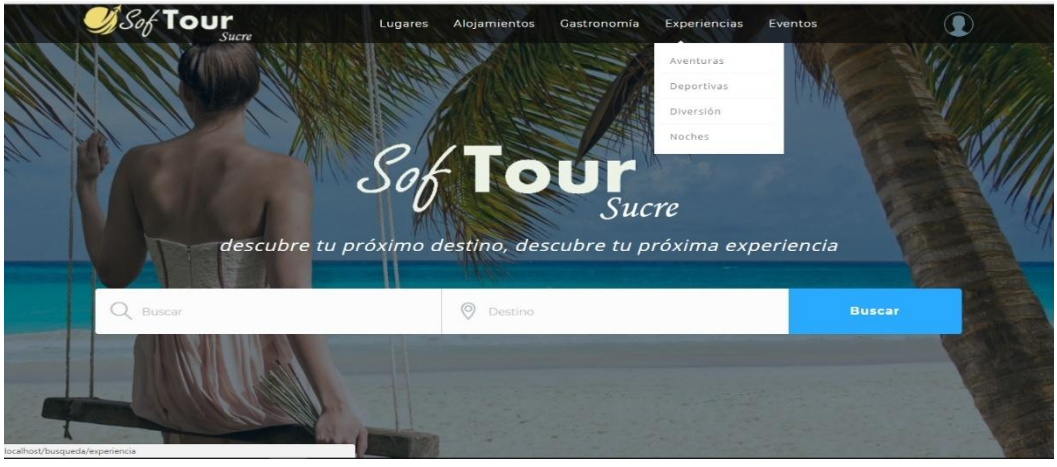


Figura OO. 10. Prueba consultar experiencias por usuario Turista. Paso 1

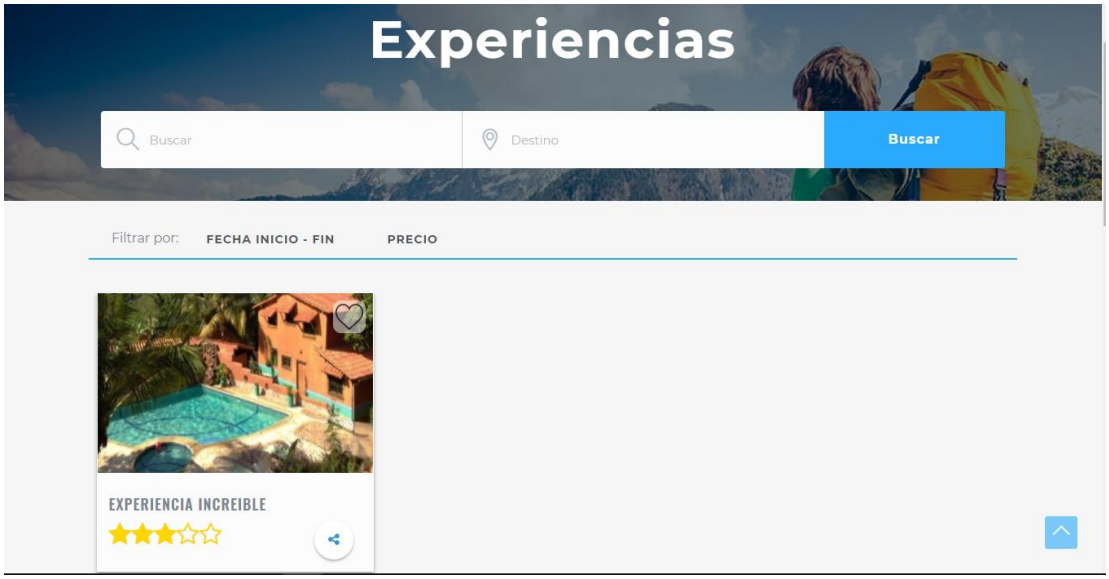


Figura OO. 11. Prueba consultar experiencia por turista. Paso 2

Conclusión de la prueba

La función se ejecuta de forma correcta.

Prueba de interfaz y usabilidad

Durante encuesta realizada previamente se determinó que la interfaz cumple con estándares de usabilidad.

Apéndice PP. Consultar Gastronomía por Turista

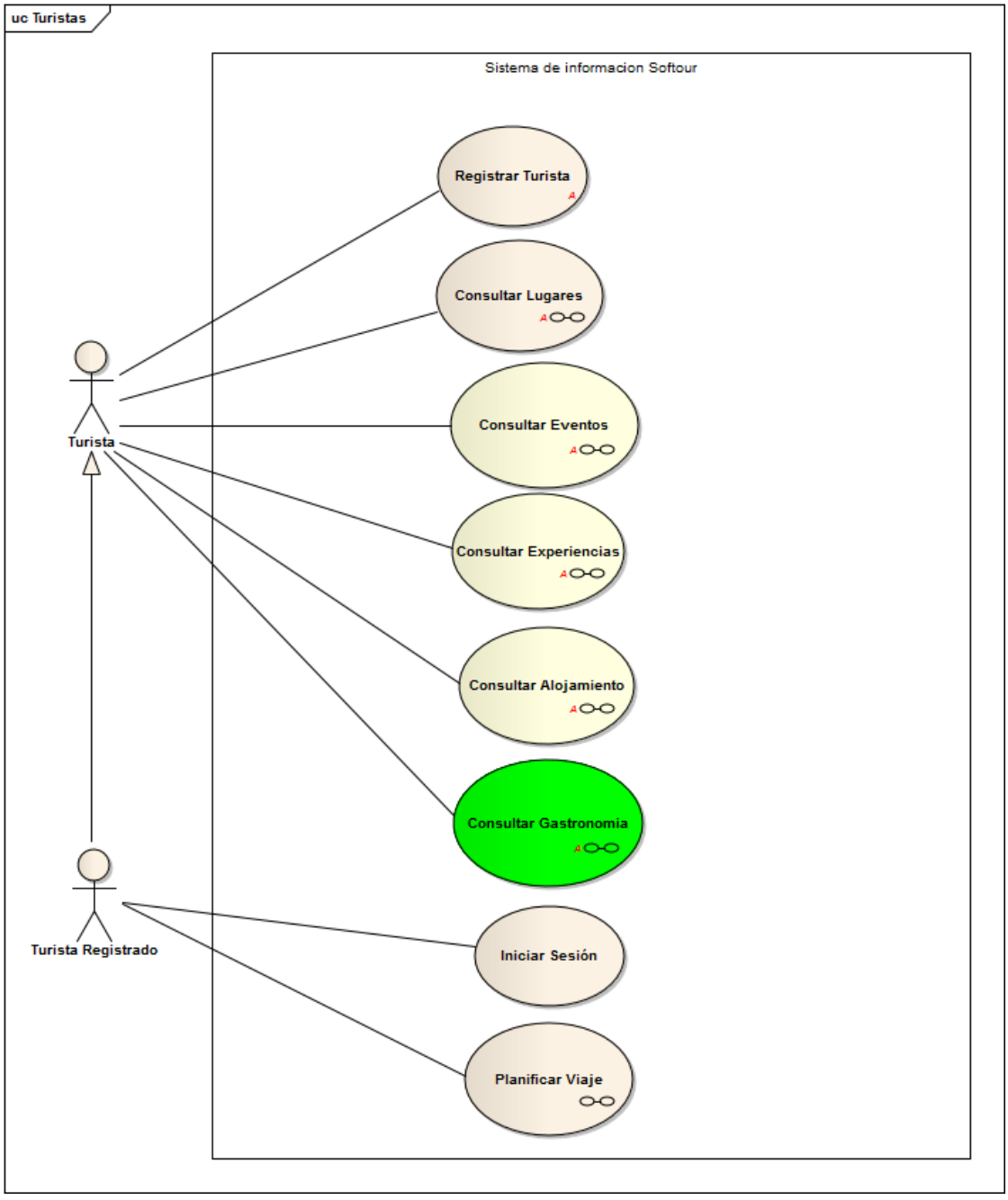


Figura PP. 1. Caso de uso: Consultar Gastronomía por usuario Turista.

Tabla PP. 1. Descripción Caso de uso Consultar Gastronomía por usuario Turista.

Nombre del Caso de uso	Consultar Gastronomía	
Versión	1.0 03/11/2017	
Actor	Turista.	
Descripción del Caso de Uso	El caso de uso consiste en consultar la gastronomía disponible desde una perspectiva visual, organizada y llamativa. Permitiendo enlazar el caso de uso con patrones de búsqueda específicos relacionados con lo que desea el turista.	
Precondición	Ninguna	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El caso de uso comienza cuando el Turista selecciona la opción "Consultar Gastronomía" desde el perfil del turista.
	2	El sistema despliega la gastronomía disponible a la fecha que cumplan un orden preestablecido (como por ejemplo: más visitados, mejores calificados, promocionados por prestadores de servicios, entre otras). El sistema posibilita hacer búsquedas más específicas, acordes a lo que desea el turista. Esto se realiza extendiendo <<extend>> la funcionalidad de la búsqueda con el caso de uso: a.- Administrar patrones de búsqueda de gastronomías.
	3	El Turista visualiza la gastronomía disponible que se muestra y selecciona una de ellas para ver el detalle de la misma.
	4	El sistema muestra el detalle de la gastronomía seleccionada.
	5	El Turista visualiza el detalle de la gastronomía y presiona aceptar.
	6	El sistema cierra el caso de uso y retorna el control.
Postcondición	Se consulta la gastronomía disponible desde la perspectiva del turista, satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Turista decide no consultar la gastronomía y selecciona cancelar el caso de uso (selecciona el botón cerrar o el botón cancelar).
	4.1.	El sistema finaliza el caso de uso.
Reglas del negocio	1. Las consultas a la gastronomía que aparecen en un primer plano deben considerar algún orden de prioritario de búsqueda y aparición, como por ejemplo: los más visitados, los mejores calificados, los promocionados por los prestadores de servicios. 2. Las consultas de la gastronomía en los que se incluyan filtros o patrones de búsquedas deben considerarse extensiones de este caso de uso. 3. Al seleccionar una gastronomía, el turista podrá visualizar mayor detalle del mismo.	
Inclusiones	Ninguna	
Extensiones	1. Administrar patrones de búsqueda de gastronomías.	

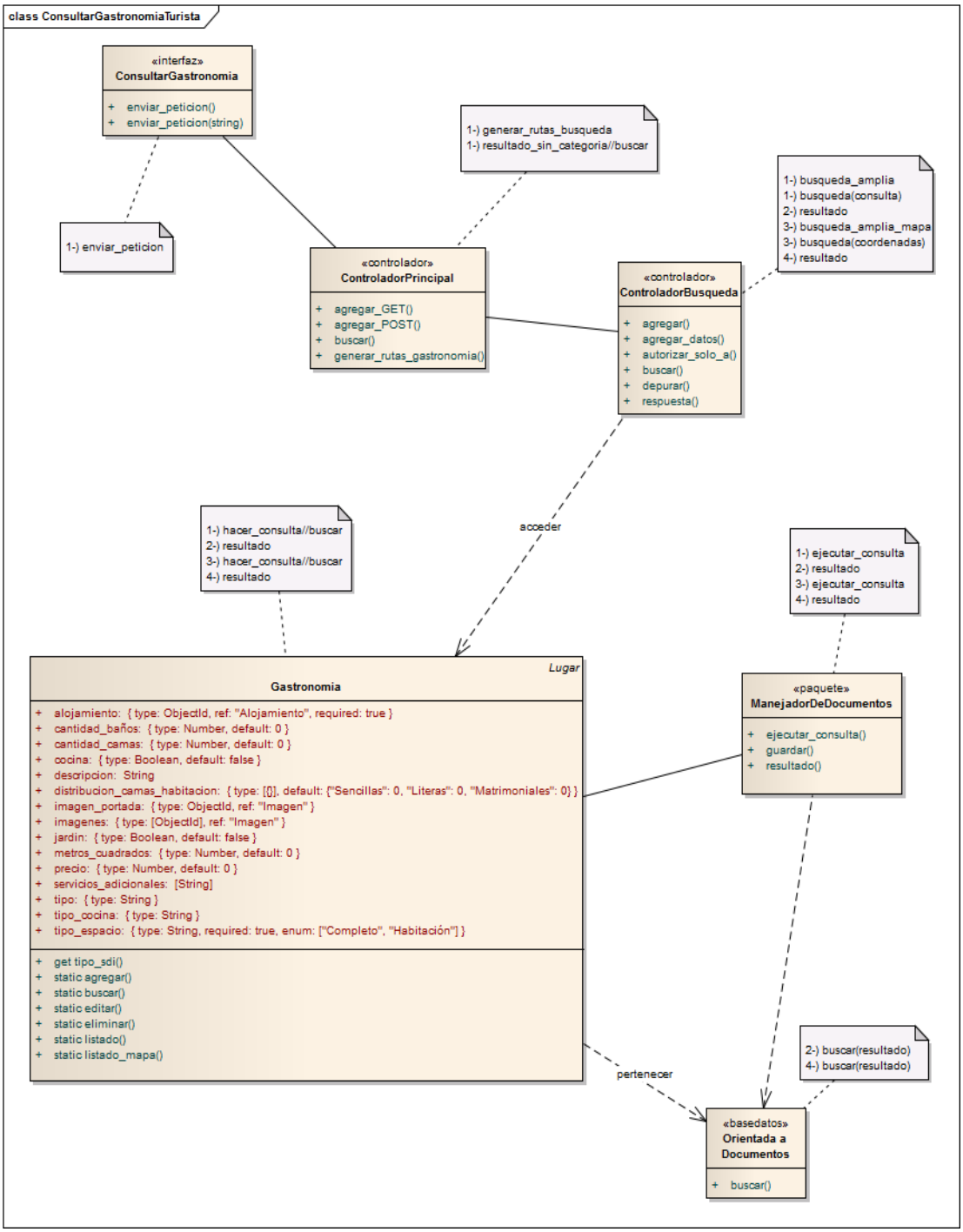


Figura PP. 2. Diagrama de Clases: Consultar Gastronomía por Usuario Turista.

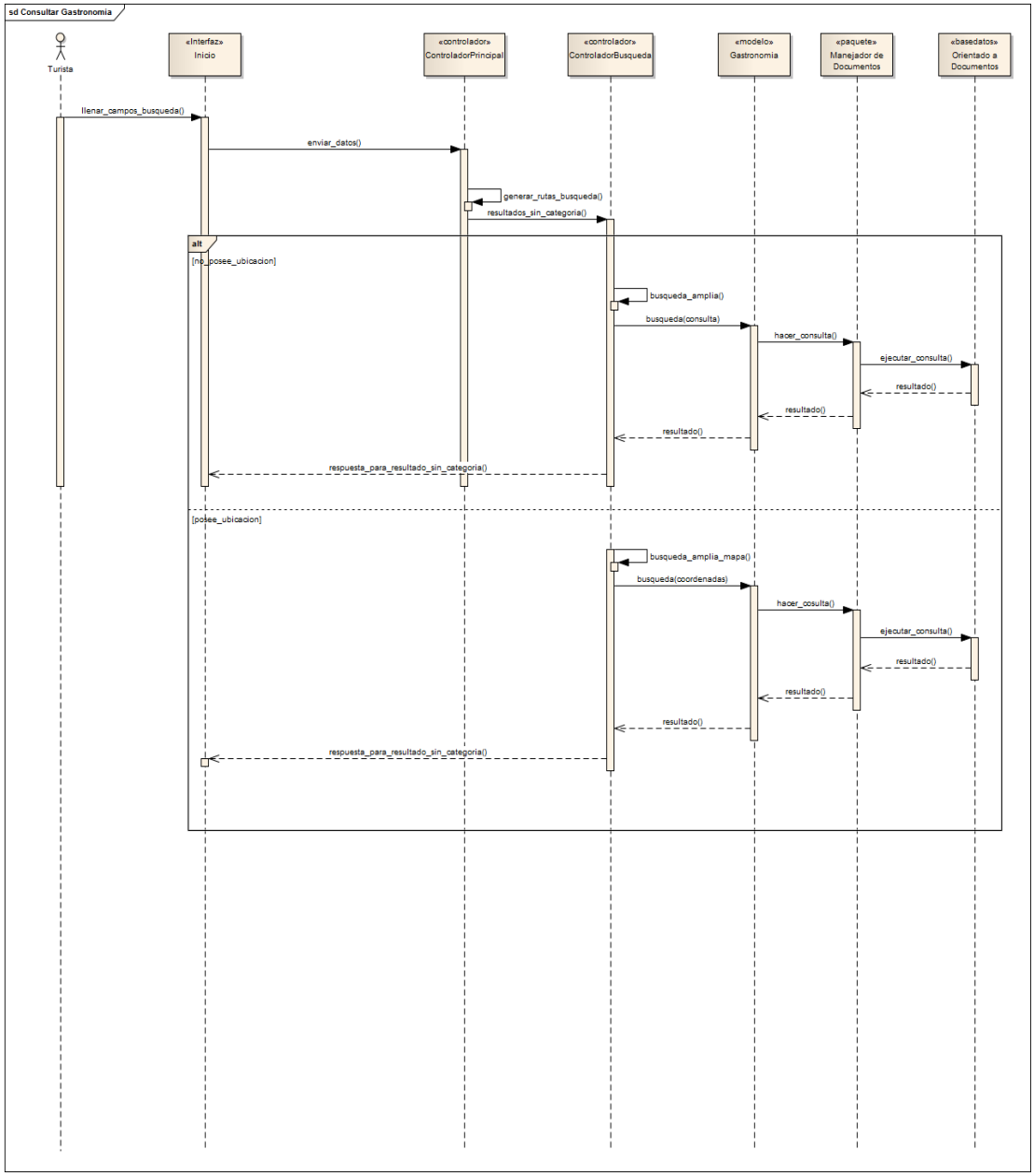


Figura PP. 3. Diagrama de secuencia: Consultar Gastronomía por usuario Turista

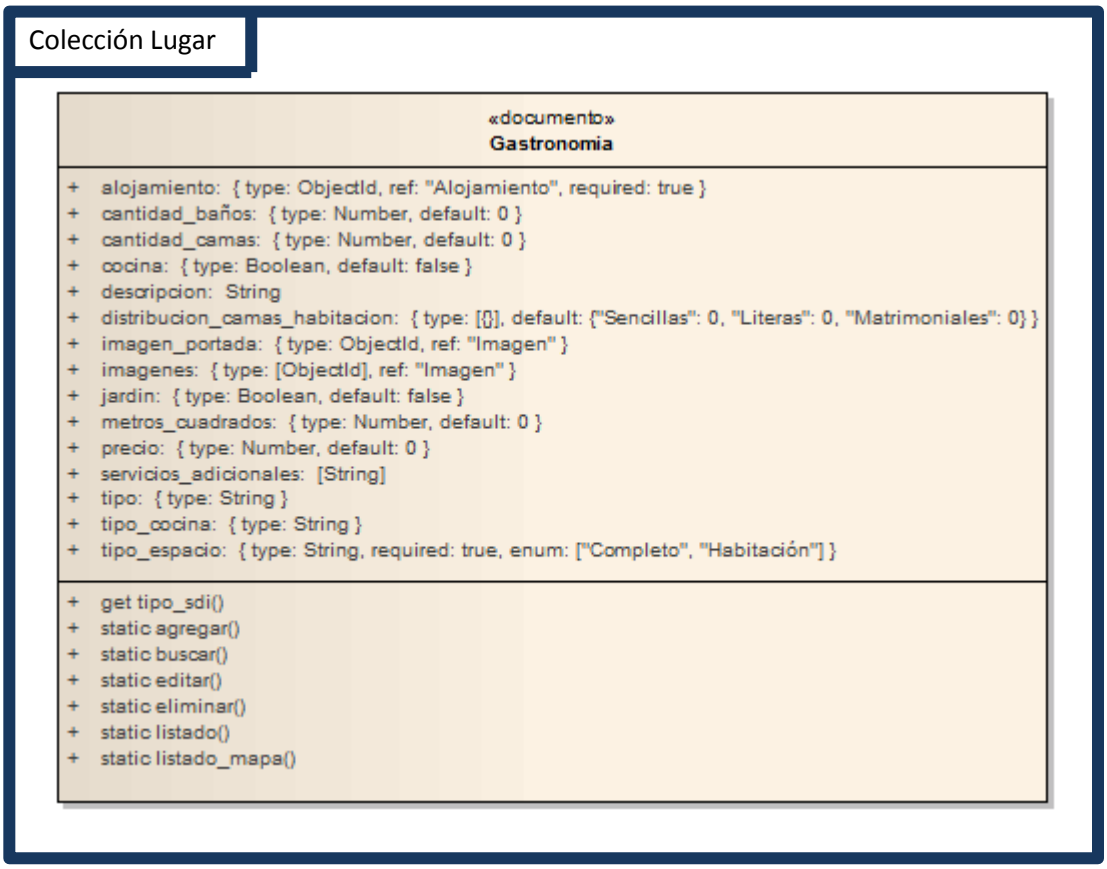


Figura PP. 4. Colección Gastronomía: Consultar Gastronomía por usuario Turista.

A continuación, se muestran los modelos de plantillas para el proceso de consultar gastronomía por usuario turista.

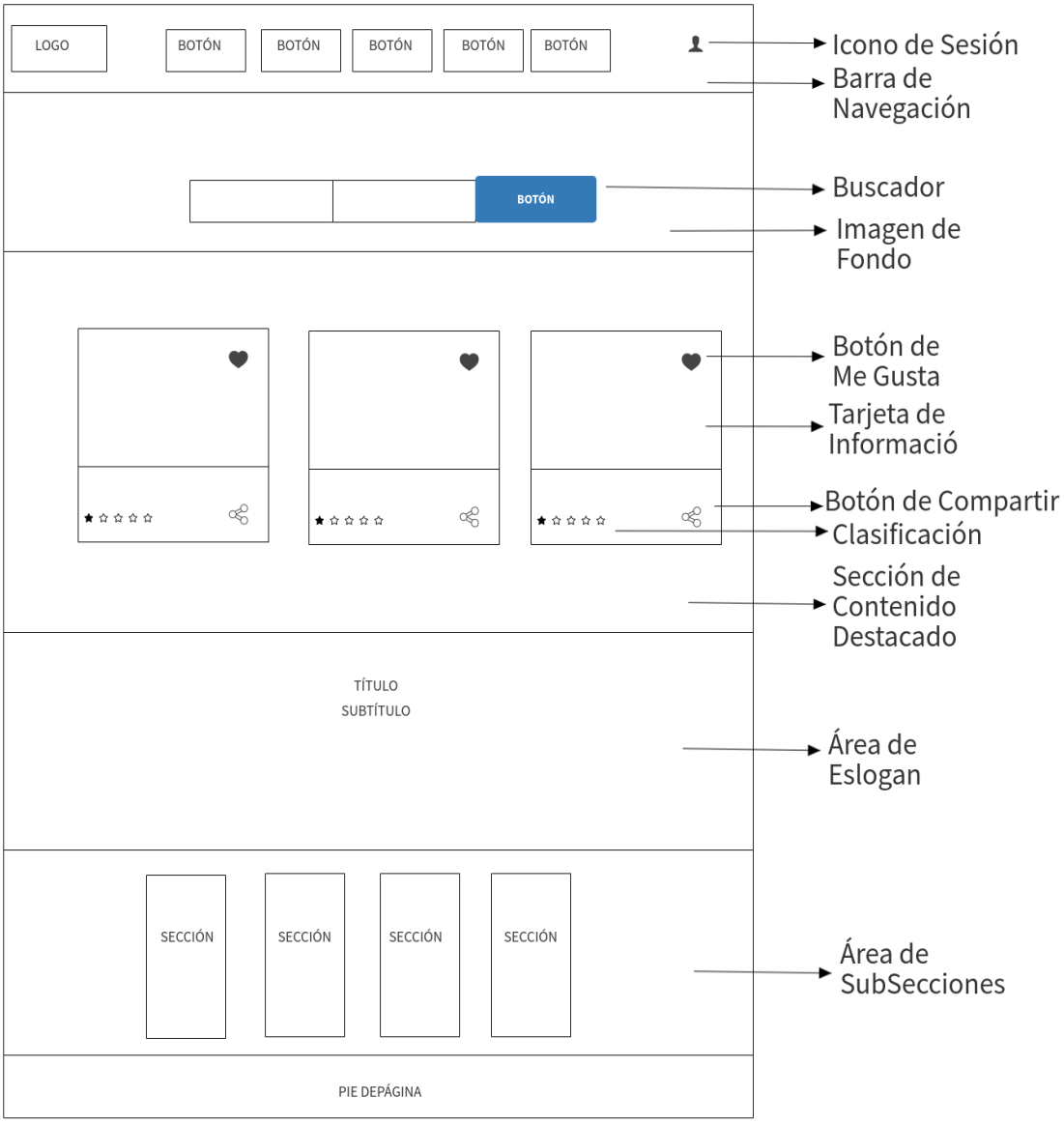


Figura PP. 5. Plantilla de interfaz general: Consultar Gastronomía por usuario Turista.

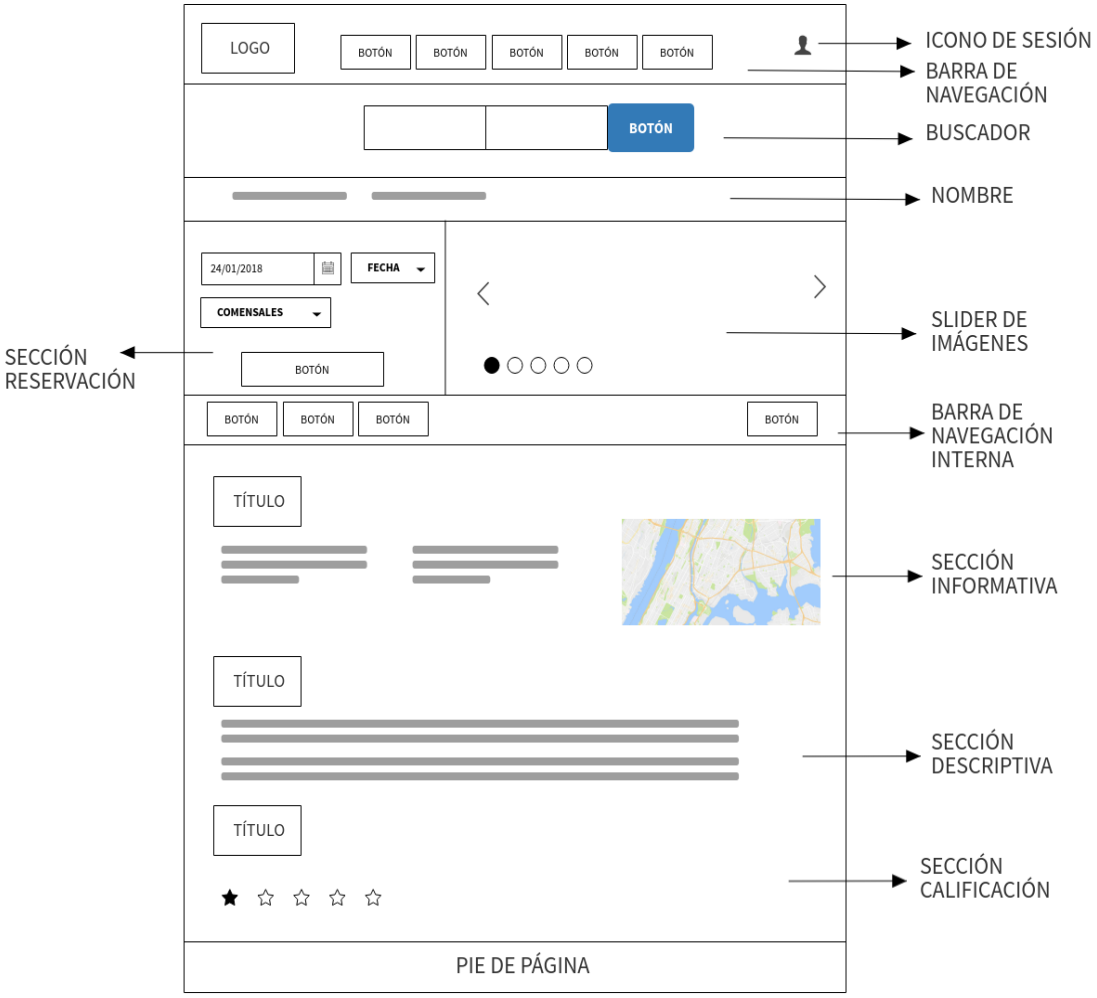


Figura PP. 6. Plantilla de interfaz especifica: Consultar Gastronomía por usuario Turista.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Gastronomía, y luego selecciona un apartado en específico y el sistema responde mostrando la interfaz con las especificaciones.

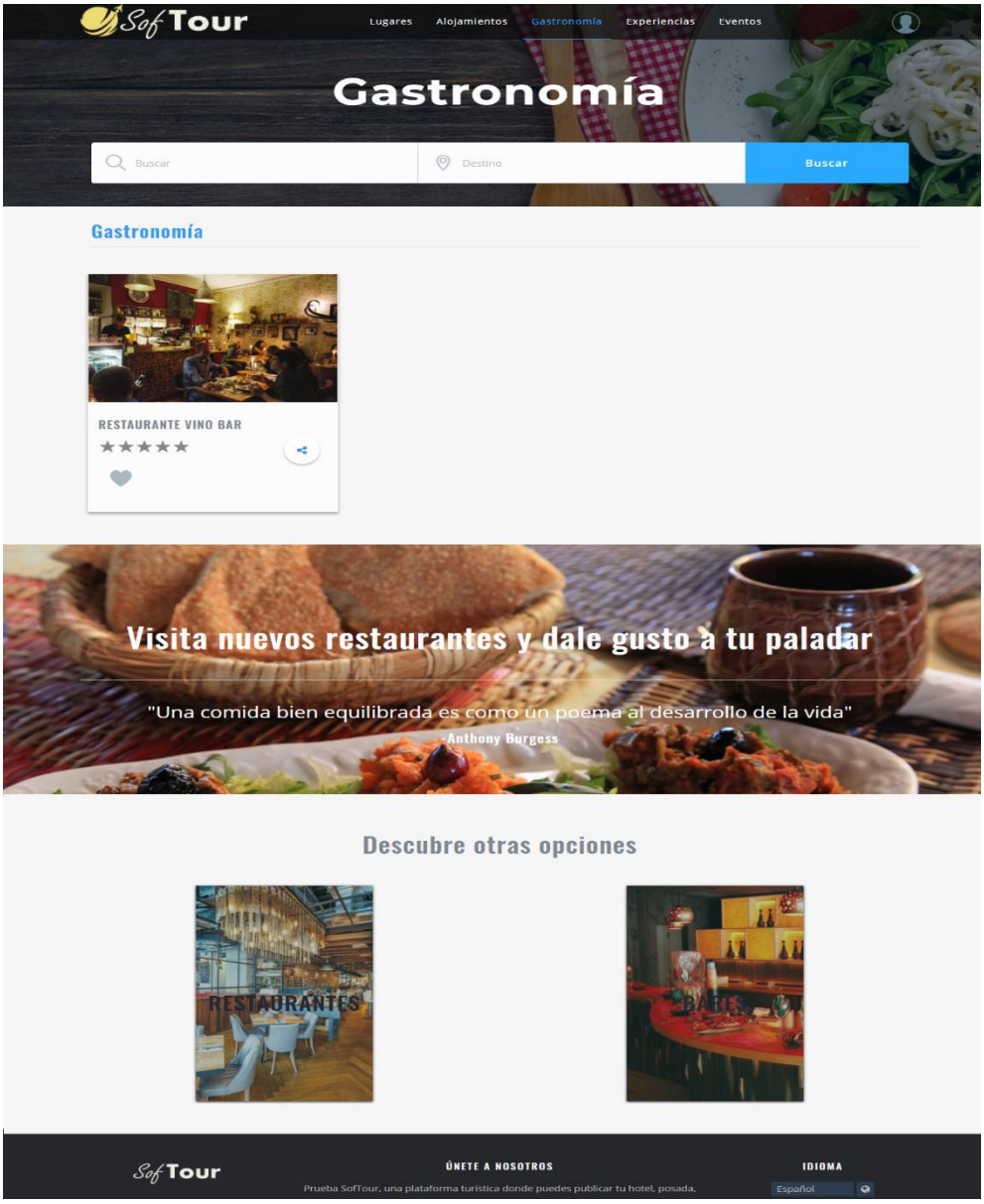


Figura PP. 7. Interfaz 1 de proceso de Consultar Gastronomía por usuario Turista

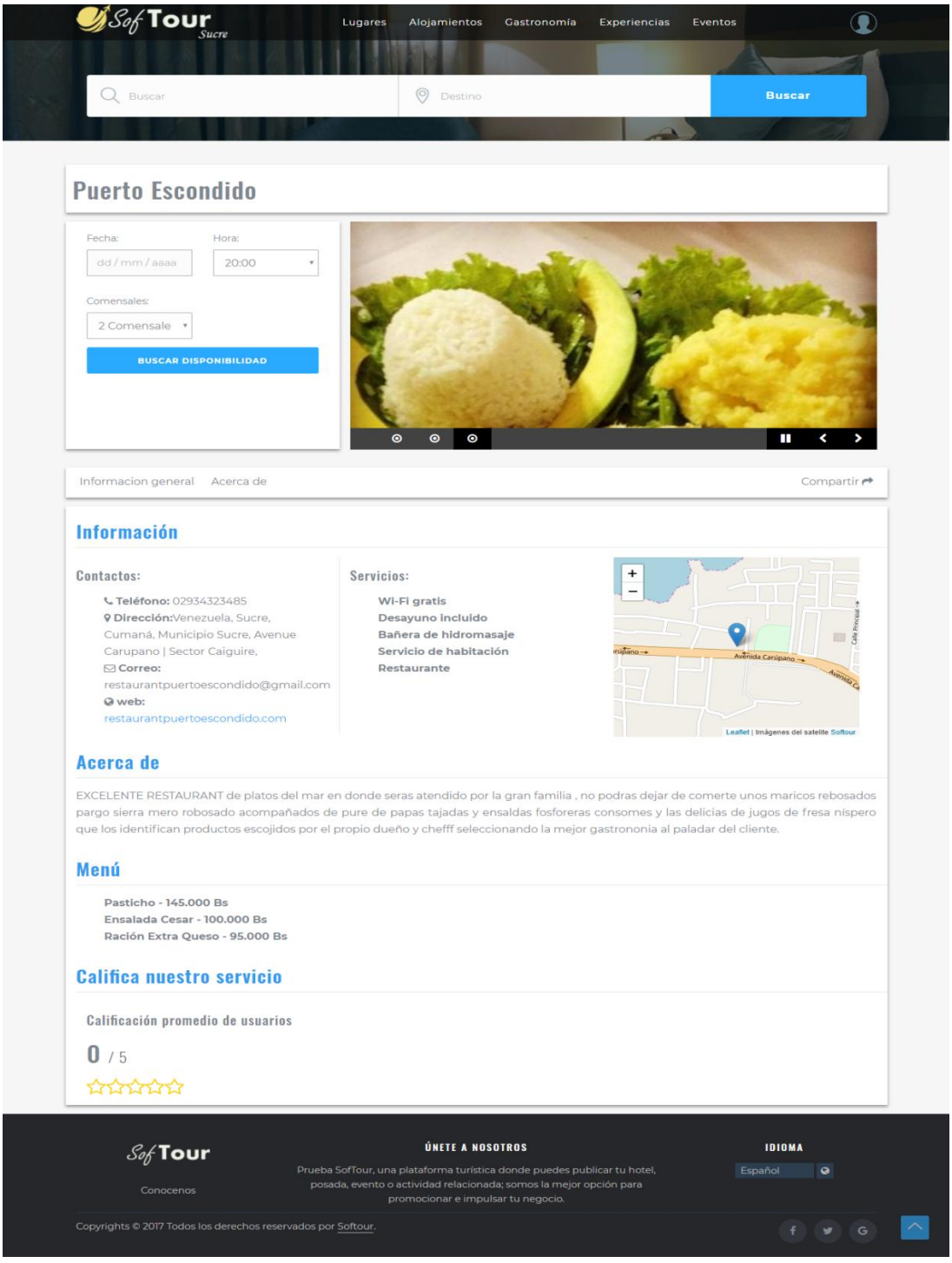


Figura PP. 8. Interfaz 2 de proceso de Consultar Gastronomía por usuario Turista

Código desarrollado para la funcionalidad Consultar Gastronomía por usuario Turista.

```
consultar(req, res, next) {
    let parametros = {
        "__t": req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1),
        "url": req.params.url.toLowerCase()
    };
    if(parametros.__t)
        parametros.aprobado = true;

    this._Lugar.buscar_consulta(parametros, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        if(buscado.__t){
            this.CategoriaAlojamiento.find({"alojamiento": buscado._id}, (error, lista_categorias) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    if (req.isAuthenticated())
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_consultar_producto(req, res);
                }

                res.locals.datos = {
                    "value": buscado,
                    "lista_categorias": lista_categorias
                };

                if (req.isAuthenticated())
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                this._respuesta.para_consultar(req, res);
            });
        }
        else {
            res.locals.datos = { "value": buscado };

            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
    })
}
```

Figura PP. 9. Código desarrollado para el proceso Consultar Gastronomía Por usuario Turista.

Prueba Consultar gastronomía

La sección gastronomía cuenta con todo lo referente a restaurantes o bares que se puedan localizar en determinado sitio, en la siguiente prueba se validará que se muestre de forma correcta su interfaz ante el turista.

Figura PP. 10. Prueba consultar gastronomía por turista. Paso 1

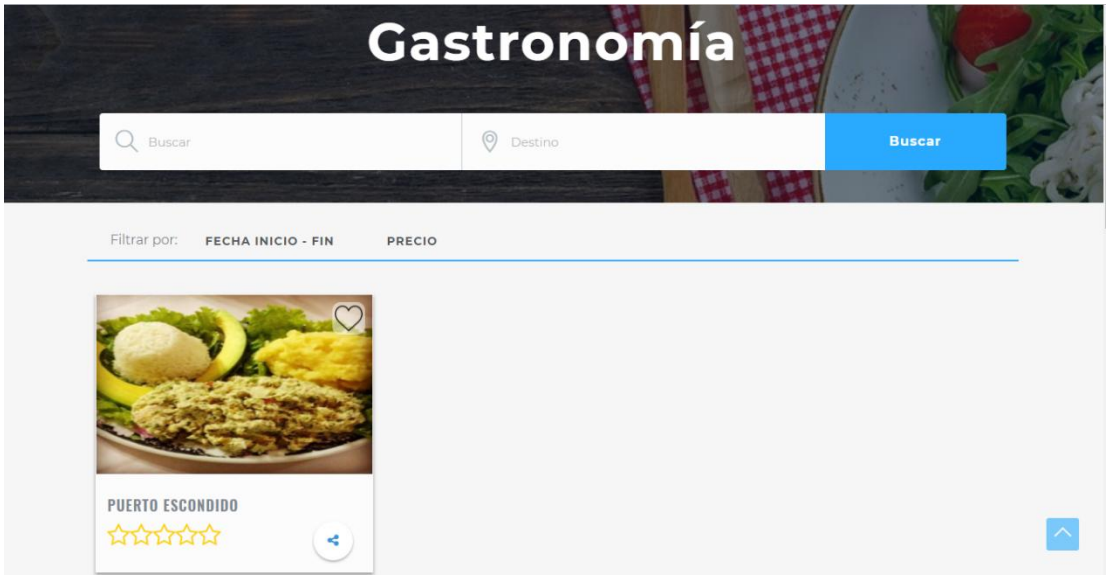


Figura PP. 11. Prueba Consultar Gastronomía por Turista. Paso 2.

Conclusión de la prueba

La función cumple con el objetivo para la cual fue desarrollada

Prueba de interfaz y usabilidad

En encuesta realizada con anterioridad se obtuvo información la cual indica que la interfaz es óptima para el uso del usuario.

Apéndice QQ. Consultar Lugar por Turista

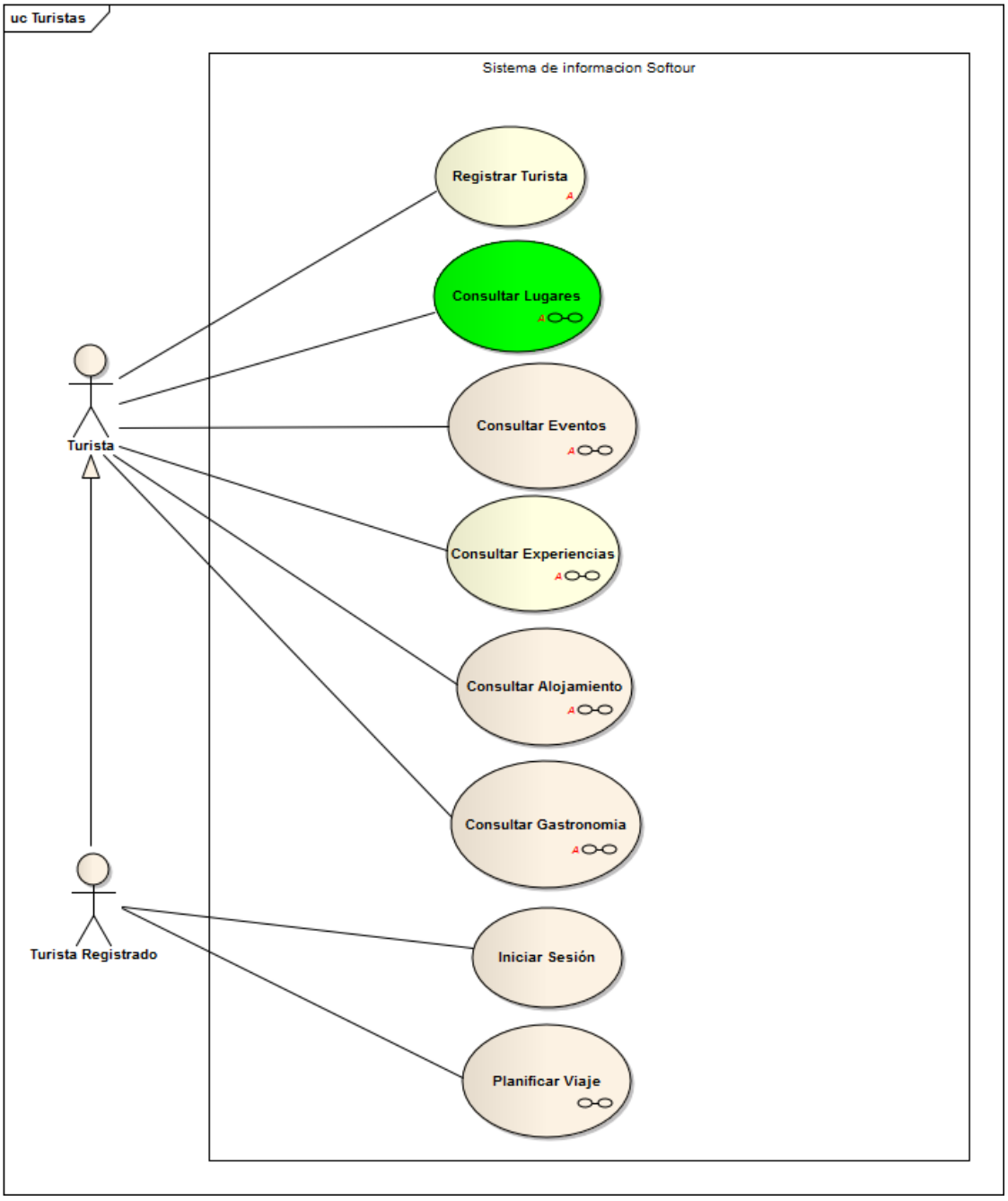


Figura QQ. 1. Caso de uso: Consultar Lugar por usuario Turista.

Tabla QQ. 1. Descripción Caso de uso Consultar Lugar por usuario Turista.

Nombre del Caso de uso	Consultar lugar	
Versión	1.0 03/11/2017	
Actor	Turista.	
Descripción del Caso de Uso	El caso de uso consiste en consultar los lugares desde una perspectiva visual, organizada y llamativa. Permitiendo enlazar el caso de uso con patrones de búsqueda específicos relacionados con lo que desea el turista.	
Precondición	Ninguna	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El caso de uso comienza cuando el Turista hace clic en el menú en la categoría “Lugares” o alguna subcategoría.
	2	El sistema despliega los lugares disponibles y muestra los posibles filtros que aplican para refinar aún más la búsqueda, acordes a lo que desea el turista. Esto se realiza extendiendo <<extend>> la funcionalidad de la búsqueda con el caso de uso: a.- Administrar patrones de búsqueda de lugares.
	3	El Turista visualiza los lugares disponibles que se muestran y selecciona uno de ellos para ver el detalle del mismo.
	4	El sistema muestra el detalle del lugar seleccionado.
	5	El Turista visualiza los atributos del lugar.
	6	El sistema cierra el caso de uso y retorna el control.
Postcondición	Se consulta el lugar desde la perspectiva del turista satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
Reglas del negocio	1. Las consultas de los lugares en los que se incluyan filtros o patrones de búsquedas deben considerarse extensiones de este caso de uso. 2. Al seleccionar un lugar, el turista podrá visualizar mayor detalle del mismo.	
Inclusiones	Ninguna	
Extensiones	1. Administrar patrones de búsqueda de lugar.	

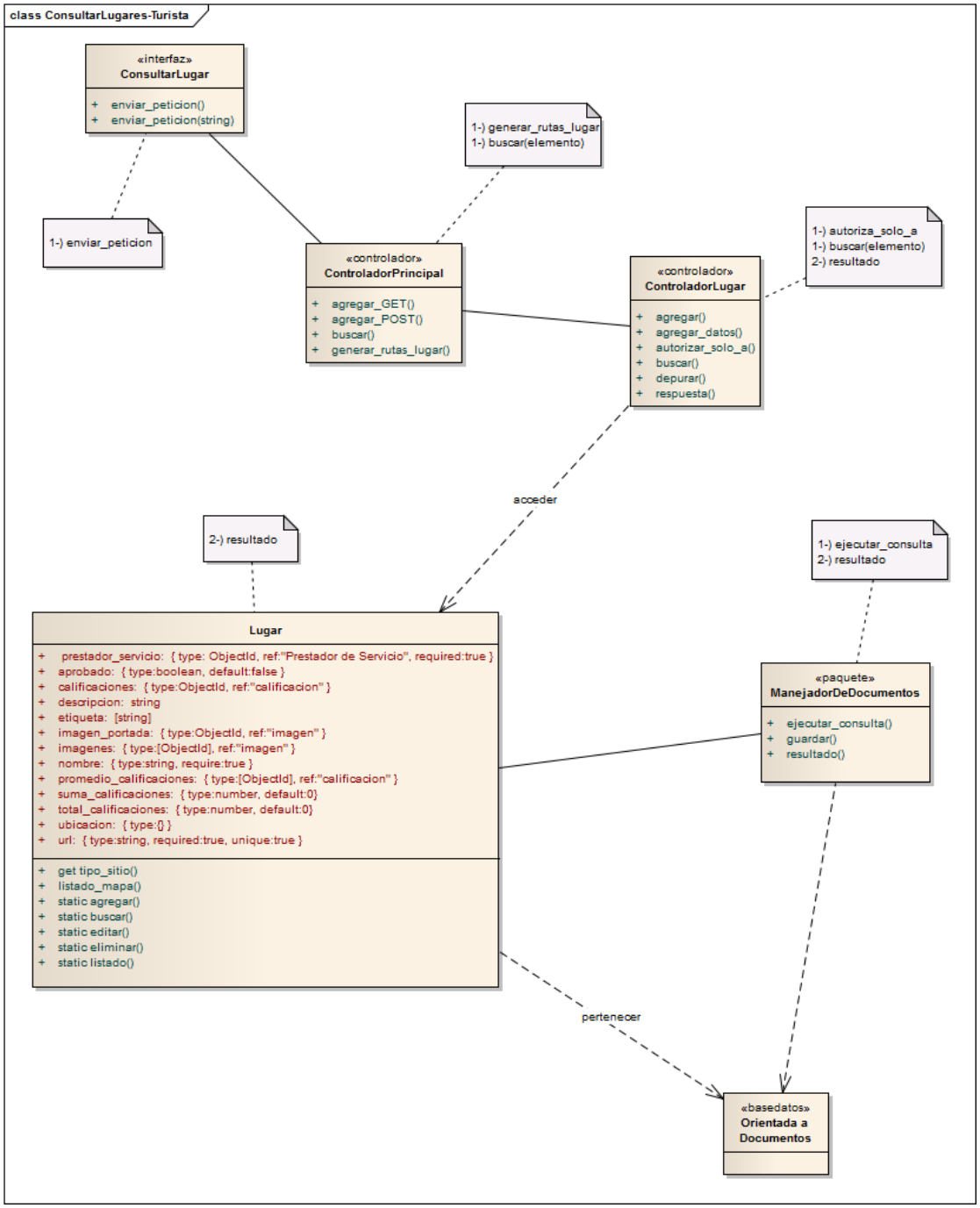


Figura QQ. 2. Diagrama de Clases: Consultar Lugar por Usuario Turista.

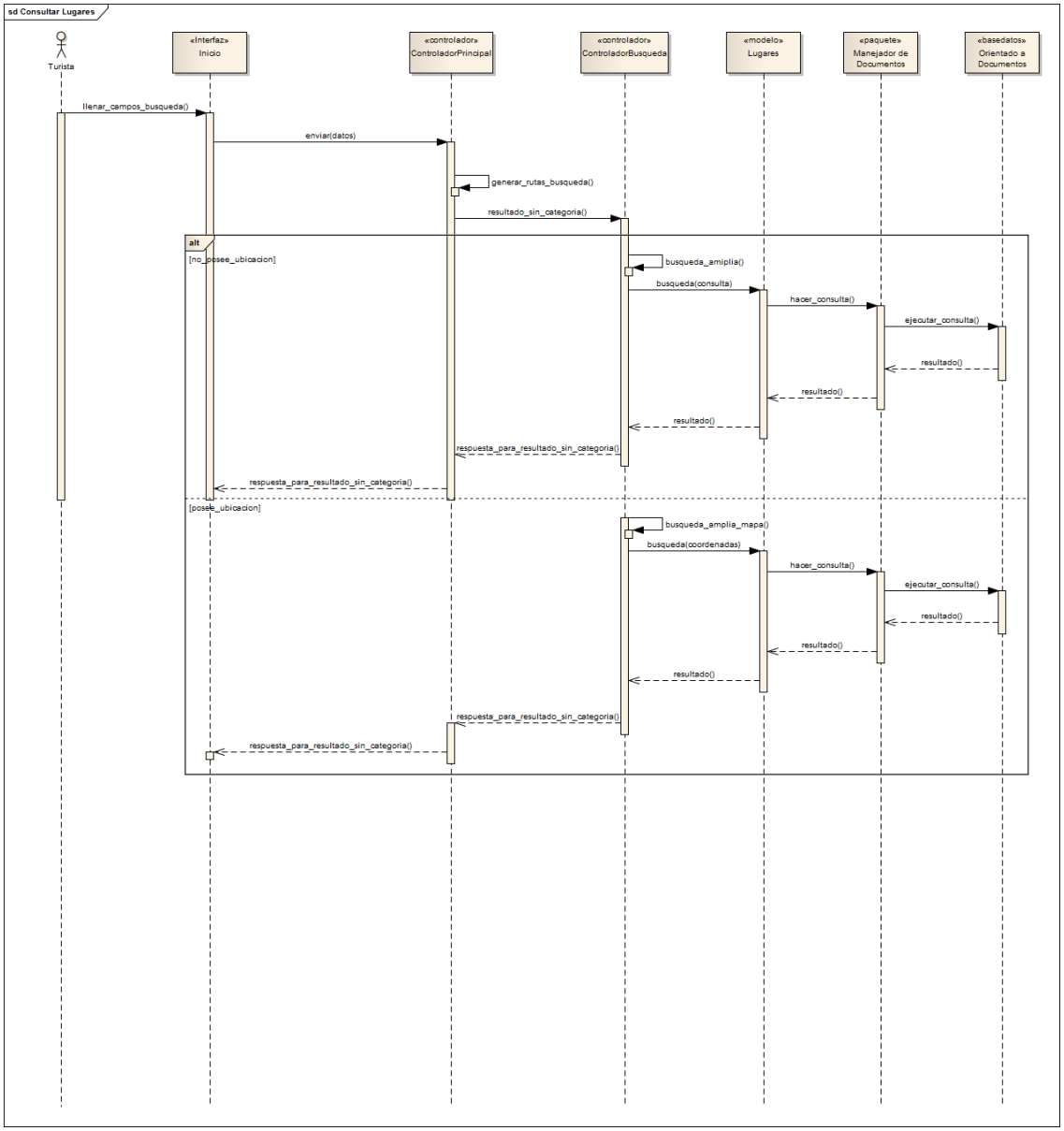


Figura QQ. 3. Diagrama de secuencia: Consultar Alojamiento por usuario Turista.

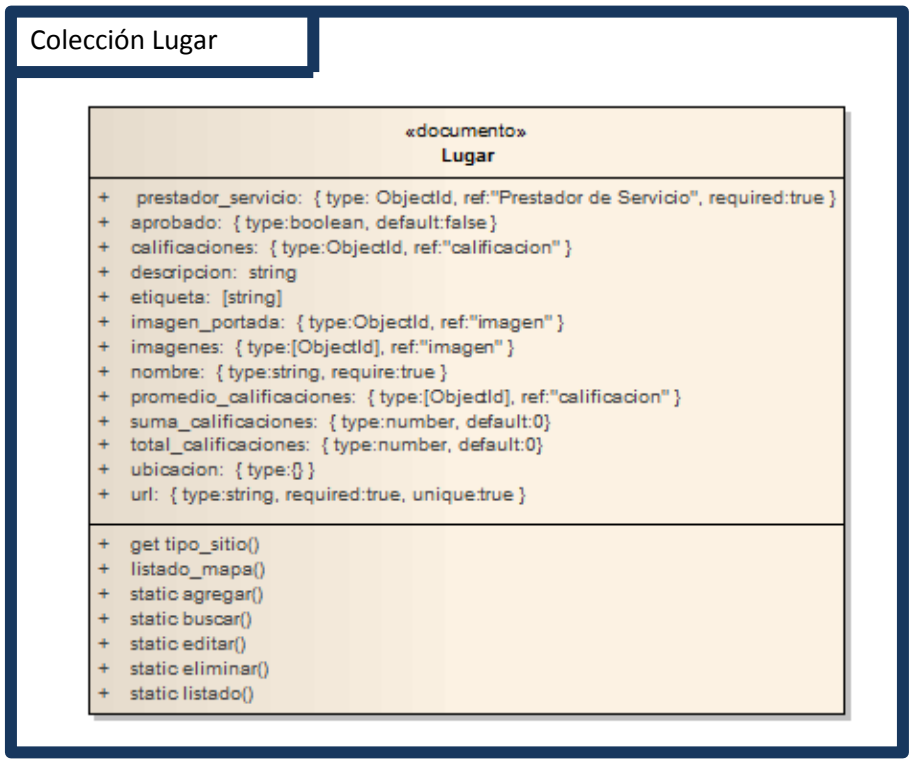


Figura QQ. 4. Colección Lugar: Consultar lugar por usuario Turista.

A continuación, se muestran los modelos de plantillas para el proceso de consultar lugar.

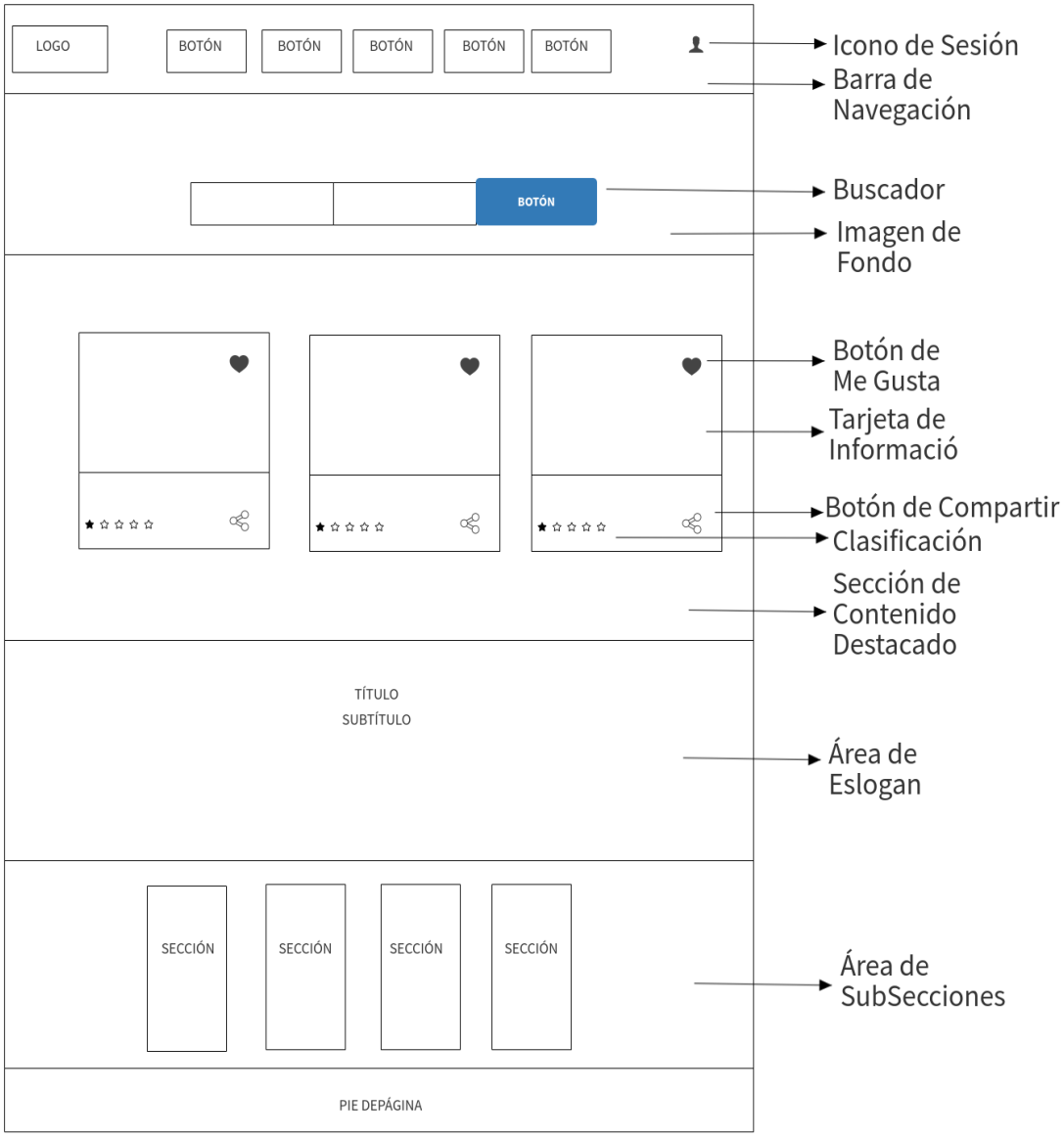


Figura QQ. 5. Plantilla de interfaz general: Consultar Lugar por usuario Turista.

Figura QQ. 6. Plantilla de interfaz especifica: Consultar Lugar por usuario Turista.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Lugar, y luego selecciona un lugar en específico y el sistema responde mostrando la interfaz con las

especificaciones.

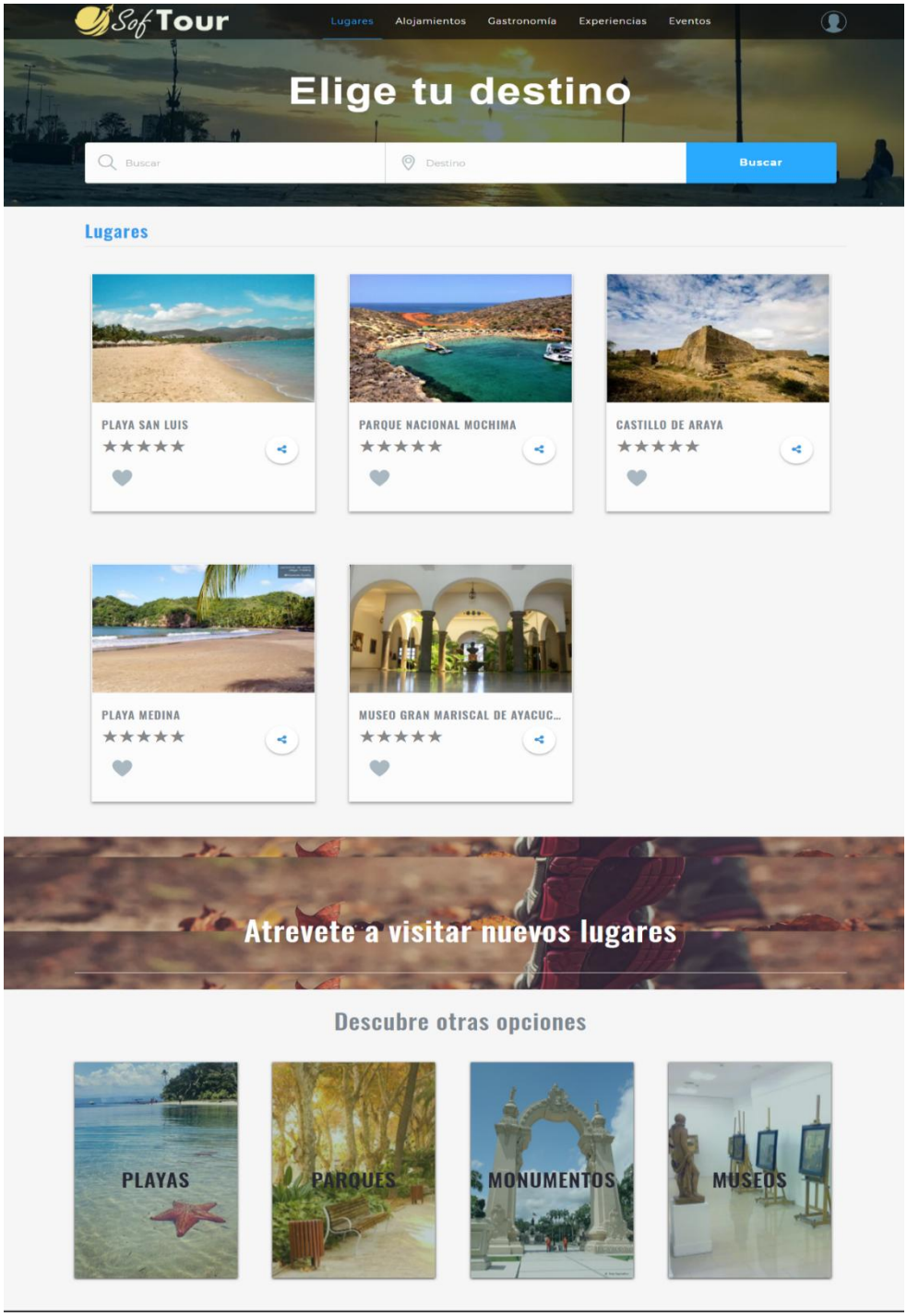


Figura QQ. 7. Interfaz 1 de proceso de Consultar Lugar por usuario Turista

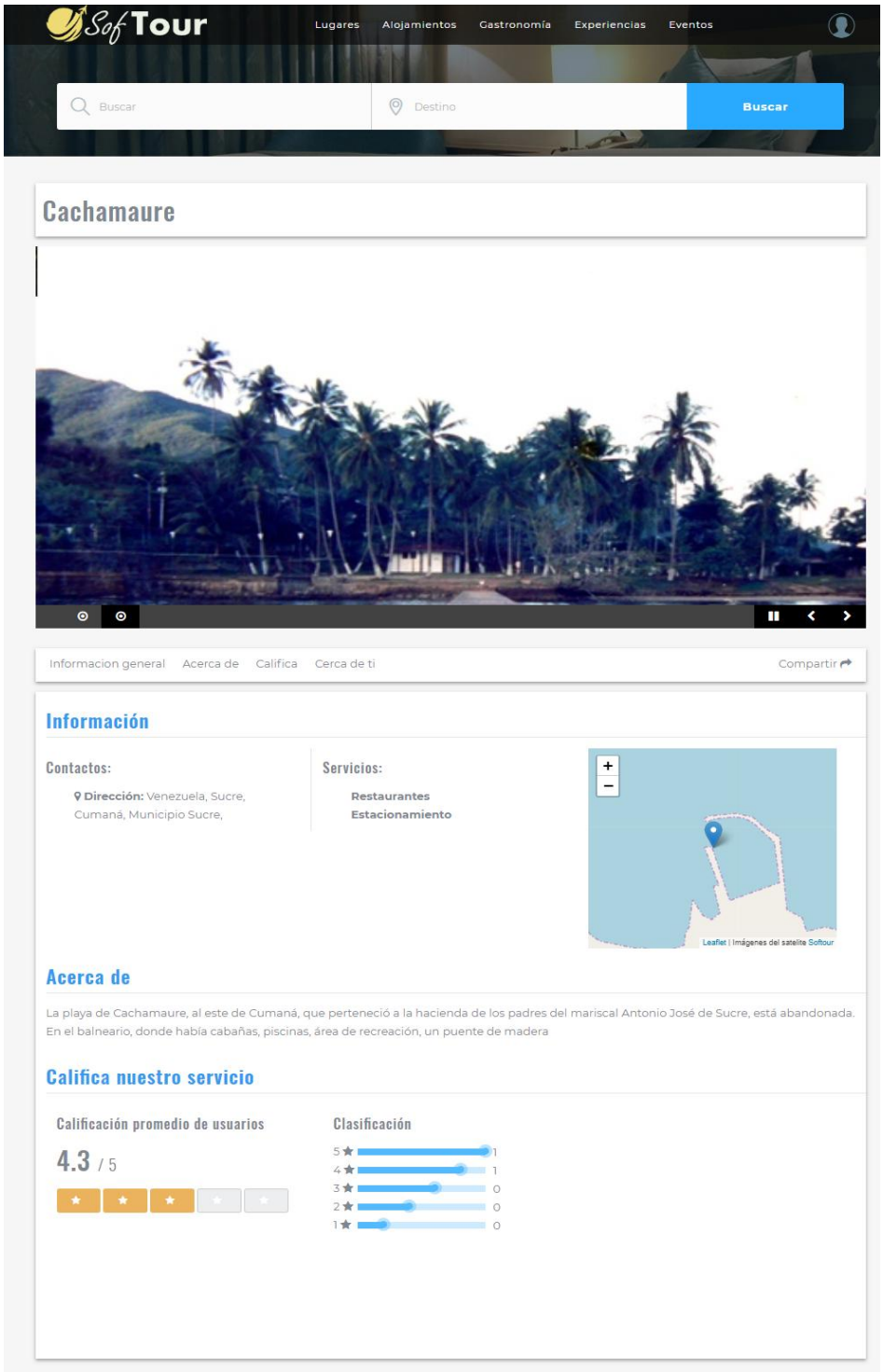


Figura QQ. 8. Interfaz 2 de proceso de Consultar Lugar por usuario Turista.

Código desarrollado para la funcionalidad Consultar Lugar por usuario Turista.

```
consultar(req, res, next) {
    let parametros = {
        "__t": req.params.tipo_entidad == "lugar" ? null : req.params.tipo_entidad.substring(0,1).toUpperCase()+req.params.tipo_entidad.substring(1),
        "url": req.params.url.toLowerCase()
    };
    if(parametros.__t)
        parametros.aprobado = true;

    this._Lugar.buscar_consulta(parametros, (error, buscado) => {
        if(error) {
            res.locals.datos = {"error": error};
            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
        if(!buscado) return next(new this._error_NoEncontrado());

        if(buscado.__t){
            this.CategoriaAlojamiento.find({"alojamiento": buscado._id}, (error, lista_categorias) => {
                if(error) {
                    res.locals.datos = {"error": error};
                    if (req.isAuthenticated())
                        res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                    this._respuesta.para_consultar_producto(req, res);
                }

                res.locals.datos = {
                    "value": buscado,
                    "lista_categorias": lista_categorias
                };

                if (req.isAuthenticated())
                    res.locals.datos.value.usuario_sesionado = req.session.passport.user;
                this._respuesta.para_consultar(req, res);
            });
        }
        else {
            res.locals.datos = { "value": buscado };

            if (req.isAuthenticated())
                res.locals.datos.value.usuario_sesionado = req.session.passport.user;
            this._respuesta.para_consultar(req, res);
        }
    })
}
```

Figura QQ. 9. Código desarrollado para el proceso Consultar Lugar por usuario Turista.

Prueba Consultar Lugar

Las pruebas realizadas a las funcionalidades que puede ejercer el Usuario Turista van dirigidas a ver que se muestren los componentes cargados en el sistema en sus diferentes apartados.

A continuación, se muestran imágenes de la sección lugares:

Se seleccione el apartado de los lugares en la interfaz principal de SofTour como se muestra a continuación:

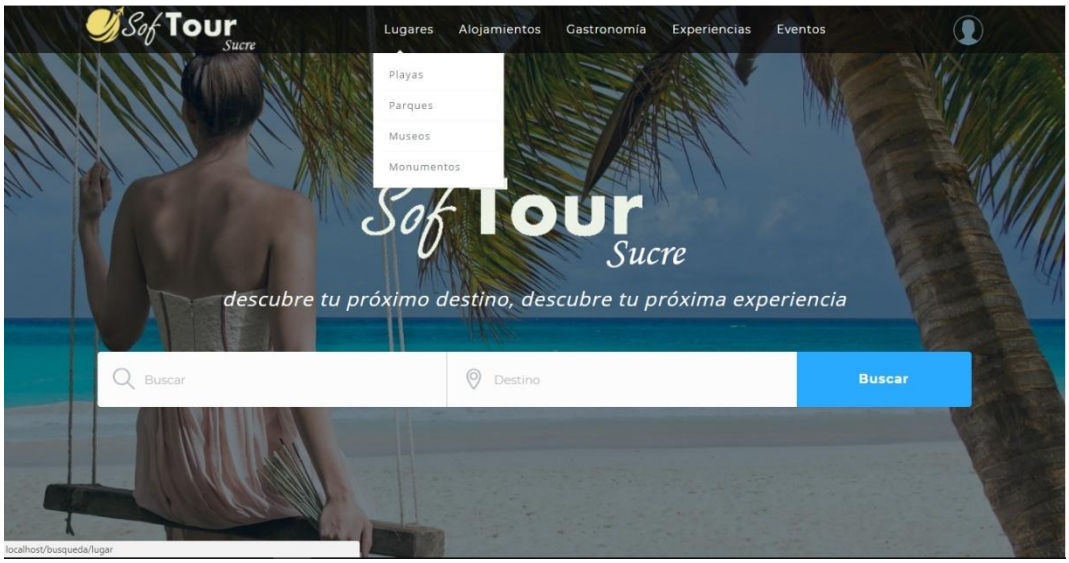


Figura QQ. 10. Prueba Consultar Lugar por Turistas. Paso 1

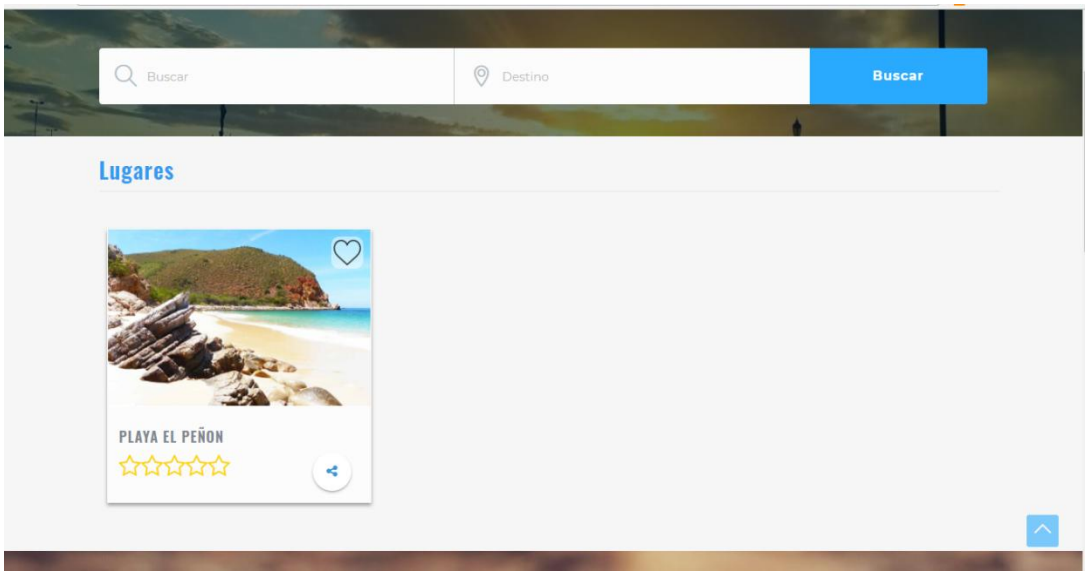


Figura QQ. 11. Prueba Consultar Lugar por Turistas. Paso 2.

Conclusión de la prueba

Se observo la interfaz correctamente lo cual indica que se cumple la función.

Prueba de interfaz y usabilidad

Para realizar esta prueba se utilizó como herramienta la siguiente encuesta

Formato de entrevista utilizado para obtener estos resultados

ENTREVISTADO:

FECHA:

HORA:

LUGAR:

ASUNTO O TEMA A TRATAR:

Dar inicio a la entrevista

Presentación y agradecimiento al entrevistado.

Breve descripción de lo que se desea con la entrevista.

A continuación preguntas, marque con una **X** su respuesta.

1.- ¿La organización de la página es constante?

SI _____ NO_____

De ser No su respuesta explique porque _____

2.- ¿Los contenidos del sistema están organizados de manera adecuada para que usted como usuario pueda acceder a ellos?

SI _____ NO_____

De ser No su respuesta explique porque _____

3.- ¿Los botones, menú y otros mecanismos de navegación son fáciles de entender?

SI _____ NO_____

De ser No su respuesta explique porque _____

4.- ¿La distribución del sistema posee una estructura firme?

SI _____ NO_____

De ser No su respuesta explique porque _____

5.- ¿El contenido está bien escrito y es claro?

SI _____ NO_____

De ser No su respuesta explique porque _____

6.- ¿El color y tamaño de las letras utilizadas son los apropiados?

SI _____ NO_____

De ser No su respuesta explique porque _____

7.- ¿Son atractivos los colores utilizados en el sistema?

SI _____ NO_____

De ser No su respuesta explique porque _____

8.- ¿La apariencia del sistema atrae su atención como para usarlo?

SI _____ NO_____

De ser No su respuesta explique porque_____

9.- ¿Soluciona su problema de búsqueda el uso de la aplicación?

SI _____ NO_____

De ser No su respuesta explique porque _____

10.- ¿El sistema cumple con el propósito para el cual se desarrolló?

SI _____ NO_____

De ser No su respuesta explique porque _____

11.- ¿La navegación a través del sistema es cómoda?

SI _____ NO_____

De ser No su respuesta explique porque _____

Concluir la entrevista agradeciendo al entrevistado por su colaboración.

Los resultados obtenidos al realizar la encuesta están detallados en la siguiente tabla

Preguntas a los usuarios	**Si**	**No**
1) ¿La organización de la página es constante?	8	2
2) ¿Los contenidos del sistema están organizados de manera adecuada para que usted como usuario pueda acceder a ellos?	9	1
3) ¿los botones, menú y otros mecanismos de navegación son fáciles de entender y utilizar?	9	1
4) ¿La distribución del sistema posee una estructura firme?	8	2
5) ¿El contenido está bien escrito y es claro?	7	3
6) ¿El color y tamaño de las letras utilizadas son los apropiados?	9	1
7) ¿Son atractivos los colores utilizados en el sistema?	10	0
8) ¿La apariencia del sistema atrae su atención como para usarlo?	9	1
9) ¿Soluciona su problema de búsqueda el uso de la aplicación?	7	3
10) ¿El sistema cumple con el propósito para el cual se desarrolló?	8	2
11) ¿La navegación a través del sistema es cómoda?	8	2

Apéndice RR. Inicio Sesión Turista.

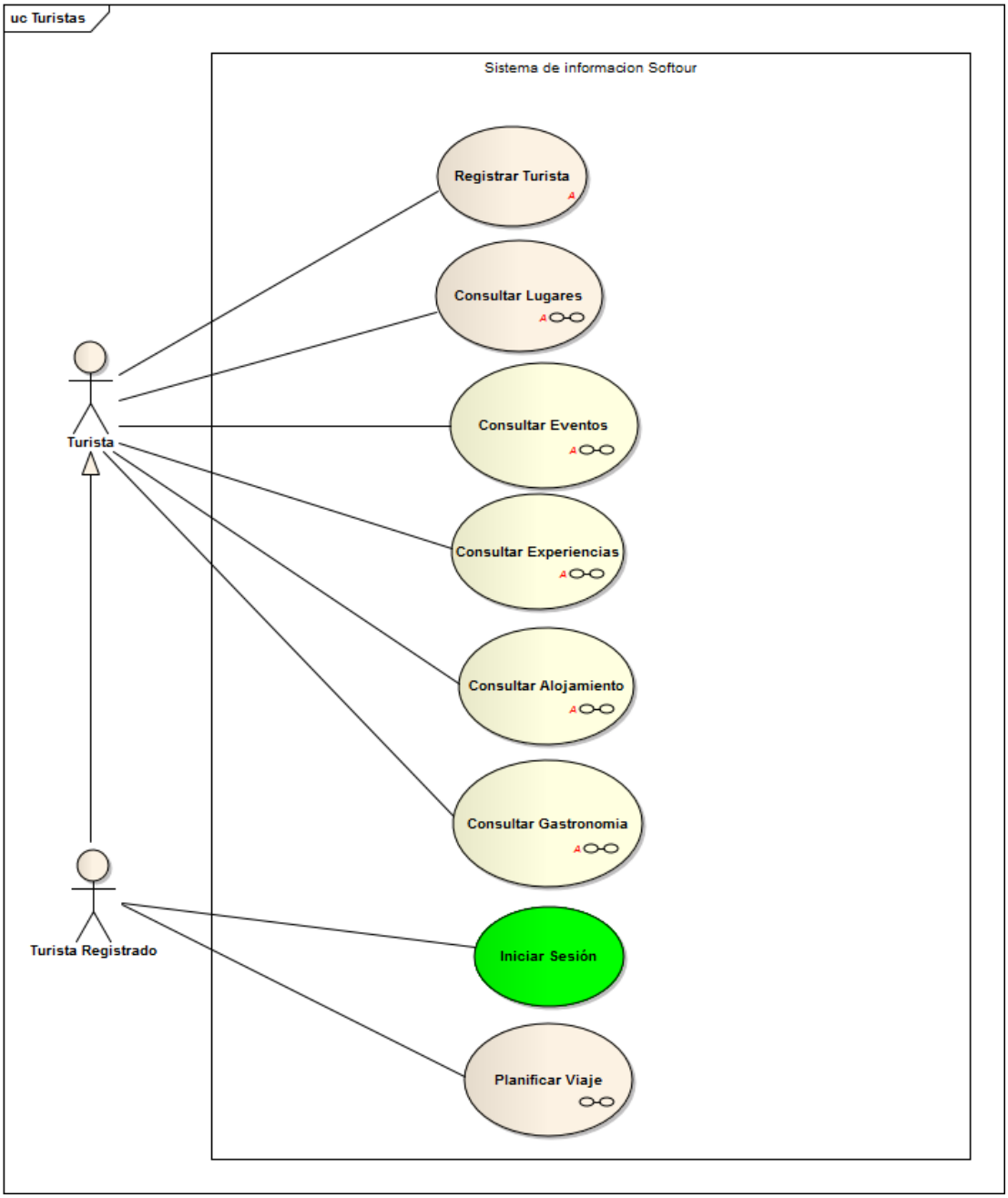


Figura RR. 1. Caso de uso: Inicio Sesión Turista.

Tabla RR. 1. Descripción Caso de uso Inicio Sesión Turista.

Nombre del Caso de uso	Iniciar Sesión.	
Versión	1.0 25/10/2017	
Actor	Turista.	
Descripción del Caso de Uso	El caso de uso consiste en iniciar sesión satisfactoriamente.	
Precondición	Para iniciar sesión, debe estar registrado en la base de datos del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Turista selecciona la opción “iniciar sesión”.
	2	El Sistema ejecuta la búsqueda del inicio de sesión, mostrando un formulario con Usuario y Contraseña.
	3	El Turista llena los campos correspondientes al formulario.
	4	El Sistema valida el tipo de dato ingresado.
	5	El Turista selecciona “iniciar sesión”.
	6	El Sistema lo re direcciona a su perfil.
Postcondición	Se inicia sesión satisfactoriamente.	
Curso alternativo de Eventos	**Paso**	**Acción**
	4	En el paso 4: a) El Sistema muestra un mensaje “No se consiguió ningún usuario con estos datos” retornando al paso 3. b) El Turista corrige los dato(s), selecciona “iniciar sesión”, luego el sistema lo re direcciona a su perfil y finaliza el caso de uso.
Reglas del negocio	1. El Turista debe tener un correo y una contraseña.	
Inclusiones	Ninguna.	
Extensiones	Ninguna.	

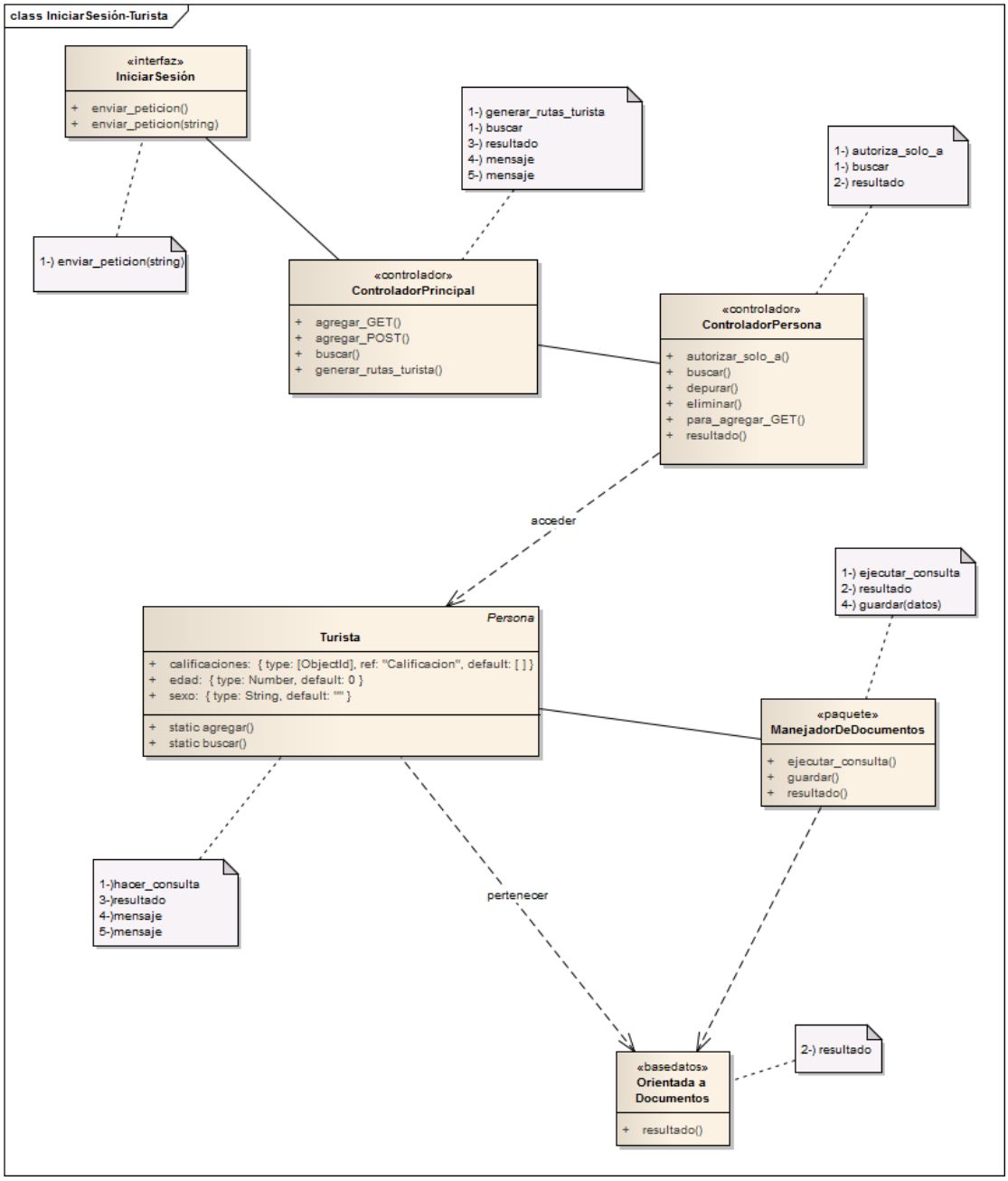


Figura RR. 2. Diagrama de Clases: Inicio Sesión Turista.

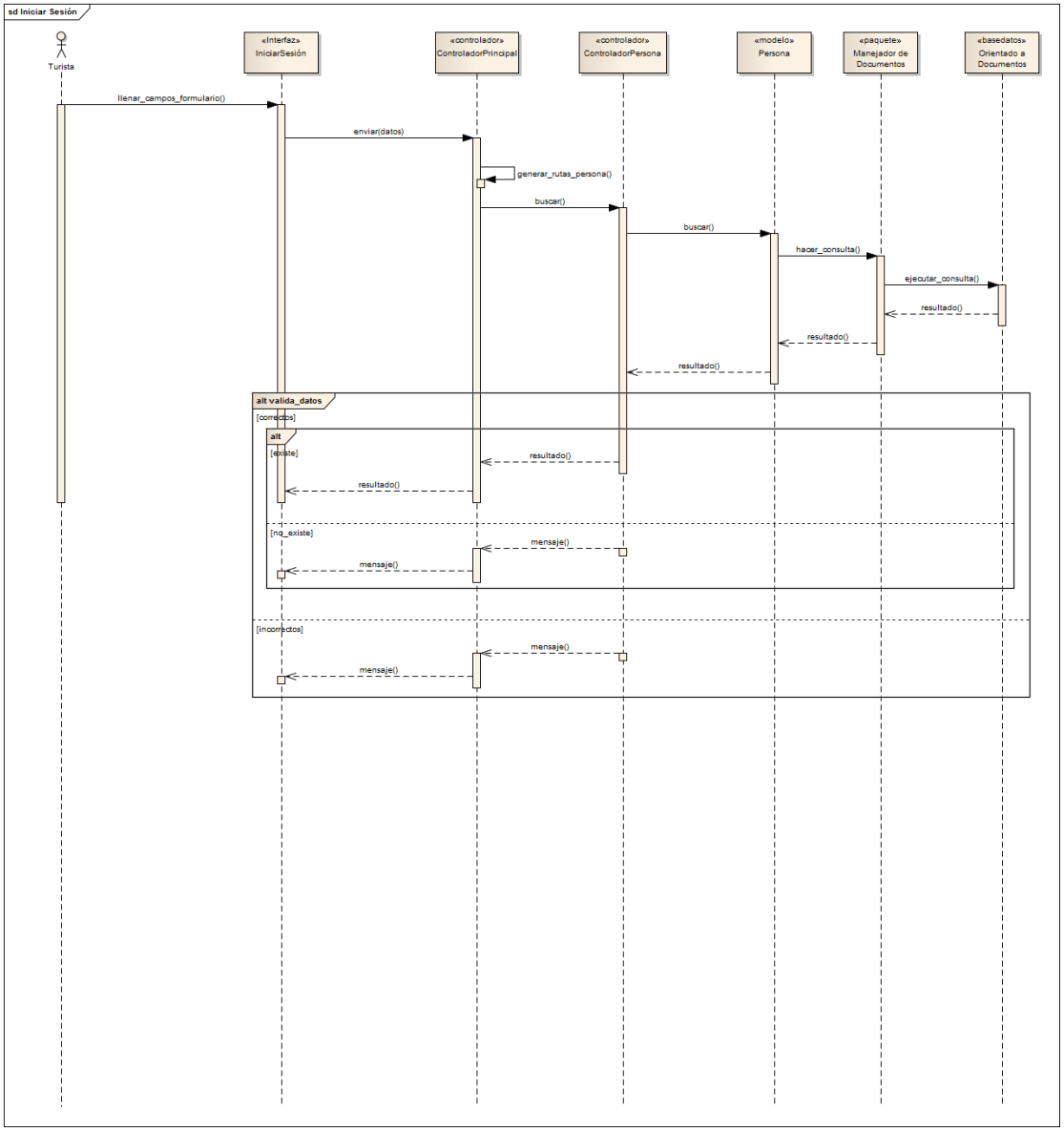


Figura RR. 3. Diagrama de secuencia: Inicio Sesión Turista.

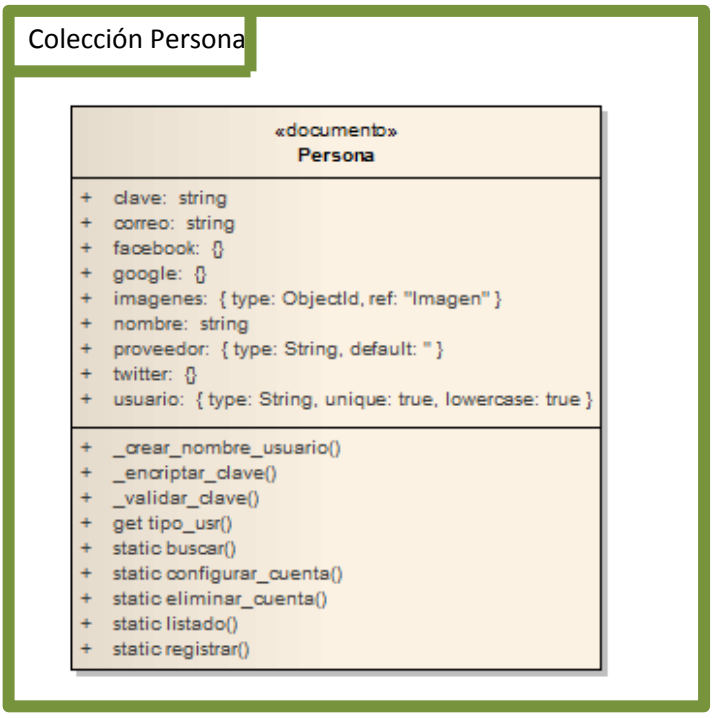


Figura RR. 4. Colección Persona: Inicio sesión por usuario Turista.

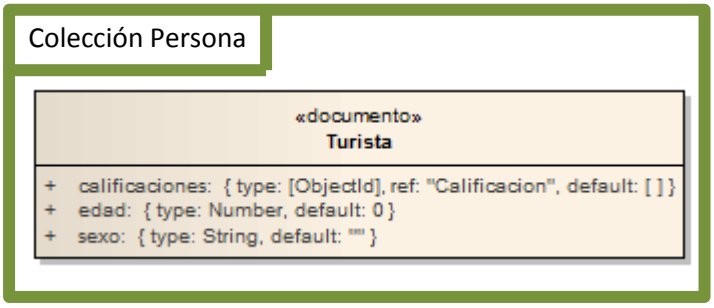


Figura RR. 5. Colección Persona: Inicio sesión por usuario Turista.

A continuación, se muestran los modelos de plantillas para el proceso de iniciar sesión.

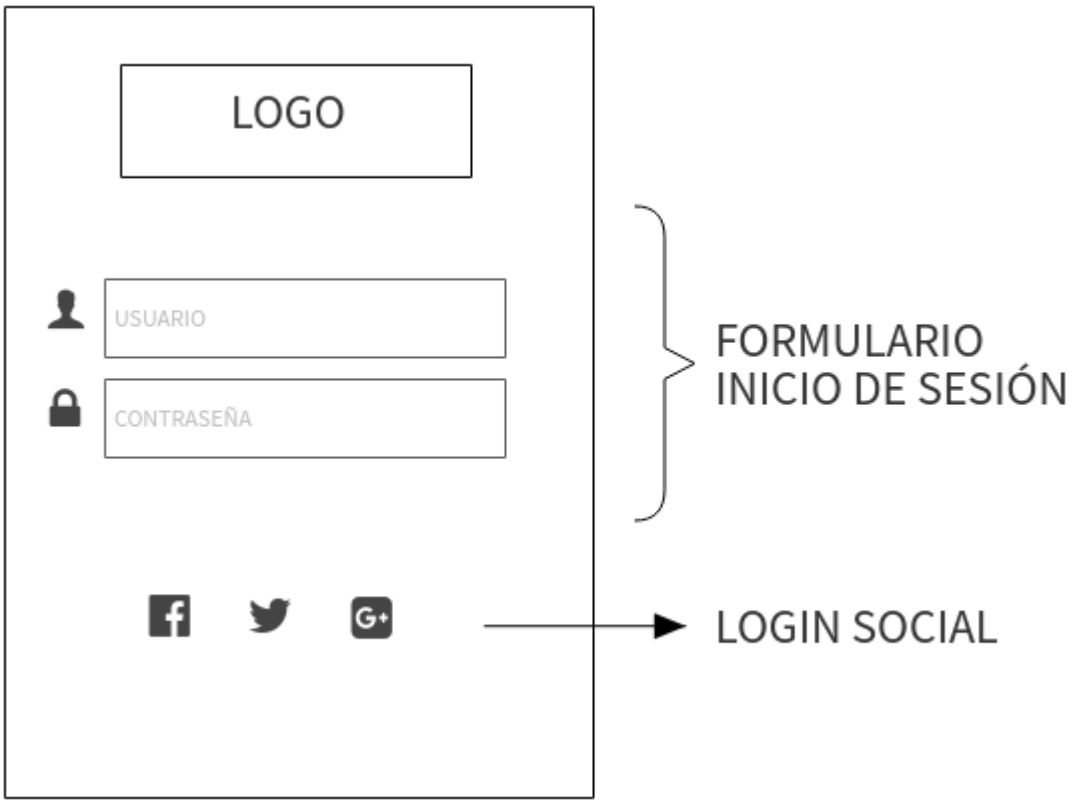


Figura RR. 6. Interfaz de proceso Inicio de Sesión Turista

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona iniciar sesión y el sistema muestra el formulario a llenar para ingresar como usuario Turista.

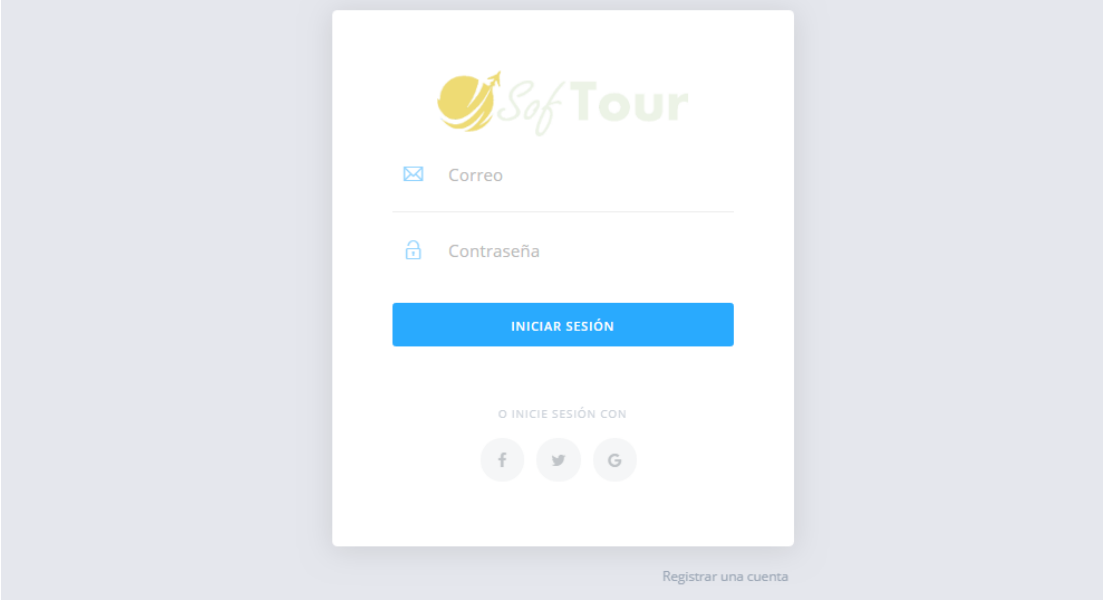


Figura RR. 7. Interfaz de proceso Inicio de Sesión Turista.

```
sesionarme_POST(req, res, next) {

    res.locals.datos = this._validar.campos(req.body, req.body.tipo_usr);

    if(res.locals.datos.error)
        this._respuesta.para_sesionarme_POST(req, res);

    else
        this._passport.authenticate('sesionar-usuario', { failureFlash : true},
         (error, usuario, datos_error, next) => {
            if (error) return next(error);

            if (!usuario || datos_error){
                res.locals.datos.error = datos_error;
                this._respuesta.para_sesionarme_POST(req, res);
            }

            else
                req.login(usuario, (error) => {
                    if (error) return next(error);

                    res.locals.datos = {"value": usuario};
                    this._respuesta.para_sesionarme_POST(req, res);
                });
        }) (req, res, next);
}
```

Figura RR. 8. Código desarrollado para el proceso Inicio Sesión Por usuario Turista.

Apéndice SS. Regsitrar Turista.

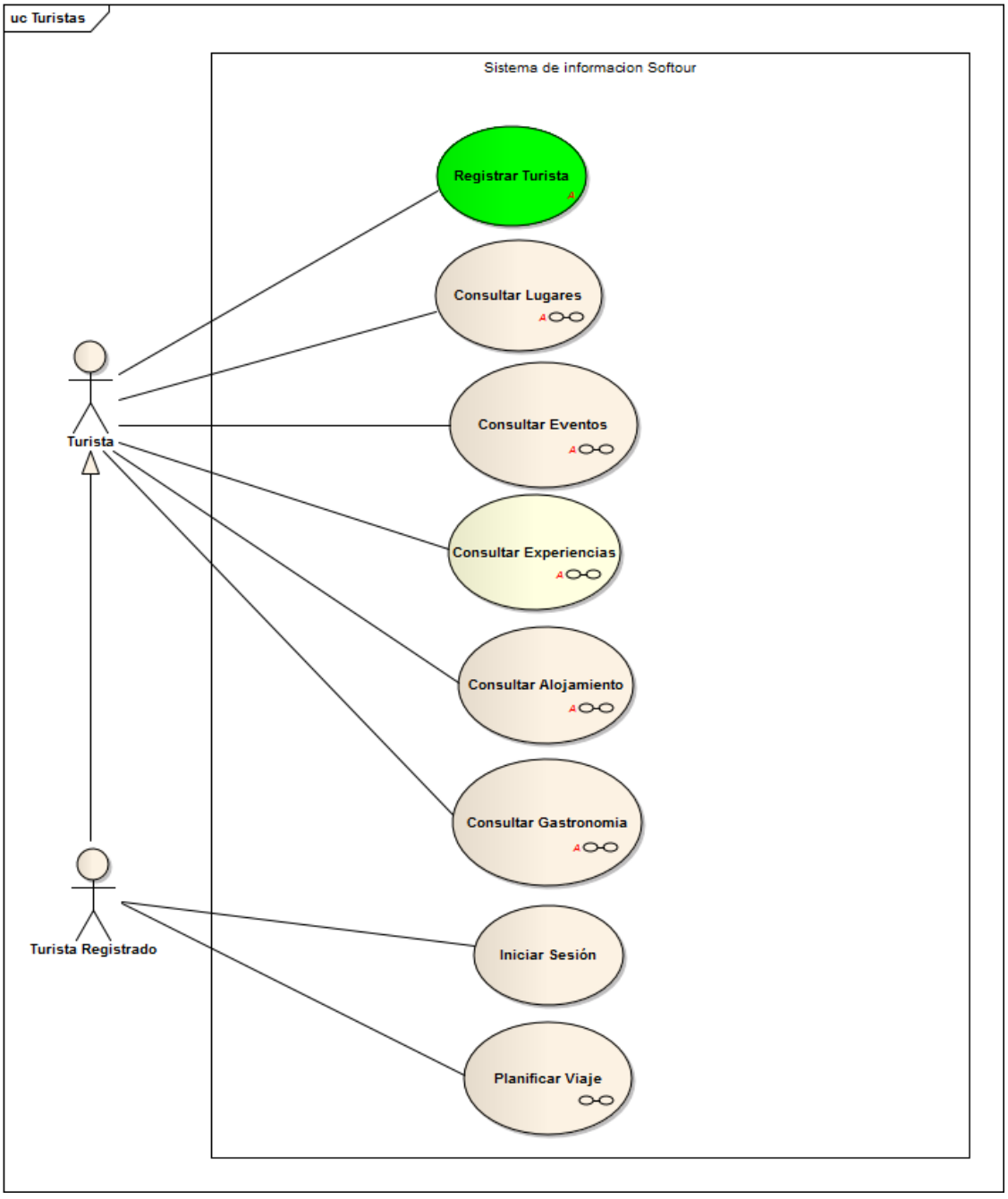


Figura SS. 1. Caso de uso: Registrar Turista.

Tabla SS. 1. Descripción Caso de uso Registrar Turista.

Nombre del Caso de uso	Registrar Turista	
Versión	1.0 03/11/2017	
Actor	Turista.	
Descripción del Caso de Uso	El caso de uso consiste en registrar al turista, ingresando los atributos del mismo de manera satisfactoria para que pase a ser un turista registrado.	
Precondición	Para registrar a un turista, este no debe estar en la BD del sistema.	
Secuencia Normal de Eventos	**Paso**	**Acción**
	1	El Turista selecciona la opción “Registrarse”.
	2	El sistema muestra el formulario correspondiente.
	3	El Turista ingresa los atributos: correo electrónico, usuario, clave, nombre.
	4	El sistema almacena los atributos de asociados al Turista y finaliza el caso de uso.
Postcondición	Se ingresa los atributos del Turista de manera satisfactoria.	
Curso alternativo de Eventos	**Paso**	**Acción**
	3.1.	El Turista decide no registrarse y selecciona cancelar el caso de uso (selecciona el botón cerrar el formulario o el botón cancelar).
	4.1.	El sistema cierra el formulario y finaliza el caso de uso.
Reglas del negocio	1. El Turista no debe estar ingresado en la BD del sistema para que pueda ser ingresada. Si el correo electrónico del turista existe no se puede duplicar el registro.	
Inclusiones	Ninguna.	
Extensiones	Ninguna	

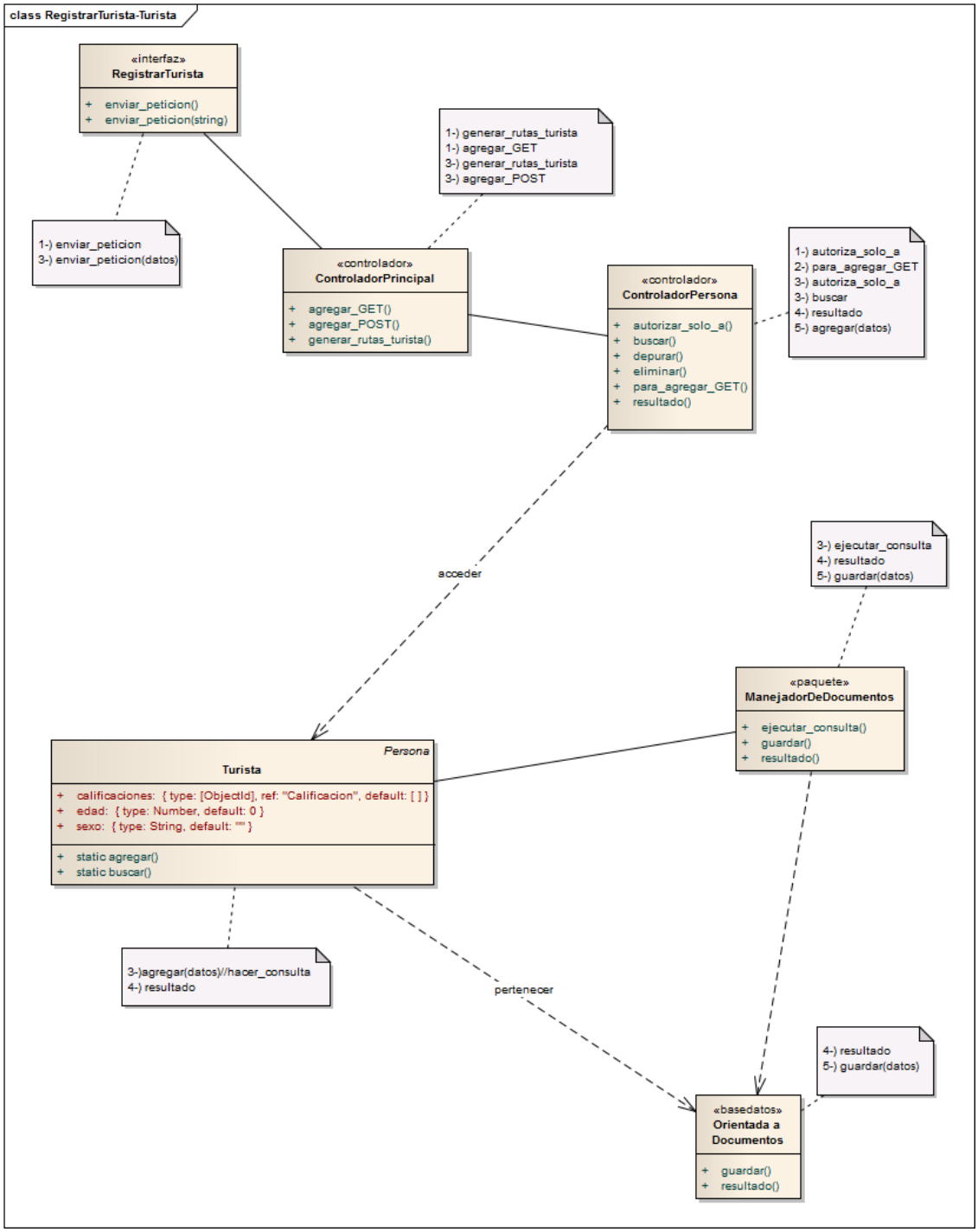


Figura SS. 2. Diagrama de Clases: Registrar Turista.

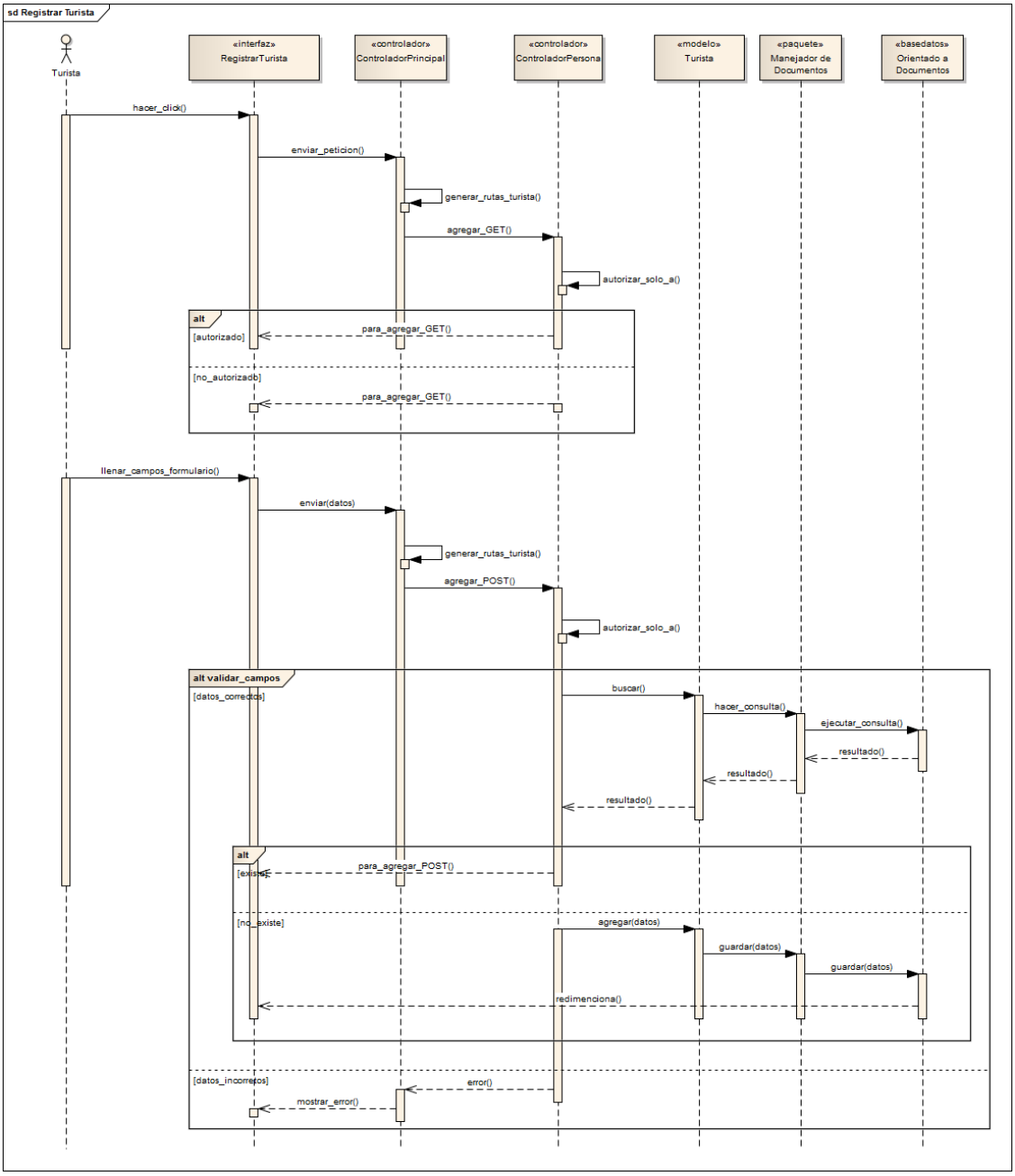


Figura SS. 3. Diagrama de secuencia: Registrar Turista

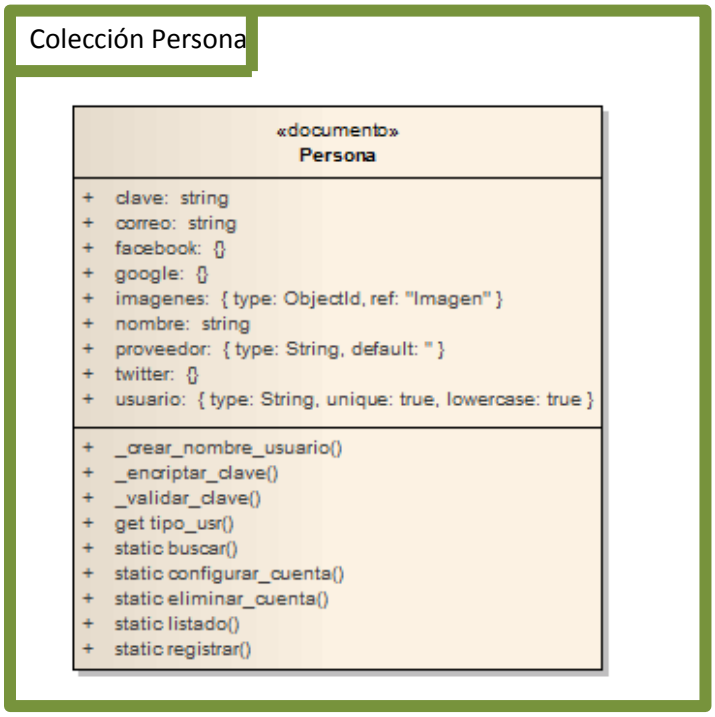


Figura SS. 4. Imagen Colección Persona:

Registrar usuario Turista.

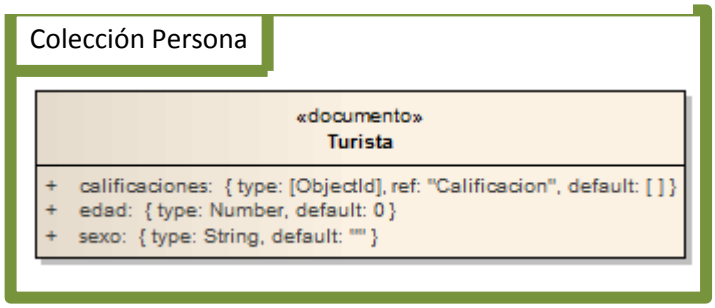


Figura SS. 5. Imagen Colección Persona:

Registrar usuario Turista.

A continuación, se muestran los modelos de plantillas para el proceso de iniciar sesión.

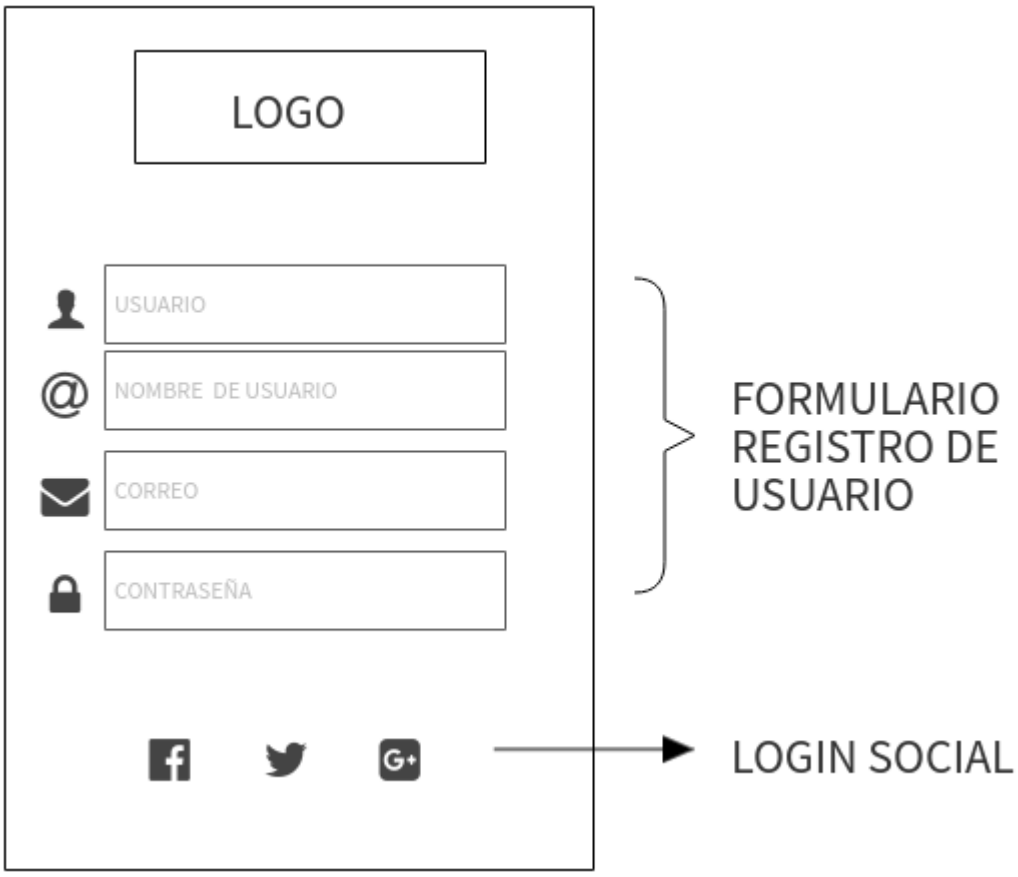


Figura SS. 6. Interfaz de proceso de Registrar Turista.

Flujo a seguir para la funcionalidad.

El Prestador de servicio selecciona la opción Registrar y el sistema le muestra formulario para ingreso de datos, luego el sistema valida que los datos ingresados no contengan anomalías y procede a registrar el usuario.

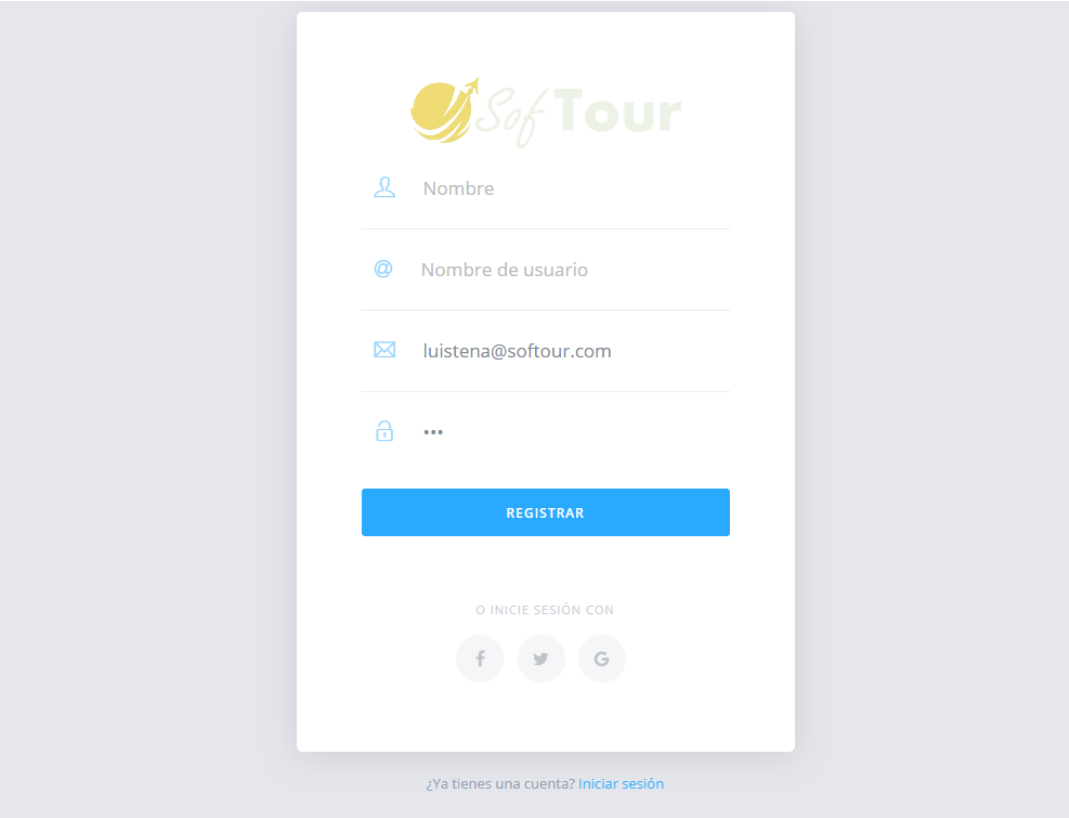


Figura SS. 7. Interfaz de proceso de Registrar Turista

Código desarrollado para la funcionalidad Registrar Turista.

```
ingresar_POST(req, res, next) {

    res.locals.datos = this._validar.campos(req.body, req.body.tipo_usr);

    if(res.locals.datos.error)
        this._respuesta.para_ingresar_POST(req, res);

    else
        // si SESSION es true, entonces q genere la sesion si es false entonces significa que el admin intenta registrar a un usuario
        Prestador Servicio y no se requiere sesionarlo
        return this._passport.authenticate('registrar-usuario', {
                failureFlash: true,
                session: res.locals.datos.value.tipo_usr == "Turista" ? true : false
            }, (error, usuario, datos_error, next) => {
            if(error) return next(error);

            if(datos_error)
                res.locals.datos.error = datos_error;
            else
                res.locals.datos = {"value": usuario};

            this._respuesta.para_ingresar_POST(req, res);
        }) (req, res, next);
}
```

Figura SS. 8. Código desarrollado para el proceso Registrar Turista.

Apéndice TT. Plantilla de Volére para la documentación de requisitos y descripción de casos

Identificador del Requisito: 01	**Tipo de Requisito:** Funcional	**Caso de Uso/Evento:** Agregar Lugar.
Descripción: El usuario registrado "Administrador" ingresa un lugar con el fin de cargar las características o atributos relevantes (etiqueta, nombre, descripción, dirección física, URL, valoración) relacionados con el mismo, que serán almacenados en el sistema para futuras consultas.		
Justificación del Requisito: Es necesario registrar el lugar para que los usuarios puedan visualizarlos, poder seleccionar entre múltiples opciones en la geografía específica que desee conocer el usuario final.		
Fuente (que interesado lo propone): Prestador del servicio.	**Unidad en la que se origina:** Administrador de empresa.	
Criterios de validación: Para registrar un sitio de interés debe ser administrador de empresa.		
Grado de satisfacción del interesado: 6	**Grado de insatisfacción del interesado:** 4	
Dependencias (qué requisitos depende de este): 02, 03, 05, 06, 08, 11, 20.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 01/03/2017.	
Proyecto: Softour	**Analista:** Grupo Informático de Desarrollo de Software (GIDSoft)	

Figura TT. 1. Plantilla de Volére Agregar Lugar, requisito 01

Identificador del Requisito: 02	**Tipo de Requisito:** Funcional	**Caso de Uso/Evento:** Agregar Alojamiento.
Descripción: El usuario registrado “Administrador” agrega un alojamiento relacionado con un sitio de interés con el fin de cargar las características o atributos relevantes del mismo (etiqueta, nombre, descripción, dirección física, URL, valoración, cantidad de habitaciones, categoría, rutas de acceso, persona contacto, tipos de habitación) relacionados con el mismo, que serán almacenados en el sistema para futuras consultas.		
Justificación del Requisito: Es necesario registrar los alojamientos relacionados con los sitios de interés para que los usuarios puedan visualizarlos, poder seleccionar entre múltiples opciones de hospedajes según el sitio de interés específico que desee conocer el usuario final.		
Fuente (que interesado lo propone): Prestador del servicio.	**Unidad en la que se origina:** Administrador de empresa.	
Criterios de validación: Para registrar un hospedaje debe ser administrador de empresa.		
Grado de satisfacción del interesado: 6	**Grado de insatisfacción del interesado:** 4	
Dependencias (qué requisitos depende de este): 01, 03, 05, 06, 08, 11.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 01/03/2017.	
Proyecto: Softour	**Analista:** Grupo Informático de Desarrollo de Software (GIDSoft)	

Figura TT. 2. . Plantilla de Volére Agregar Alojamiento, requisito 02

.

Identificador del Requisito: 03	**Tipo de Requisito:** Funcional	**Caso de Uso/Evento:** Agregar Transporte.
Descripción: El usuario registrado "Administrador de Empresa" agrega un transporte relacionado con un sitio de interés y con hospedaje con el fin de cargar las características o atributos relevantes del mismo (etiqueta, nombre, descripción, dirección física, URL, valoración, lugar de salida, hora de salida, lugar de llegada, hora de llegada, persona contacto, tipos de transporte) relacionados con el mismo, que serán almacenados en el sistema para futuras consultas.		
Justificación del Requisito: Es necesario registrar los transportes relacionados con los sitios de interés y los hospedajes para que los usuarios puedan visualizarlos, poder seleccionar entre múltiples opciones de transportes según el lugar a donde quiera ir el usuario final.		
Fuente (que interesado lo propone): Prestador del servicio.	**Unidad en la que se origina:** Administrador de empresa.	
Criterios de validación: Para registrar un transporte debe ser administrador de empresa.		
Grado de satisfacción del interesado: 7	**Grado de insatisfacción del interesado:** 3	
Dependencias (qué requisitos depende de este): 01, 02, 05, 06, 08, 11.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 01/03/2017.	
Proyecto: Softour	**Analista:** Grupo Informático de Desarrollo de Software (GIDSoft)	
Identificador del Requisito: 03	**Tipo de Requisito:** Funcional	**Caso de Uso/Evento:** Agregar Transporte.
Descripción: El usuario registrado "Administrador de Empresa" agrega un transporte relacionado con un sitio de interés y con hospedaje con el fin de cargar las características o atributos relevantes del mismo (etiqueta, nombre, descripción, dirección física, URL, valoración, lugar de salida, hora de salida, lugar de llegada, hora de llegada, persona contacto, tipos de transporte) relacionados con el mismo, que serán almacenados en el sistema para futuras consultas.		
Justificación del Requisito: Es necesario registrar los transportes relacionados con los sitios de interés y los hospedajes para que los usuarios puedan visualizarlos, poder seleccionar entre múltiples opciones de transportes según el lugar a donde quiera ir el usuario final.		
Fuente (que interesado lo propone): Prestador del servicio.	**Unidad en la que se origina:** Administrador de empresa.	
Criterios de validación: Para registrar un transporte debe ser administrador de empresa.		
Grado de satisfacción del interesado: 7	**Grado de insatisfacción del interesado:** 3	
Dependencias (qué requisitos depende de este): 01, 02, 05, 06, 08, 11.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 01/03/2017.	
Proyecto: Softour	**Analista:** Grupo Informático de Desarrollo de Software (GIDSoft)	

Figura TT. 3. Plantilla de Volére Agregar Transporte, requisito 03.

Identificador del Requisito: 04	**Tipo de Requisito:** Funcional	**Caso de Uso/Evento:** Registrar Usuario
Descripción: El usuario final podrá registrarse indicando nombre (login), clave (password) y correo electrónico, creando un perfil identificable como turista para interactuar con el sistema, dicha información será almacenado en el sistema para futuras consultas.		
Justificación del Requisito: Es necesario brindar al usuario final la opción de registrarse para ser considerado usuario turista, el cual posee mayores privilegios de interacción con el sistema y los prestadores de servicios involucrados en el mismo, ya que todo usuario turista tendrá la capacidad de contactar a los prestadores de servicios e interactuar con las funcionalidades del sistema para comentar y calificar los sitios de interés.		
Fuente (que interesado lo propone): Prestador del servicio.	**Unidad en la que se origina:** Usuario Final.	
Criterios de validación: Para registrar un usuario turista debe poseer correo electrónico.		
Grado de satisfacción del interesado: 10	**Grado de insatisfacción del interesado:** 0	
Dependencias (qué requisitos depende de este): 05, 06, 07, 08, 11, 12.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 01/03/2017.	
Proyecto: Softour	**Analista:** Grupo Informático de Desarrollo de Software (GIDSoft)	

Figura TT. 4. Plantilla de Volére Registrar Usuario, requisito 04.

Identificador del Requisito: 05	**Tipo de Requisito: No** Funcional	**Caso de Uso/Evento:**
Descripción: Realizar comentario.		
Justificación del Requisito: El sistema deberá añadir comentarios usualmente con el propósito de hacer el código fuente más fácil de entender con vistas a su mantenimiento o reutilización.		
Fuente (que interesado lo propone): SOFTOUR	**Unidad en la que se origina:** SOFTOUR	
Criterios de validación:		
Grado de satisfacción del interesado: 7	**Grado de insatisfacción del interesado:** 5	
Dependencias (qué requisitos depende de este):	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 20/03/17.	
Proyecto: SOFTOUR	**Analista:** Grupo informático de desarrollo de software.	

Figura TT. 5. Plantilla de Volére Realizar Comentario, requisito 05.

Identificador del Requisito: 06	**Tipo de Requisito:** No Funcional	**Caso de Uso/Evento:**
Descripción: Generar Estadísticas.		
Justificación del Requisito: El sistema deberá facilitar la consulta de la información almacenada en intervalos semanales, mensuales y anuales.		
Fuente (que interesado lo propone): SOFTOUR		**Unidad en la que se origina:** SOFTOUR
Criterios de validación: Deben ingresar al sistema y registrarse.		
Grado de satisfacción del interesado: 6	**Grado de insatisfacción del interesado:** 5	
Dependencias (qué requisitos depende de este): 04.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 22/03/17.	
Proyecto: SOFTOUR.	**Analista:** Grupo Informático de Desarrollo de Software.	

Figura TT. 6. Plantilla de Volére Generar Estadísticas, requisito 06.

Identificador del Requisito: 07	**Tipo de Requisito:** No Funcional	**Caso de Uso/Evento:** Mostrar estadisticas
Descripción: Mostrar estadísticas.		
Justificación del Requisito: El sistema permitirá visualizar estadísticas generadas por medio de tablas y gráficos.		
Fuente (que interesado lo propone): SOFTOUR		**Unidad en la que se origina:** SOFTOUR
Criterios de validación:		
Grado de satisfacción del interesado: 8	**Grado de insatisfacción del interesado:** 5	
Dependencias (qué requisitos depende de este):	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 20/03/17.	
Proyecto: SOFTOUR	**Analista:** Grupo informático de desarrollo de software	

Figura TT. 7. Plantilla de Mostrar Estadísticas, requisito 07

Identificador del Requisito: 08	**Tipo de Requisito:** Funcional	**Caso de Uso/Evento:** Realizar búsqueda.
Descripción: El usuario registrado "Turista" realiza una búsqueda relacionada con un sitio de interés, hospedajes y transportes, que sea de manera versátil, con el fin de visualizar características o atributos relevantes de los mismo (etiqueta, nombre, descripción, dirección física, URL, valoración, cantidad de habitaciones, categoría, rutas de acceso, persona contacto, tipos de habitación, lugar de salida, hora de salida, lugar de llegada, hora de llegada, tipos de transporte) relacionados con el interés del mismo.		
Justificación del Requisito: Es necesario realizar búsquedas que permitan consultar los aspectos relacionados con los sitios de interés, hospedajes y transportes para que los usuarios puedan visualizarlos, creando cronogramas tentativos de actividades y poder seleccionar entre múltiples opciones según el lugar a donde quiera ir el usuario final.		
Fuente (que interesado lo propone): Prestador del servicio.	**Unidad en la que se origina:** Usuario registrado turista.	
Criterios de validación: Para realizar una búsqueda debe ser un usuario registrado turista.		
Grado de satisfacción del interesado: 5	**Grado de insatisfacción del interesado:** 5	
Dependencias (qué requisitos depende de este): 01, 02, 03, 06, 17, 20.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 01/03/2017.	
Proyecto: Softour	**Analista:** Grupo Informático de Desarrollo de Software (GIDSoft)	

Figura TT. 8. Plantilla de Volére Realizar Búsqueda, requisito 08

Identificador del Requisito: 09	**Tipo de Requisito:** No Funcional	**Caso de Uso/Evento:**
Descripción: Aumentar texto contemplado en sistema.		
Justificación del Requisito: El sistema deberá tener la opción de aumentar el texto en el sistema para permitirles a los usuarios tener una visualización adecuada del mismo, sin importar que resolución se utilice.		
Fuente (que interesado lo propone): SOFTOUR	**Unidad en la que se origina:** SOFTOUR	
Criterios de validación:		
Grado de satisfacción del interesado: 8	**Grado de insatisfacción del interesado:** 5	
Dependencias (qué requisitos depende de este):	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 20/04/17.	
Proyecto: SOFTOUR	**Analista:** Grupo Informático de Software.	

Figura TT. 9. Plantilla de Volére hacer Zoom, requisito 09.

Identificador del Requisito: 10	**Tipo de Requisito:** No Funcional	**Caso de Uso/Evento:**
Descripción: Establecer una interfaz acorde a los servicios turísticos.		
Justificación del Requisito: El sistema deberá establecer una interfaz que satisfaga las necesidades de los usuarios, teniendo en cuenta los diferentes perfiles a quienes va dirigido.		
Fuente (que interesado lo propone): SOFTOUR.		**Unidad en la que se origina:** SOFTOUR
Criterios de validación:		
Grado de satisfacción del interesado: 5	**Grado de insatisfacción del interesado:** 5	
Dependencias (qué requisitos depende de este):	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 20/03/17.	
Proyecto: SOFTOUR.	**Analista:** Grupo Informático de software.	

Figura TT. 10. Plantilla de Volére Generar Interfaz, requisito 10.

Identificador del Requisito: 11	**Tipo de Requisito:** Funcional	**Caso de Uso/Evento:** Modificar Información por administrador
Descripción: El administrador de empresa podrá modificar información relacionada con sitios de interés y prestadores de servicio, para mantener la información actualizada.		
Justificación del Requisito: Es necesario que el administrador de empresas modifique la información relacionada con los sitios de interés y los prestadores del servicio para mantener actualizado el sistema de información.		
Fuente (que interesado lo propone): Prestador del servicio.	**Unidad en la que se origina:** Administrador de empresa.	
Criterios de validación: Para modificar información referente al sitio de interés y prestadores de servicios debe ser administrador de empresa.		
Grado de satisfacción del interesado: 8	**Grado de insatisfacción del interesado:** 2	
Dependencias (qué requisitos depende de este): 01, 02, 03, 05, 06, 07, 08, 20.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 01/03/2017.	
Proyecto: Softour	**Analista:** Grupo Informático de Desarrollo de Software (GIDSoft)	

Figura TT. 11. Plantilla de Volére Modificar Información, requisito 11

.

Identificador del Requisito: 12	**Tipo de Requisito:** Funcional	**Caso de Uso/Evento:** Eliminar Información por administrador
Descripción: El administrador de empresa podrá eliminar información relacionada con sitios de interés y prestadores de servicio, que no esté vigentes y solventes, que deseen pasar a inactivos o salir del sistema, para mantener la información actualizada.		
Justificación del Requisito: Es necesario que el administrador de empresas elimine la información relacionada con los sitios de interés y los prestadores del servicio para mantener actualizado el sistema de información con material autorizado.		
Fuente (que interesado lo propone): Prestador del servicio.	**Unidad en la que se origina:** Administrador de empresa.	
Criterios de validación: Para eliminar información referente al sitio de interés y prestadores de servicios debe ser administrador de empresa.		
Grado de satisfacción del interesado: 8	**Grado de insatisfacción del interesado:** 2	
Dependencias (qué requisitos depende de este): 01, 02, 03, 05, 06, 07, 08, 11, 20.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 01/03/2017.	
Proyecto: Softour	**Analista:** Grupo Informático de Desarrollo de Software (GIDSoft)	

Figura TT. 12. Plantilla de Volére Eliminar Información, requisito 12

Identificador del Requisito: 13	**Tipo de Requisito:** No Funcional	**Caso de Uso/Evento:**
Descripción: Desarrollar la aplicación en lenguaje Javascript.		
Justificación del Requisito: Se utiliza ya que es un sencillo lenguaje de programación que gracias a su compatibilidad con muchos navegadores modernos se ha convertido en un estándar como lenguaje de programación del lado del cliente.		
Fuente (que interesado lo propone): SOFTOUR.	**Unidad en la que se origina:** SOFTOUR.	
Criterios de validación:		
Grado de satisfacción del interesado: 5	**Grado de insatisfacción del interesado:** 5	
Dependencias (qué requisitos depende de este):	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 20/03/17.	
Proyecto: SOFTOUR.	**Analista:** Grupo informático de desarrollo de Software.	

Figura TT. 13. Plantilla de Volére Codificar en .js, requisito 13.

Identificador del Requisito: 14	**Tipo de Requisito:** No Funcional	**Caso de Uso/Evento:**
Descripción: Desarrollar la aplicación utilizando el gestionador de base de datos mongo DB.		
Justificación del Requisito: Se utiliza ya que es sistema de base de datos NOSQL orientado a documentos, desarrollado bajo el concepto de código abierto, que permite una gran escalabilidad y un alto rendimiento.		
Fuente (que interesado lo propone): SOFTOUR.		**Unidad en la que se origina:** SOFTOUR.
Criterios de validación:		
Grado de satisfacción del interesado: 5	**Grado de insatisfacción del interesado:** 5	
Dependencias (qué requisitos depende de este):	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 20/04/17.	
Proyecto: SOFTOUR.	**Analista:** Grupo Informático de desarrollo de software.	

Figura TT. 14. Plantilla de Volére utilizar MongoDB, requisito 14.

Identificador del Requisito: 15	**Tipo de Requisito:** No Funcional	**Caso de Uso/Evento:**
Descripción: Realizar un sistema multiplataforma.		
Justificación del Requisito: El sistema debe ser capaz de ejecutarse en diversos navegadores web o sistemas operativos existentes.		
Fuente (que interesado lo propone): SOFTOUR		**Unidad en la que se origina:** SOFTOUR
Criterios de validación:		
Grado de satisfacción del interesado: 5	**Grado de insatisfacción del interesado:** 5	
Dependencias (qué requisitos depende de este):	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 20/04/17.	
Proyecto: SOFTOUR.	**Analista:** Grupo Informático de desarrollo de software.	

Figura TT. 15. Plantilla de Volére realizar sistema multiplataforma, requisito 15.

Identificador del Requisito: 16	**Tipo de Requisito:** No Funcional	**Caso de Uso/Evento:**
Descripción: Imprimir planillas estadísticas.		
Justificación del Requisito: El sistema estará en la capacidad de reportar datos estadísticos de todos los usuarios que ingresan al sitio mensual y anualmente.		
Fuente (que interesado lo propone): SOFTOUR		**Unidad en la que se origina:** SOFTOUR
Criterios de validación:		
Grado de satisfacción del interesado: 5		**Grado de insatisfacción del interesado:** 5
Dependencias (qué requisitos depende de este):		**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.
Documentos de soporte:		**Histórico de cambios:** 20/04/17.
Proyecto: SOFTOUR.		**Analista:** Grupo Informático de desarrollo de software.

Figura TT. 16. Plantilla de Volére Agregar Elaborar Estadísticas, requisito 16.

Identificador del Requisito: 17	**Tipo de Requisito:** Funcional	**Caso de Uso/Evento:** Identificar los por nombre y etiqueta
Descripción: El usuario final podrá identificar por nombre y etiqueta (montaña, playa, hotel, posada, centro comercial, cine, castillo, eventos, entre otros), para hacer búsquedas efectivas y dinámicas.		
Justificación del Requisito: Es necesario que el usuario final tenga a disposición mecanismos de búsquedas versátiles y efectivos, que muestren la información que realmente desea ver sin necesidad de excesiva navegación por un sitio jerárquico.		
Fuente (que interesado lo propone): Prestador del servicio.	**Unidad en la que se origina:** Usuario final	
Criterios de validación: Se deben manejar etiquetas con nombres claves.		
Grado de satisfacción del interesado: 8	**Grado de insatisfacción del interesado:** 2	
Dependencias (qué requisitos depende de este): 01, 02, 03, 05, 06, 07, 08, 20.	**Conflictos (qué requisitos son incompatibles o inconsistentes con este):** N/A.	
Documentos de soporte:	**Histórico de cambios:** 01/03/2017.	
Proyecto: SOFTOUR	**Analista:** Grupo Informático de Desarrollo de Software (GIDSoft)	

Figura TT. 17. Plantilla de Volére Búsquedas por Etiquetas, requisito 17.

Apéndice UU. Diagrama de Clases Persistentes.

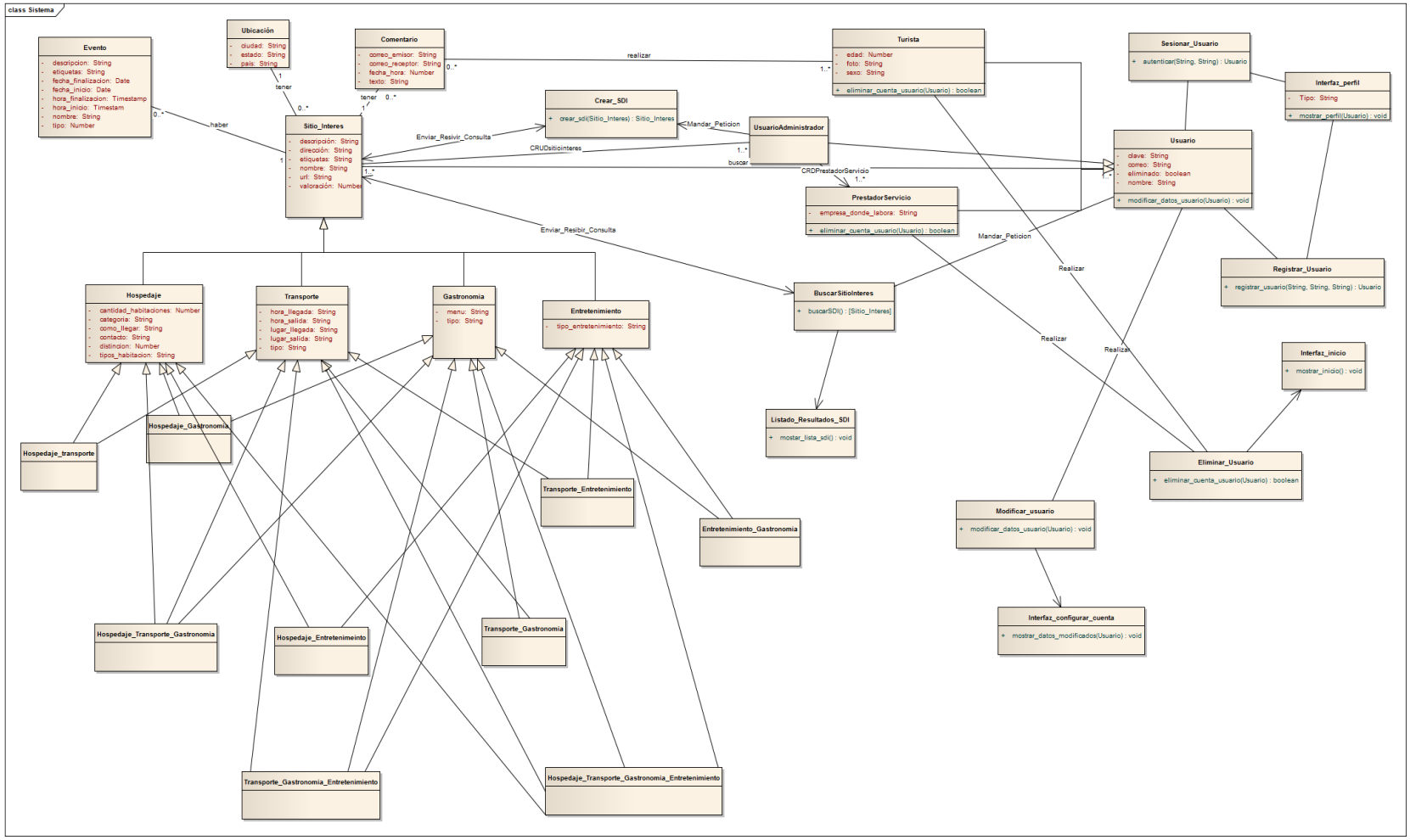


Figura UU. 1 Diagrama de clases persistentes.

Apéndice VV. Diagrama de datos.

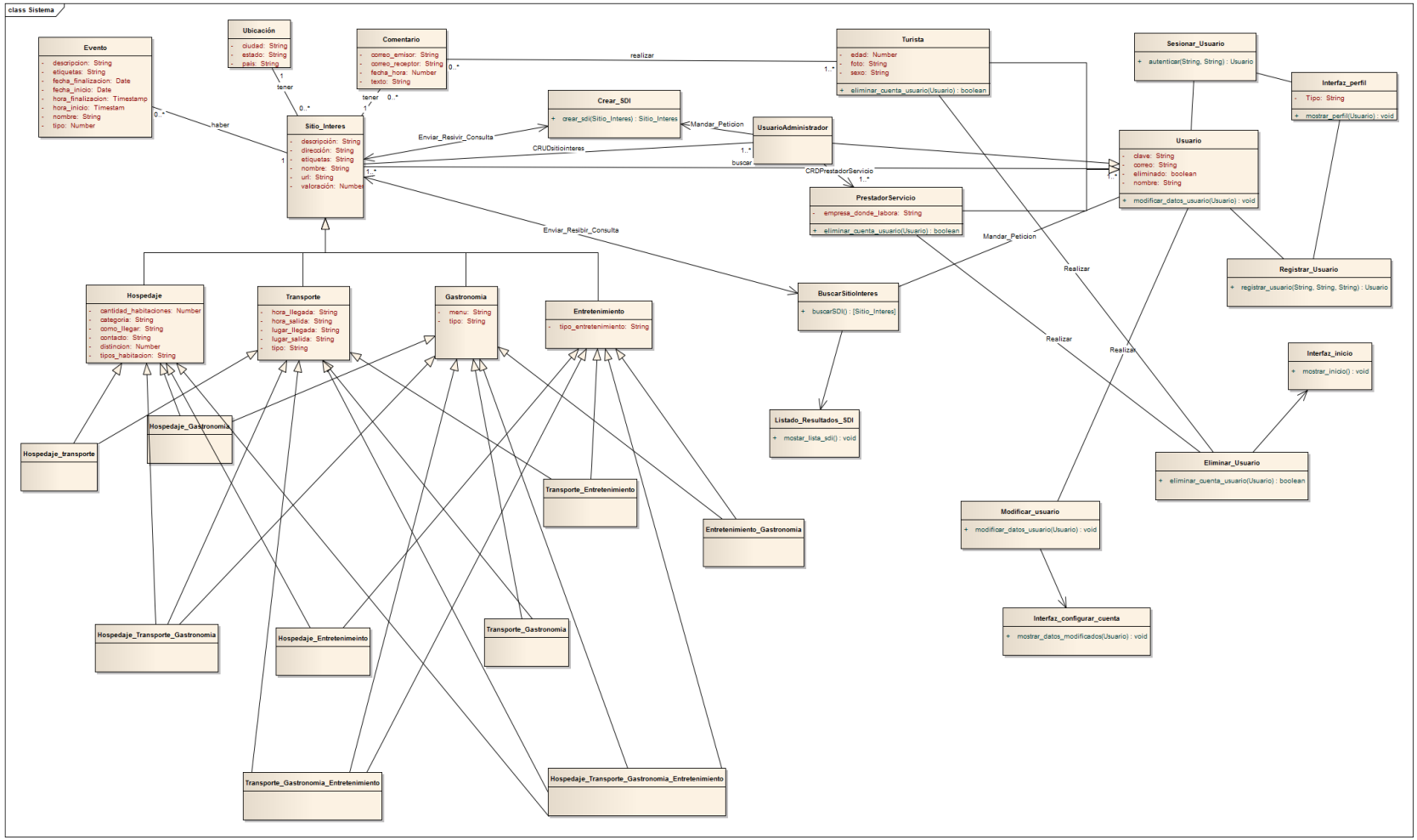


Figura VV. 1. Colecciones de datos general

Apéndice WW. Diagrama de componentes.

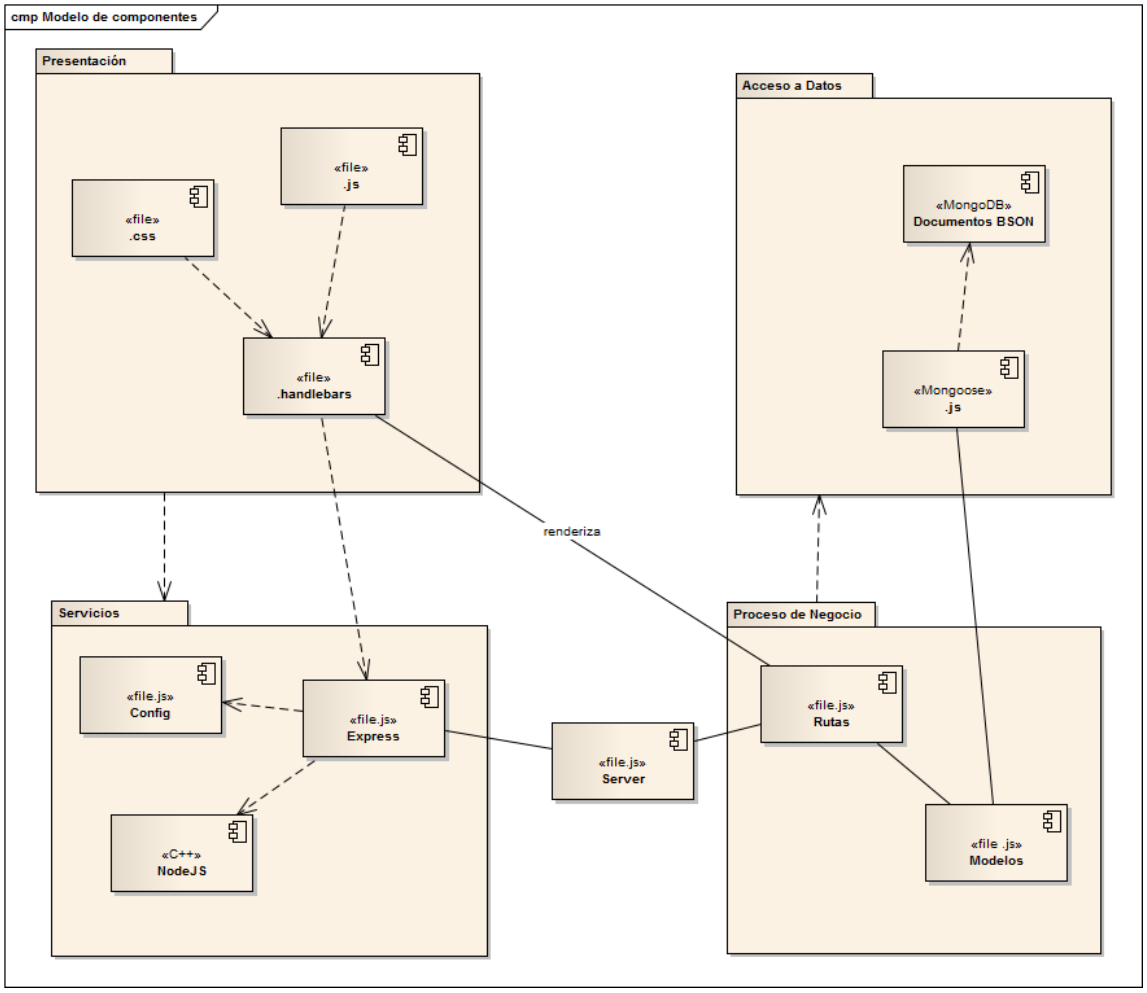


Figura WW. 1. Diagrama de componentes.

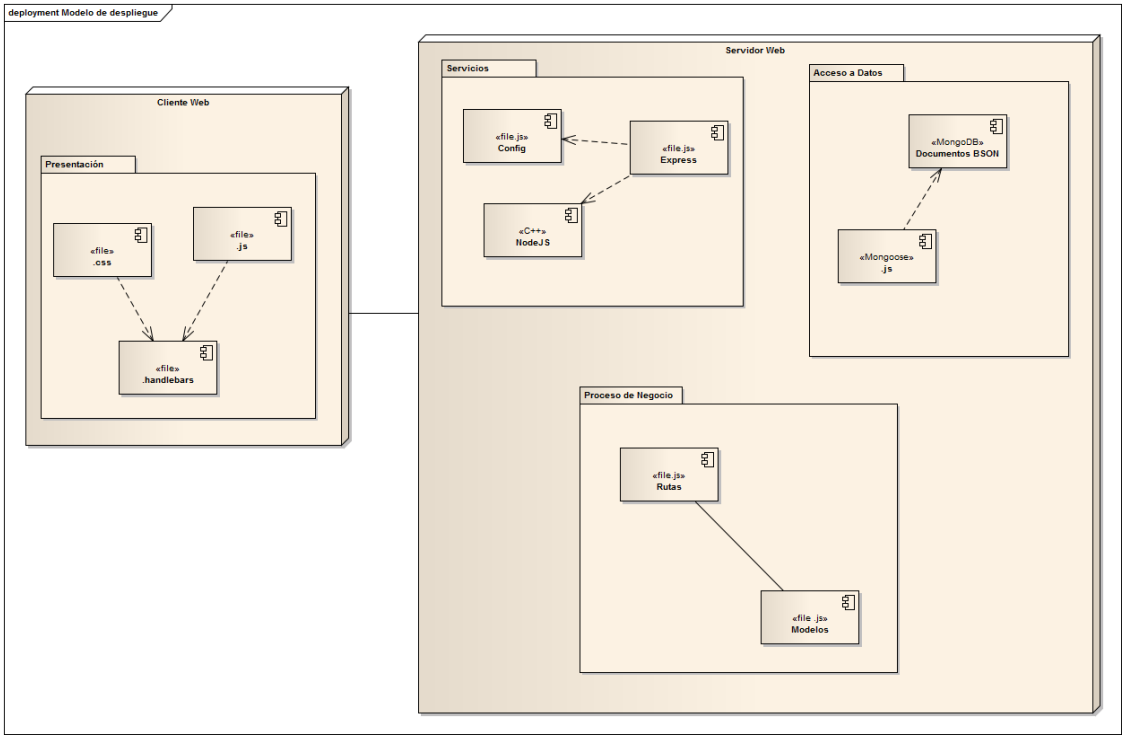


Figura WW. 2. Diagrama de componentes.

Apéndice XX. Diagrama de reglas del negocio.

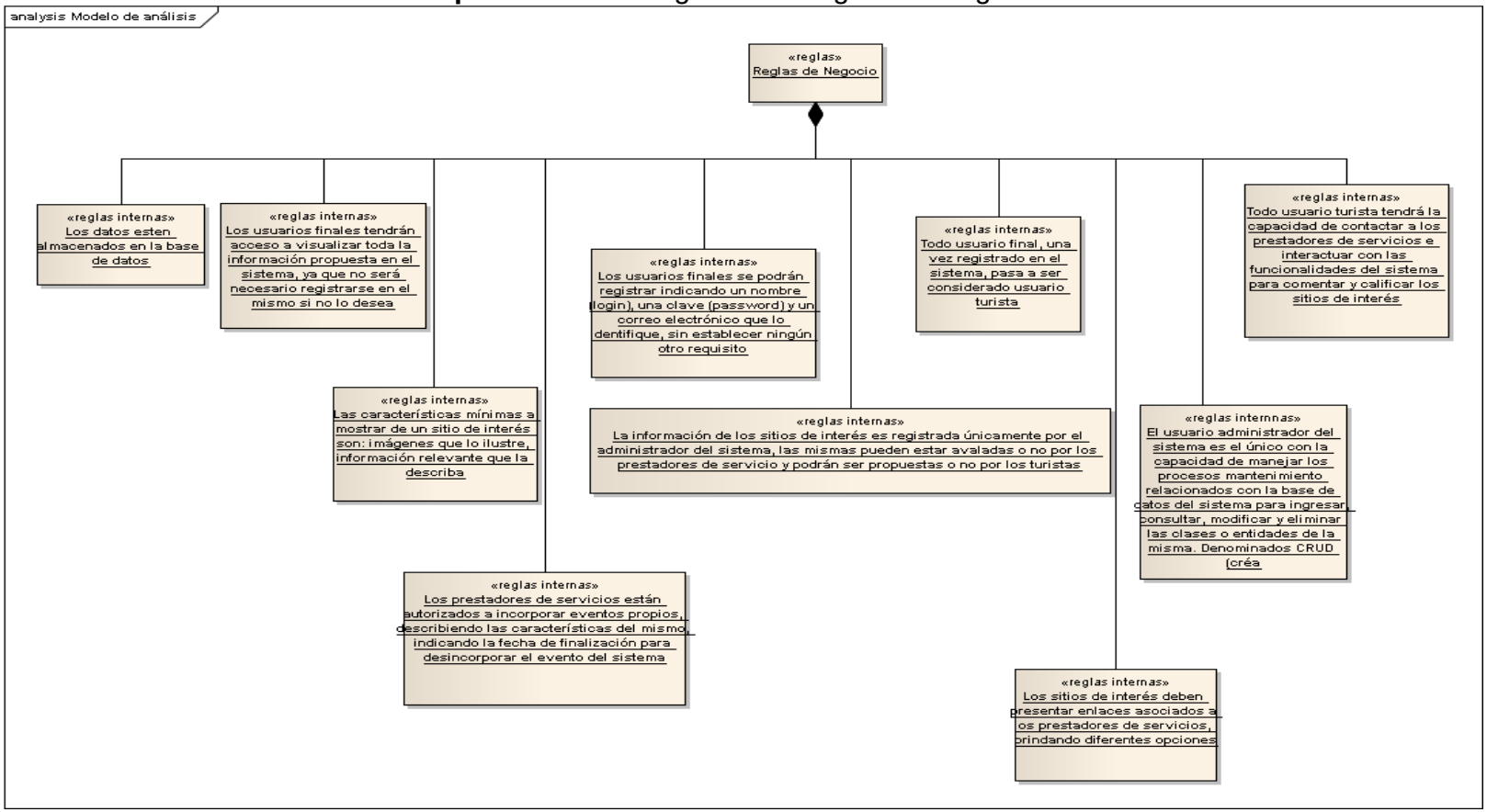


Figura XX. 1. Diagrama de reglas del negocio

HOJAS DE METADATOS

Hoja de Metadatos para Tesis y Trabajos de Ascenso – 1/6

Título	Sistema De Información Bajo Ambiente Web Para El Incentivo, Promoción Y Apoyo Al Sector Turismo

Autor(es)

Apellidos y Nombres	**Código CVLAC / e-mail**	
Goméz A., Arnella J.	**CVLAC**	15.743.977
	e-mail	**Arnella.serman@gmail.com**
	e-mail	
González R., Jhoendris del C.	**CVLAC**	**22.921.975**
	e-mail	**Jhoegonzalez10@gmail.com**
	e-mail	
Tena R., Luis J	**CVLAC**	**19.756.821**
	e-mail	**Luistena.developer@hotmail.com**
	e-mail	
Contasti R., Manuel J.	**CVLAC**	**20.345.775**
	e-mail	**macontasti@gmail.com**
	e-mail	
Marín M., Marcos J.	**CVLAC**	**19.980.293**
	e-mail	**Marcosjmarinm@gmail.com**
	e-mail	
Carmona S., Miguel S.	**CVLAC**	**19.761.045**
	e-mail	**Mcarmona606@gmail.com**
	e-mail	

Palabras o frases claves:

Palabras Claves: Turismo, Sistema de Información, Método Watch, Node.js, MongoDB, Lugares, Eventos, Experiencias, Alojamientos, Gastronomía.

Hoja de Metadatos para Tesis y Trabajos de Ascenso – 2/6
Líneas y sublíneas de investigación:

Área	Subárea
CIENCIAS	Informática

Resumen (abstract):

Se desarrolló un sistema de información Web para el incentivo, promoción y apoyo al sector turismo, caso de estudio Softour, ofreciendo las funcionalidades para los perfiles de usuarios administrador, prestador de servicios y turista, con aspectos relevantes para la gestión y aprovechamiento de regiones y países en las que se deseen resaltar lugares, eventos, experiencias, alojamientos y gastronomías, brindando mecanismos de búsquedas avanzadas, una interfaz visual acorde a los requerimientos de los nuevos paradigmas de las aplicaciones turísticas, creando una asociación entre el prestador del servicio y el turista de ganar/ganar, manejando la retroalimentación de la calificación y el compartir las experiencias vividas, optimizando los tiempos de respuesta a través del manejo de colecciones de datos, programación versátil en .js. Se tomó como referencia la metodología WATCH (Montilva, Barrios y Rivero, 2008) en su instanciación del método *Blue WATCH* para la implementación de los procesos, productos y actores (Montilva y Barrios, 2014). Durante el ciclo de aplicación se realizó un Modelado de Negocios empleando notación UML, el Desarrollo de Requisitos, el Diseño Arquitectónico; en el ciclo de versión se ejecutaron actividades del ciclo anterior anexando el Diseño Detallado; finalmente, en el ciclo de incremento se realizó la construcción de las interfaces utilizando herramientas como Bootstrap, HTML, CSS, JQuery, Handlebars y JavaScript; la creación de la base de datos en MongoDB orientada hacia las colecciones de datos no estructuradas, conjuntamente con la librería de Mongoose para asociarla hacia la orientación a objetos del modelo; la codificación bajo el lenguaje de programación Node.js, Sublime y Postman; para el control de versiones Git y trabajo colaborativo en equipo GitHub. Cabe destacar que el ciclo de desarrollo del proyecto fueron controlados y administrados mediante los procesos de gestión y soporte compuestos por la gestión de proyecto, de riesgos, de requisitos, de la configuración, planificación y control de versiones y el plan de verificación y validación, en los cuales se resaltó el uso de las herramientas Trello, Scrum y Slack. El sistema obtenido ofrece a las organizaciones turísticas, los prestadores de servicios, los destinos turísticos y a los turistas, la intermediación, asociación y capacidad para incentivar, promocionar y apoyar al sector turismo, utilizando herramientas de punta, para conectarlas con el universo virtual y hacerlas más competitivas, adentrando en conceptos de innovación y calidad, sin olvidar la sencillez en el acceso, el uso y la comprensión de lo que se ofrece. Se consideró inicialmente un caso de prueba para el Estado Sucre pero todas las funcionalidades descritas en la presente investigación pueden ser adaptadas a otras zonas turísticas ya que no existen limitaciones considerables y el sistema se desarrolló con una visión de uso universal, con la posibilidad de visualizar la aplicación en los idiomas español e inglés.

Hoja de Metadatos para Tesis y Trabajos de Ascenso – 3/6

Contribuidores:

Apellidos y Nombres	ROL / Código CVLAC / e-mail	
Betancourt., Eugenio	**ROL**	**CA** [ ] **AS** [x] **TU** [ ] **JU** [ ]
	CVLAC	**13.729.842**
	e-mail	
	e-mail	
Aguiar., Dianelina	**ROL**	**CA** [ ] **AS** [x] **TU** [ ] **JU** [**x**]
	CVLAC	**11.831.828**
	e-mail	
	e-mail	
Fernández., Lisbeth	**ROL**	**CA** [ ] **AS** [ ] **TU** [ ] **JU** [**x**]
	CVLAC	**9.979.836**
	e-mail	
	e-mail	
Acuña, Leopoldo.	**ROL**	**CA** [ ] **AS** [ ] **TU** [ ] **JU** [**x**]
	CVLAC	**9.976.457**
	e-mail	

Fecha de discusión y aprobación:

Año	Mes	Día
2018	**01**	**25**

Lenguaje: **SPA**

Hoja de Metadatos para Tesis y Trabajos de Ascenso – 4/6

Archivo(s):

Nombre de archivo	Tipo MIME
Curso Especial de Grado-GomezMarinGonzalezContastiCarmonaTena.docx	**Aplication/word**

Alcance:

Espacial: Universal

Temporal: Intemporal

Título o Grado asociado con el trabajo: Licenciado en Informática

Nivel Asociado con el Trabajo: Licenciado

Área de Estudio: Sistemas de Información Empresariales (SIE)

Institución(es) que garantiza(n) el Título o grado: Universidad de Oriente

UNIVERSIDAD DE ORIENTE
CONSEJO UNIVERSITARIO
RECTORADO

CU Nº0975

Cumaná, 04 AGO 2009

Ciudadano
Prof. JESÚS MARTÍNEZ YÉPEZ
Vicerrector Académico
Universidad de Oriente
Su Despacho

Estimado Profesor Martínez:

Cumplo en notificarle que el Consejo Universitario, en Reunión Ordinaria celebrada en Centro de Convenciones de Cantaura, los días 28 y 29 de julio de 2009, conoció el punto de agenda ***"SOLICITUD DE AUTORIZACIÓN PARA PUBLICAR TODA LA PRODUCCIÓN INTELECTUAL DE LA UNIVERSIDAD DE ORIENTE EN EL REPOSITORIO INSTITUCIONAL DE LA UDO, SEGÚN VRAC Nº 696/2009".***

Leído el oficio SIBI – 139/2009 de fecha 09-07-2009, suscrita por el Dr. Abul K. Bashirullah, Director de Bibliotecas, este Cuerpo Colegiado decidió, por unanimidad, autorizar la publicación de toda la producción intelectual de la Universidad de Oriente en el Repositorio en cuestión.

Comunicación que hago a usted a los fines consiguientes.

UNIVERSIDAD DE ORIENTE
SISTEMA DE BIBLIOTECA
RECIBIDO POR
FECHA 5/8/09 HORA 5:30

Cordialmente,

JUAN A. BOLAÑOS CURIEL
Secretario

REPÚBLICA BOLIVARIANA DE VENEZUELA * UNIVERSIDAD DE ORIENTE * SECRETARÍA CONSEJO UNIVERSITARIO

C.C: Rectora, Vicerrectora Administrativa, Decanos de los Núcleos, Coordinador General de Administración, Director de Personal, Dirección de Finanzas, Dirección de Presupuesto, Contraloría Interna, Consultoría Jurídica, Director de Bibliotecas, Dirección de Publicaciones, Dirección de Computación, Coordinación de Teleinformática, Coordinación General de Postgrado.

JABC/YGC/maruja

Hoja de Metadatos para Tesis y Trabajos de Ascenso- 6/6

Artículo 41 del REGLAMENTO DE TRABAJO DE PREGRADO (vigente a partir del II Semestre 2009, según comunicación CU-034-2009) : "los Trabajos de Grado son de la exclusiva propiedad de la Universidad de Oriente, y sólo podrán ser utilizados para otros fines con el consentimiento del Consejo de Núcleo respectivo, quien deberá participarlo previamente al Consejo Universitario para su autorización".

Goméz A., Arnella J.
Autor

González R., Jhoendris del C.
Autor

Tena R., Luis J
Autor

Contasti R., Manuel J.
Autor

Marín M., Marcos J.
Autor

Carmona S., Miguel S.
Autor

Prof. Betancourt., Eugenio
Asesor

Prof. Aguiar, Dianelina
Asesor

Prof. Acuña, Leopoldo
Asesor

Prof. Fernández., Lisbeth
Asesor

Nota: Esta hoja debe ser firmada para ser anexada en el formato Digital. (Scanear)