UNIVERSIDAD DE ORIENTE
NÚCLEO DE ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

"DESARROLLO DE UN SOFTWARE QUE PERMITA LA AUTOMATIZACIÓN DE LAS ACTIVIDADES ADMINISTRATIVAS ASOCIADAS A LA SALA DE LECTURA DEL DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS DEL NÚCLEO DE ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"

REALIZADO POR:

Amer M., Sair Daniel

Saavedra S., Oscar Rafael

Trabajo de grado presentado como requisito parcial para optar al Título de

INGENIERO EN COMPUTACIÓN

Barcelona, Diciembre de 2009

UNIVERSIDAD DE ORIENTE
NÚCLEO DE ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

"DESARROLLO DE UN SOFTWARE QUE PERMITA LA AUTOMATIZACIÓN DE LAS ACTIVIDADES ADMINISTRATIVAS ASOCIADAS A LA SALA DE LECTURA DEL DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS DEL NÚCLEO DE ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"

ASESOR:

_____

Ing. Julima Anato
Asesor Académico

Barcelona, Diciembre de 2009

UNIVERSIDAD DE ORIENTE

NÚCLEO DE ANZOÁTEGUI

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

"DESARROLLO DE UN SOFTWARE QUE PERMITA LA AUTOMATIZACIÓN DE LAS ACTIVIDADES ADMINISTRATIVAS ASOCIADAS A LA SALA DE LECTURA DEL DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS DEL NÚCLEO DE ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE"

JURADO CALIFICADOR:

_____

Ing. Julima Anato
Asesor Académico

_____ _____

Ing. Claudio Cortínez
Jurado Principal

Ing. Aquiles Torrealba
Jurado Principal

Barcelona, Diciembre de 2009

Resolución

ARTÍCULO Nº 41 Del Reglamento de Trabajo de Grado

"Los trabajos de grado son de exclusiva propiedad de la Universidad y sólo podrán ser utilizados para otros fines con el conocimiento del Consejo de Núcleo respectivo, quién lo participará al Consejo Universitario".

Resumen

La Sala de Lectura del Departamento de Computación y Sistemas de la Universidad de Oriente sirve como fuente de información más especializada que la ofrecida por la biblioteca central a todos los estudiantes que hacen vida en el Departamento de Computación y Sistemas. En ella se realizan una cantidad de procesos administrativos para el manejo de la información de los estudiantes, así como de los libros y tesis que en ella se encuentran. Dichos procesos se realizan en la actualidad de manera manual, lo cual conlleva una dificultad mayor, así como también una mayor inversión de tiempo y esfuerzo por parte de los ayudantes y colaboradores de la sala. Debido al incremento en la cantidad de estudiantes que ingresan cada semestre al Departamento se hace imperiosa la necesidad de automatizar el manejo de los procesos administrativos que se realizan en la Sala de Lectura mediante la construcción e implementación de un software, para así optimizar su funcionamiento y maximizar su calidad de servicio para los estudiantes. Este software será desarrollado utilizando Java como lenguaje de programación y PostgreSQL como sistema gestor de base de datos.

AGRADECIMIENTOS

Primeramente a Dios, ya que sin el nada es posible.

A mis Padres, Irma Josefina Morales Silva y Saul Nobory Amer. Su trabajo incansable es una gran inspiración y ejemplo a seguir para mí; y sin su apoyo incondicional no habría cumplido jamás tantas metas.

A mi novia Arianna Ramos, a quien amo con todo mi cuerpo y alma. Gracias por todo tu apoyo en los momentos que más lo necesite.

A mi asesor, la profesora Julima Anato, por toda la ayuda y colaboarción prestada.

A la profesora Zulirais Garcia, por su guía en el tema de UML y RUP.

A todos los que hicieron posible este gran logro.

Sair Daniel Amer Morales

AGRADECIMIENTOS

A Dios y a la Virgen porque sin ellos nada de esto hubiese sido posible.

A mis padres por su apoyo incondicional durante toda mi carrera.

A mi novia, por apoyarme cuando más lo necesitaba.

A mis amigos y compañeros de clase por todas las cosas buenas y malas que compartimos.

A los profesores y amigos con los que tuve el placer de aprender tantas cosas buenas de esta carrera.

Oscar Rafael Saavedra Salazar

INDICE

INDICE DE TABLAS

INDICE DE FIGURAS

Capítulo I: Introducción

1.1 Planteamiento del problema

Los fundamentos y principios de los seres humanos, se basan en conocimientos adquiridos por sus propias experiencias, la educación es el reflejo de estos conocimientos. El desarrollo adecuado de una sociedad bien organizada depende de una educación satisfactoria de todos los individuos que en ella se desenvuelven, para ello es de suma importancia el poder contar con fuentes de información y cultura que permitan promover el aumento de la sabiduría y de los conocimientos de todos los eslabones sociales.

El recinto que guarda los valores textuales de la sabiduría adquirida por los seres humanos durante su evolución es la biblioteca, y, en menor medida, las salas de lectura independientes de cada departamento.

La Sala de Lectura del Departamento de Computación y Sistemas de la Universidad de Oriente, Núcleo Anzoátegui fue creada a finales de los años 90, y sirve como fuente de información más especializada que la ofrecida por la biblioteca central a todos los estudiantes que hacen vida en el Departamento de Computación y Sistemas. En ella se realizan una cantidad de procesos administrativos para el manejo de la información de los estudiantes, así como de los libros y tesis que en ella se encuentran.
Entre algunos de los procesos antes mencionados se encuentran los siguientes:

- **Inscripción de estudiantes y tesistas como usuarios de la sala.**
- **Gestión de préstamos internos de libros y tesis.**
- **Gestión de préstamos circulantes de libros y tesis.**
- **Inventario de libros y trabajos de grado.**
- **Control de asistencia de los ayudantes de la sala.**

Dichos procesos se realizan en la actualidad de manera manual, lo cual conlleva una dificultad mayor, así como también una mayor inversión de tiempo y esfuerzo por parte de los ayudantes y colaboradores de la sala.

Debido al incremento en la cantidad de estudiantes que ingresan cada semestre al Departamento se hace imperiosa la necesidad de automatizar el manejo de los procesos administrativos que se realizan en la Sala de Lectura, para así optimizar su funcionamiento y maximizar su calidad de servicio para los estudiantes.

Cabe destacar que no se tiene conocimiento de una iniciativa similar a la que aquí se presenta, por lo que constituye una innovación, que además resulta ser más que necesaria.

El propósito de este proyecto es diseñar e implementar un software que facilite al máximo la realización de los procesos administrativos realizados en la Sala de Lectura.

La importancia de este proyecto radica en el hecho de que actualmente todos los procesos realizados en la sala de lectura del departamento de Computación y Sistemas de la Universidad de Oriente, Núcleo de Anzoátegui se realiza de manera manual, por lo que la implantación de este sistema permitiría una mejora considerable en el rendimiento de la sala, asi como una mayor capacidad de crecimiento en un futuro cercano.

El alcance de este proyecto contempla el diseño e implementación de un software que automatiza las actividades administrativas realizadas por los ayudantes en la sala de lectura del departamento de Computación y Sistemas.

Los datos que se utilizarón durante el desarrollo del proyecto fueron obtenidos mediante la realización de entrevistas no estructuradas a los ayudantes y colaboradores de la sala de lectura del departamento de Computación y Sistemas. Tambien se obtuvieron datos mediante la observación directa del funcionamiento de la sala de lectura.

Esta trabajo de grado esta estructurado de la siguiente manera:

- **Capítulo 1 Introducción: Presenta una descripción de la situación actual de la sala de lectura, la descripción del problema que se plantea resolver, y los objetivos planteados para este proyecto.**

- **Capítulo II Marco Teórico: En este capítulo se presentan los diversos trabajos de grado que sirven de antecedente para este proyecto, así como también conceptos que sirven de basamento teórico para la realización del sistema.**

- **Capítulo III Fase de Início: Aquí se incluye el análisis a profundidad del funcionamiento actual de la sala, realizado mediante entrevistas no estructuradas y observación directa; a partir de las cuales se generaron los diagramas UML de Domínio y Caso de Usos, que permiten una mejor descripción del problema estudiado y los requisitos funcionales a tener en cuenta para el desarrollo del software.**

- **Capítulo IV Fase de Elaboración: En esta fase se procede al diseño del sistema utilizando los diagramas UML de clase de Diseño, de Secuencia, de Capas y de Componentes. Tambien se diseñan los prototipos de interfaces gráficas de usuario y una primera fase de implementación de algunos paquetes del sistema.**

- **Capítulo V Fase de Construcción: Dentro de este capítulo se incluyen las especificaciones de las herramientas escogidas para el desarrollo del sistema. Tambien se incluyen las pruebas realizadas al sistema y los resultados de dichas pruebas.**

- **Capítulo VI Fase de Transición: en este capítulo se muestran las herramientas desarrolladas para facilitar el uso del sistema por parte de los usuarios, particularmente el manual de usuario de la aplicación.**

1.2 Objetivos

1.2.1 Objetivo General

Desarrollar un software que permita la automatización de las actividades administrativas asociadas a la Sala de Lectura del Departamento de Computación y Sistemas del Núcleo de Anzoátegui de la Universidad de Oriente.

1.2.2 Objetivos Específicos

1. Analizar las normas y procedimientos de los procesos asociados a la Sala de Lectura del Departamento de Computación y Sistemas.
2. Realizar la descripción y especificación de los requerimientos del sistema.
3. Diseñar los distintos módulos que estructuran la aplicación y las bases de datos relacionales.
4. Efectuar la codificación de los módulos diseñados.
5. Realizar la integración y documentación de todos los módulos que forman la aplicación.
6. Realizar las pruebas de unidad, integración y validaciones del sistema.
7. Implementar el sistema en la Sala de Lectura del Departamento de Computación y Sistemas.

Capítulo II: Marco Teórico

2.1 Antecedentes

Los antecedentes de este trabajo son proyectos que presentan métodos para automatizar procesos que involucran el manejo de data de diversas índoles en distintas instituciones.

2.1.1 "Desarrollo e implementación de los servicios académicos del departamento de computación y sistemas usando tecnología www".

Resumen: Trabajo realizado como requisito parcial para obtener el título de Ingeniero en Computación en la Universidad de Oriente, Núcleo de Anzoátegui, en el que se utilizó UML para el análisis y diseño orientado a objeto.[1]
Autor: Guevara, J.
Institución: Universidad de Oriente, Núcleo de Anzoátegui, Departamento de Computación y Sistemas, Escuela de Ingeniería y Ciencias Aplicadas.
Nivel: Pregrado.
Fecha: 1998.

2.1.2 "Desarrollo de un sistema homologado de control de estudio de la universidad de oriente (sishoc)".

Resumen: Trabajo de grado realizado en la Universidad de Oriente, Núcleo Sucre. El programa fue diseñado para manejar todos los procesos académicos que involucran al departamento de control de estudio y de realizar inscripciones a los estudiantes de la Universidad de Oriente, Núcleo Sucre. [2]
Autor: Caldarello, D.
Institución: Universidad de Oriente, Núcleo de Sucre.
Nivel: Pregrado.

Fecha: 1999

2.1.3 “Desarrollo de un software como soporte para la automatización de las actividades administrativas que se llevan acabo en una institución educativa”.

Resumen: Trabajo de grado realizado en la Universidad de Oriente, Núcleo de Anzoátegui. El programa fue diseñado para automatizar los procesos tales como: inscripción de alumnos, facturación, contratación de personal, pago de nóminas, relación de ingresos vs. egresos, entre otros. [3]

Autor: Arcila, A., Sacarias, E.
Institución: Universidad de Oriente, Núcleo de Anzoátegui.
Nivel: Pregrado.
Fecha: 2006.

2.1.4 “Desarrollo de un software que permita la automatización de las actividades asociadas al departamento de admisión y control de estudios de la extensión centro/sur del núcleo de Anzoátegui de la universidad de oriente”.

Resumen: Proyecto de investigación en el cual se desarrolló un software que permite gestionar los datos en el departamento del control de estudios de la Universidad de Oriente, extensión región centro/sur. [4]
Autor: Millan, L., Garelli, L.
Institución: Universidad de Oriente, Núcleo de Anzoátegui.
Nivel: Pregrado.
Fecha: 2007.

2.2 Fundamentos Teóricos

2.2.1 Programación orientada a objetos

La programación orientada a objetos tiene la ventaja de ser un paradigma natural donde se pueden programar sistemas. Los seres humanos perciben el mundo como si estuvieran formado por objetos: mesas, sillas, computadores, coches, partidos de futbol, etc. También es un instinto humano intentar organizar esos objetos disponiéndolos de una forma concreta, optando por destacar determinadas características de algunos objetos que los destacan de otros.

Los niveles de dicha categoría y los métodos de clasificación de objetos en el mundo son infinitos. La forma en que las personas clasifican las cosas depende en gran medida de lo que deseen hacer con ellas y las características que mas le llamen la atención. A la vez que se agrupan los objetos atendiendo a esquemas de clasificación, también se tiende a resaltar determinados atributos de objetos mostrando su preferencia sobre otros.

Esta idea de crear jerarquías de objetos relacionados se utiliza en la programación orientada a objetos. En 1960, los investigadores ya observaron que muchas de las entidades del modelo de programas de computadora se podían nombrar y que se podían describir sus propiedades y comportamientos.

La programación orientada a objetos se puede describir rápidamente como identificación de objetos importantes, su organización en jerarquías, la adición de atributos a los objetos que describen características relevantes en el contexto del problema y de adición de las funciones (métodos) a los objetos para realizar las tareas necesarias en el objeto. Los detalles son algo mas complejo, pero lo fundamental es que se trata de un proceso simple y natural.

No obstante, que sea un proceso simple y natural no significa que sea sencillo, ya que un conjunto de objetos se podría clasificar de muchas formas distintas. La clave es la posibilidad de identificar los atributos importantes de objetos y formar abstracciones y jerarquías idóneas. Incluso en el contexto de un dominio problemático, a

veces resulta bastante difícil determinar los niveles correctos de abstracción y jerarquías de clasificación correcta. Simplemente, la decisión de la clase o grupo al que un objeto pertenece puede ser una tarea bastante difícil. [5]

2.2.2 Características de los lenguajes de programación orientada a objetos

Los lenguajes de programación orientada a objetos (como C++, C# y Java) se caracterizan por cuatro conceptos claves: encapsulación, herencia, polimorfismo y Abstracción, que son compatibles con este aspecto natural de identificación y clasificación de objetos. [5]

2.2.2.1 Encapsulación

La encapsulación facilita la comprensión de los grandes programas; la ocultación de datos les hace más eficaces.

Los objetos pueden interactuar con otros objetos sólo a través de los atributos y métodos del objeto que se muestran públicamente. Cuantos mas atributos y métodos se muestren públicamente, mas difícil será modificar la clase sin que ello afecte al código que utiliza la clase. Una variable oculta se podría cambiar de un long a un double, sin que ello afecte al código que utilicen los objetos creados (instanciados) de esa clase. El programador sólo se debería preocupar por los métodos en las clase que han tenido acceso a esa clase, en lugar de por todos los sitios en el programa que un objeto ha instanciados desde los que se puede llamar a la clase.

2.2.2.2 Herencia

La herencia proporciona dos ventajas evidentes a los programas. La primera, y mas importante, es que permite crear jerarquías que expresen las relaciones entre los diferentes tipos.

La segunda ventaja es que las clases pueden heredar características más generales de las clases superiores dentro de la jerarquía. En lugar de desarrollar nuevas clases a partir de cero, las clases nuevas podrán heredar las funciones de las clases

existentes y, a continuación, modificar o ampliar esta función. La clase principal desde la que la nueva clase hereda las propiedades se conoce como la clase básica y la nueva clase se denomina la clase derivada.

2.2.2.3 Polimorfismo

Es una palabra que significa múltiples formas y es una de las características más importantes de la programación orientada a objetos. En realidad, polimorfismo es la propiedad por la que a la misma palabra se le asignan múltiples definiciones.

En esencia, polimorfismo es la capacidad para enviar el mismo mensaje a objetos totalmente diferentes, cada uno de los cuales responde a ese mensaje de un modo específico.

2.2.2.4 Abstracción

Una de las consideraciones más importantes para tratar el problema de la complejidad del software es el concepto de abstracción. La idea básica de la abstracción es reducir el nivel de primitivas o representaciones básicas necesarias para producir un sistema de software. De manera sencilla esto se logra mediante el uso de lenguajes de programación que contengan estructuras de datos de alto nivel. En otras palabras, la pregunta opuesta sería: ¿por qué no programar en código binario, o sea 0s y 1s ? La respuesta es que ninguna persona sería capaz de comprender una aplicación al verse el código y por otro lado requeriría de programas extremadamente extensos para representar la aplicación completa dada la simplicidad de la primitiva básica.

Los sistemas de software construidos con lenguajes de programación de más alto nivel reducen el número total de líneas de código por lo tanto reducen su complejidad. Con la programación orientada a objetos se definen dos niveles de abstracción. El nivel más alto, el de los objetos, es utilizado para describir la aplicación mientras que el nivel más bajo, el de los datos y las funciones, es utilizado para describir sus detalles.

Este nivel inferior corresponde al único nivel de la programación tradicional. Esto refleja que la complejidad se maneja de mejor manera con la tecnología orientada a

objetos. En general cuanto más podamos simplificar las tareas de desarrollo mejor será el manejo de la complejidad. Por otro lado el objeto como estructura básica sirve para separar el “qué” de una aplicación del “cómo”, o sea sus detalles, al contrario de la programación tradicional donde el “qué” y el “cómo” se resuelven a la vez.

2.2.3 Clases y Objetos

Las clases son estructuras o plantillas que sirven para definir un objeto. Una clase de un objeto contiene una colección de métodos y definiciones de datos. Si se diseña una clase que representa a un cliente no se ha creado un objeto.

Una clase describe la constitución de un objeto y sirve como plantilla para construir objetos, especificando la interfaz pública de un objeto. Una clase tiene un nombre y especifica los miembros que pertenecen a la clase, que pueden ser campos (datos) y métodos (procedimientos).

Un objeto es una colección de datos y las subrutinas o métodos que operan sobre ellas, los objetos representan cosas físicas o abstractas, pero que tienen un estado y un comportamiento. Así ciertas propiedades definen a un objeto, y ciertas propiedades definen lo que hacen.

Un objeto es una instancia (ejemplar, caso) de una clase y, por consiguiente, puede, naturalmente, haber muchos objetos de una clase. [5]

2.2.4 Java

Java es un lenguaje de programación totalmente orientado a objetos y un entorno para ejecución de programas escritos en el lenguaje Java. Al contrario de los compiladores tradicionales, que convierten el código fuente en instrucciones a nivel de máquina, el compilador Java traduce el código fuente java en instrucciones que son interpretadas por la Máquina Virtual Java (JVM, *Java Virtual Machine*). A diferencia de C y C++ en los que está inspirado, Java es un lenguaje interpretado. [6]

2.2.5 Características de Java

Java ha conseguido una enorme popularidad. Su rápida difusión e implantación en el mundo de la programación en Internet y fuera de ella (*off line*) ha sido posible gracias a sus importantes características. Los creadores de Java escribieron un artículo, ya clásico, en el que definían al lenguaje como sencillo, orientado a objetos, distribuido, interpretado, robusto, seguro, arquitectura neutra, alto rendimiento, multihilo y dinámico. Analicemos detenidamente cada característica. [6]

2.2.5.1 Sencillo

Los lenguajes de programación orientados a objetos no son sencillos ni fáciles de utilizar, pero Java es un poco más fácil que el popular C++, lenguaje de desarrollo de software mas popular hasta la implantación de Java.

Java ha simplificado la programación en C++, añadiendo características fundamentales de C++ y eliminando alguna de las características que hacen a C++ un lenguaje difícil y complicado.

Java es simple porque consta solo de tres tipos de datos primitivos: números, *booleans* y *arrays*. Todo en Java es una clase. Por ejemplo, las cadenas son objetos verdaderos y no arrays de caracteres. Otros conceptos que hacen la programación en C++ mas complicada son los punteros y la herencia múltiple. Java elimina los punteros y reemplaza la herencia múltiple de C++ con una estructura única denominada interfaz (interface).

Java utiliza asignación y recolección automática de basura (*garbage collection*), aunque C++ requiere al programador la asignación de memoria y recolección de basura.

Otra característica importante es que la elegante sintaxis de Java hace más fácil la escritura de programas.

2.2.5.2 Orientado a objetos

La programación orientada a objetos modela el mundo real, cualquier cosa del mundo puede ser modelada como un objeto. Así, una circunferencia es un objeto, un automóvil

es un objeto, una ventana es un objeto, un libro es un objeto e incluso un préstamo o una tarjeta de crédito son objetos. Un programa Java se denomina *orientado a objetos* debido a que la programación en java se centra en la creación, manipulación y construcción de objetos.

2.2.5.3 Distribuido

La computación distribuida implica que varias computadoras trabajan juntas en la red. Java ha sido diseñado para facilitar la construcción de aplicaciones distribuidas mediante una colección de clases para uso en aplicaciones en red. La capacidad de red esta incorporada en Java. La escritura de programas en red es similar a enviar y recibir datos a y desde un archivo. La utilización de una URL (*Uniform Resource Locator*) de Java puede hacer que una aplicación acceda fácilmente a un servidor remoto.

2.2.5.4 Interpretado

Java es interpretado y se necesita un intérprete para ejecutar programas Java. Los programas se compilan en una Maquina Virtual Java generándose un código intermedio denominado bytecode. El bytecode es independiente de la maquina y se puede ejecutar en cualquier máquina con un interprete Java.

2.2.5.5 Robusto

Robusto significa fiable. Ningún lenguaje puede asegurar fiabilidad completa. Java se ha escrito pensando en la verificación de posibles errores y por ello como un lenguaje fuertemente tipificado. Java ha eliminado ciertos tipos de construcciones de programación presentes en otros lenguajes que son propensas a errores. No soporta, por ejemplo, punteros (apuntadores) y tiene una característica de manejo de excepciones en tiempo de ejecución para proporcionar robustez en la ejecución.

2.2.5.6 Seguro

Java, como lenguaje de programación para Internet, se utiliza en un entorno distribuido y en red. Se puede descargar un applet Java y ejecutarlo en su computadora sin que se produzcan daños en su sistema, ya que Java implementa diversos mecanismos de seguridad para proteger su sistema de daños provocados por un programa malicioso.

2.2.5.7 Arquitectura neutral

Una de las características más notables de Java es que es de arquitectura neutral, lo que también se define como independiente de la plataforma. Se puede escribir un programa que se ejecute en cualquier plataforma con una maquina virtual Java.

Se pueden ejecutar applets de java en un navegador Web; pero Java es algo mas que escribir applets, ya que se pueden también ejecutar aplicaciones Java autónomas directamente en sistemas operativos que utilicen un interprete Java.

2.2.5.8 Portable

Java es un lenguaje de alto nivel que permite escribir tanto programas convencionales como aplicaciones para Internet. Dado que Internet es una red formada por equipos muy diferentes interconectados por todo el mundo, resulta fundamental para los programas que rueden en ella su independencia de la plataforma en la que van a ser ejecutados. Gracias a la utilización del bytecode por parte de la maquina virtual en vez de instrucciones compiladas los programas java pueden ejecutarse en cualquier plataforma sin necesidad de ser recompilados, es decir, son muy portables. Pero la portabilidad de Java va aún más allá; Java fue diseñado de modo que pueda ser transferido a nuevas arquitecturas. El compilador Java está escrito en Java.

2.2.5.9 Alto rendimiento

Los compiladores de Java han ido mejorando sus prestaciones en las sucesivas versiones. Los compiladores conocidos como JIT (just-in.time) permiten que programas java independientes de la plataforma se ejecuten con casi el mismo rendimiento en tiempo de ejecución que los lenguajes convencionales compilados.

2.2.6 Base de Datos

Un sistema de base de datos es un sistema computarizado cuya finalidad general es almacenar información y permitir a los usuarios recuperar y actualizar esa información con base a peticiones. La información en cuestión puede ser cualquier cosa que sea de

importancia para el individuo u organización; en otras palabras, todo lo que sea necesario para ayudarle en el proceso general de su administración. [7]

2.2.7 Ventajas en el uso de bases de datos

La utilización de bases de datos como plataforma para el desarrollo de sistemas de aplicación en las organizaciones se ha incrementado notablemente en los últimos años, esto se debe a las ventajas que ofrecen su utilización, alguna de las cuales son [7]:

- Globalización de la información: permite a los diferentes usuarios considerar la información como un recurso corporativo que carece de dueños específicos.
- Eliminación de información inconsistente: si existen dos o más archivos con la misma información, los cambios que se hagan a estos deberían hacerse a todas las copias del archivo de factura.
- Permite mantener la integridad en la información: la integridad de la información es una de las cualidades altamente deseable y tiene por objetivo que sólo se almacene la información correcta.
- Independencia de datos: el concepto de independencia de datos es quizás el que más ha ayudado a la rápida proliferación del desarrollo de sistemas de base de datos. La independencia de datos implica un divorcio entre programas y datos.

2.2.8 Componentes de las bases de datos [7]

- Archivos de base de datos: estos contienen los elementos de los datos que se almacenan.
- Sistemas de administración de base de datos: llamado DBMS, es un conjunto de programas de software que administra la base de datos, controla el acceso a ella, le proporciona seguridad y realiza otras tareas.
- Sistema de interfaces de lenguaje anfitrión: esta es la parte del DBMS que se comunica con los programas de aplicación en lenguaje de alto nivel.
- Programas de aplicación: estos realizan las mismas funciones que en sistemas convencionales pero son independientes de los archivos de datos, y usan definiciones estándares de los mismos, los programas de aplicación usando el

lenguaje anfitrión de la interfaz, lo desarrollan por lo general programadores profesionales (no se definen los datos).

- Sistema de interfaces de lenguaje natural: este lenguaje de consultas permite la actualización y las consultas en línea de los usuarios que no son muy ilustrados acerca de los sistemas de cómputo (Lenguajes Query, como SQL).
- Diccionario de datos: depósito centralizado de información en forma computarizada acerca de los datos en una base de datos (el nombre de cada elemento en la base de datos y una descripción y definición de sus atributos). El diccionario incluye información acerca de la localización de estos datos en los archivos de una base de datos y muchos también contienen reglas de acceso y de seguridad y privacidad acerca de los mismos.
- Los terminales de acceso y actualización en línea: estas pueden encontrarse adyacentes en la computadora o a miles de kilómetros de distancia, pueden ser terminales inteligentes, no inteligentes o micro computadora.
- Sistema gestor de interfaces de salida: este proporciona información de trabajos de rutina, documentos o informes especiales.

2.2.9 Sistema de gestión de base de datos

Un sistema de gestión de base de datos (SGBD) es una colección de numerosas rutinas de software interrelacionadas, cada una de las cuales es responsable de alguna tarea. El SGBD es un paquete generalizado de software, que se ejecuta en un sistema computacional anfitrión, centralizando los accesos a los datos y actuando a manera de interfaz entre los datos físicos y el usuario. [7]

2.2.10 Funciones principales del SGBD [7]

- Crear y organizar la base de datos.
- Establecer y mantener las trayectorias de acceso a la base de datos, de tal manera que los datos en cualquier parte de la base de datos se puedan accesar rápidamente.
- Manejar los datos de acuerdo con las peticiones de los usuarios.
- Mantener la integridad y seguridad de los datos.

- Registrar el uso de la base de datos.

2.2.11 Arquitectura de los niveles de base de datos

Las bases de datos respetan la arquitectura de los tres niveles definida, para cualquier tipo de base de datos, por el grupo ANSI/SPARC. En esta arquitectura la base de datos se divide en los niveles externo, conceptual e interno. [7]

2.2.11.1 Nivel interno

Es el nivel mas bajo de abstracción, y define como se almacenan los datos en el soporte físico, así como los métodos de acceso.

2.2.11.2 Nivel conceptual

Es el nivel medio de abstracción. Se trata de la representación de los datos realizada por la organización, que recoge las vistas parciales de los requerimientos de los diferentes usuarios y las aplicaciones posibles.

2.2.11.3 Nivel externo

Es el nivel de mayor abstracción, a este nivel corresponde las diferentes vistas parciales que tienen de la base de datos los diferentes usuarios. En cierto modo, es la parte del modelo conceptual a la que tiene acceso.

2.2.12 Proceso Unificado de Desarrollo de Software

También llamado Rational Unified Process en inglés, habitualmente resumido como RUP, es una infraestructura flexible de desarrollo de software que proporciona prácticas recomendadas probadas y una arquitectura configurable. Es un Proceso Práctico. [8]

2.2.13 Principios del RUP [8]

Adaptar el proceso

El proceso deberá adaptarse a las características propias del proyecto u organización. El tamaño del mismo, así como su tipo o las regulaciones que lo condicionen, influirán en su diseño específico. También se deberá tener en cuenta el alcance del proyecto.

Balancear prioridades

Los requerimientos de los diversos participantes pueden ser diferentes, contradictorios o disputarse recursos limitados. Debe encontrarse un balance que satisfaga los deseos de todos. Debido a este balanceo se podrán corregir desacuerdos que surjan en el futuro.

Demostrar valor iterativamente

Los proyectos se entregan, aunque sea de un modo interno, en etapas iteradas. En cada iteración se analiza la opinión de los inversores, la estabilidad y calidad del producto, y se refina la dirección del proyecto así como también los riesgos involucrados.

Elevar el nivel de abstracción

Este principio dominante motiva el uso de conceptos reutilizables tales como patrón del software, lenguajes 4GL o marcos de referencia (frameworks) por nombrar algunos. Esto evita que los ingenieros de software vayan directamente de los requisitos a la codificación de software a la medida del cliente, sin saber con certeza qué codificar para satisfacer de la mejor manera los requerimientos y sin comenzar desde un principio pensando en la reutilización del código. Un alto nivel de abstracción también permite discusiones sobre diversos niveles y soluciones arquitectónicas. Éstas se pueden acompañar por las representaciones visuales de la arquitectura, por ejemplo con el lenguaje UML.

Enfocarse en la calidad

El control de calidad no debe realizarse al final de cada iteración, sino en todos los aspectos de la producción. El aseguramiento de la calidad forma parte del proceso de desarrollo y no de un grupo independiente.

2.2.14 Lenguaje Unificado de Modelado (UML)

Definición

El Lenguaje de Modelado Unificado contiene una notación robusta para el modelado y desarrollo de sistemas orientados a objeto. Proporciona la tecnología necesaria para apoyar la práctica de la ingeniería del software orientada a objetos.

Como resultado de la aplicación de UML se puede producir un arreglo de modelos y documentos de trabajo. Sin embargo, éstos los reducen los ingenieros de software para lograr que el desarrollo sea más ágil y reactivo ante el cambio.

Otros métodos de modelaje como OMT (Object Modeling Technique) o Booch sí definen procesos concretos. En UML los procesos de desarrollo son diferentes según los distintos dominios de trabajo; no puede ser el mismo el proceso para crear una aplicación en tiempo real, que el proceso de desarrollo de una aplicación orientada a gestión, por poner un ejemplo. [8]

2.2.15 Elementos de UML [8]

- **Actor**: es una entidad externa (de fuera del sistema) que interacciona con el sistema participando (y normalmente iniciando) en un caso de uso. Los actores pueden ser gente real (por ejemplo, usuarios del sistema), otros ordenadores o eventos externos.

  Los actores no representan a personas *físicas* o a sistemas, sino su *papel*. Esto significa que cuando una persona interacciones con el sistema de diferentes maneras, estará representado por varios actores (ver figura 2.1).

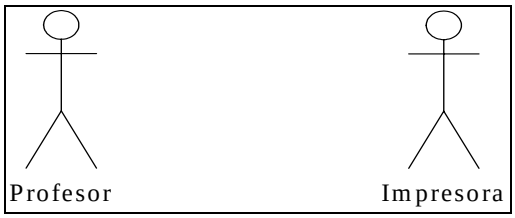

Figura 2.1: Representación de un Actor.

Fuente: Benevento, M. y Sánchez, C. (2008)

- **Clases**: En una clase se agrupan todos los objetos que comparten los mismos atributos, métodos y relaciones. Los atributos son características y propiedades comunes en todos los objetos de la clase. Los métodos son operaciones que deben cumplir las instancias de la clase. Las clases se representan como un rectángulo donde figuran el nombre de la clase, sus atributos y sus métodos (ver figura 2.2).

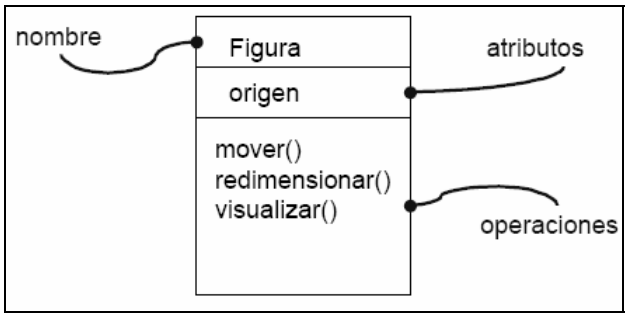

Figura 2.2: Representación de una clase

Fuente: Benevento, M. y Sánchez, C. (2008)

- **Casos de uso:** Un caso de uso es una descripción de un conjunto de secuencias de acciones que un sistema ejecuta y que produce un resultado observable de interés para un actor particular (ver figura 2.3).

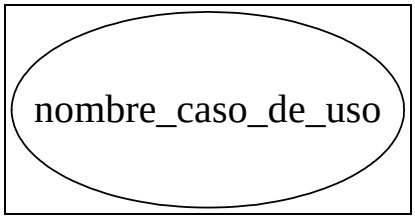

Figura 2.3: Representación de un caso de uso

Fuente: Benevento, M. y Sánchez, C. (2008)

- **Objetos**: Un objeto es una instancia de alguna clase (ver figura 2.4).

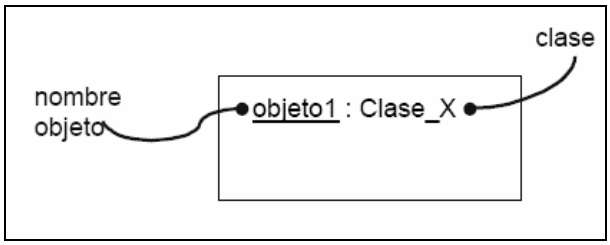

Figura 2.4: Representación de un objeto.

Fuente: Benevento, M. y Sánchez, C. (2008)

2.2.16 Diagramas de UML

Un diagrama es una representación gráfica de una colección de elementos del modelo, que habitualmente toma forma de grafo donde los arcos que conectan sus vértices son las relaciones entre los objetos y los vértices se corresponden con los elementos del modelo.

Los distintos puntos de vista de un sistema real que se quieren representar para obtener el modelo se dibuja de forma que se resaltan los detalles necesarios para entender el sistema**.** [8]

- **Diagramas de Casos de Uso**: Un diagrama de casos de uso es un diagrama que muestra un conjunto de casos de uso con sus relaciones y los actores implicados. Es un diagrama que sirve para modelar la vista estática de un programa. La vista estática nos permite visualizar el comportamiento externo del programa; de esta forma se consigue conocer qué es lo que debe hacer el programa independientemente de cómo lo haga y sabremos los elementos que interactúan con el sistema. Los elementos implicados en un diagrama de casos de uso son los casos de uso, las relaciones y los actores. Las relaciones y los casos de uso ya han sido explicados anteriormente y el papel del actor también ha sido comentado pero merece la pena detallarlo más: Un actor es un rol que interactúa con el sistema. Se define como rol porque un actor puede ser tanto un usuario de la aplicación como otro sistema o dispositivos externos. (ver figura 2.5)

Los diagramas de casos de uso se utilizan para ilustrar los requerimientos del sistema al mostrar cómo reacciona una respuesta a eventos que se producen en el mismo. En este tipo de diagrama intervienen algunos conceptos nuevos: un actor es una entidad externa al sistema que se modela y que puede interactuar con él; un ejemplo de actor podría ser un usuario o cualquier otro sistema. Las relaciones entre casos de uso y actores pueden ser las siguientes:

- **Un actor se comunica con un caso de uso.**
- **Un caso de uso extiende otro caso de uso.**
- **Un caso de uso usa otro caso de uso.**

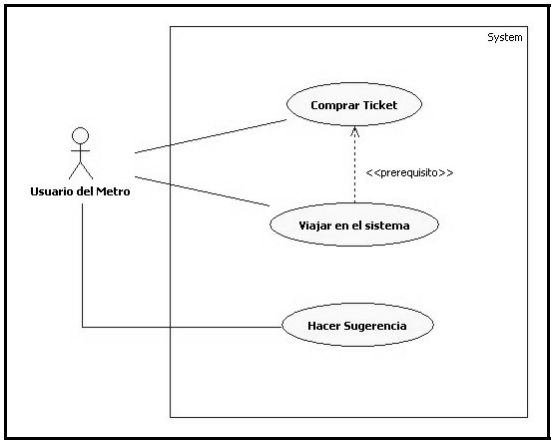

Figura 2.5: Representación de un diagrama de caso de usos

Fuente: Benevento, M. y Sánchez, C. (2008)

- **Diagramas de Secuencia**: Un diagrama de secuencia es un diagrama de interacción UML. Estos diagramas muestran la secuencia de mensajes que se van lanzando los objetos implicados en una determinada operación del programa. Dentro del diagrama los objetos se alinean en el eje X respetando su orden de aparición. En el eje Y se van mostrando los mensajes que se envían, también respetando su orden temporal. Cada objeto tiene una línea de vida donde se sitúa su foco de control. El foco de control es un rectángulo que representa el tiempo durante el que un objeto está activo ejecutando una acción. Con este sencillo esquema podemos visualizar la comunicación y sincronización bajo un estricto orden temporal de los objetos implicados en las distintas funcionalidades de un sistema.(ver figura 2.6)

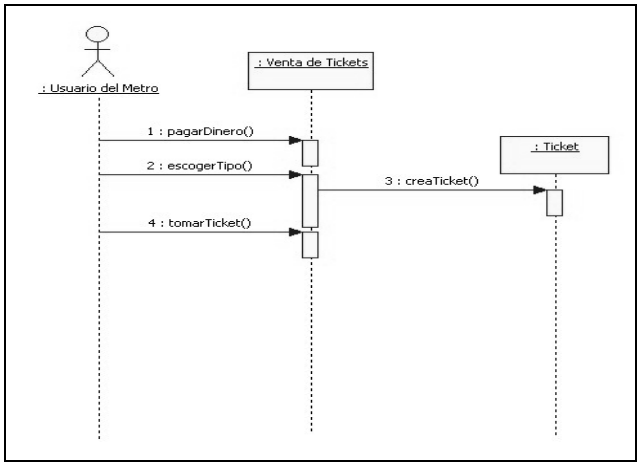

Figura 2.6: Representación de un diagrama de secuencia

Fuente: Benevento, M. y Sánchez, C. (2008)

- **Diagrama de Clases de Análisis:** Es utilizado por los desarrolladores de software para determinar los requerimientos funcionales, considerando una o varias clases, o sub-sistemas del sistema a desarrollar. Los casos de uso se describen mediante clases de análisis y sus objetos. El diagrama de clases de análisis se construye examinando los casos de usuarios, cerrando sus reacciones e identificando los roles de los clasificadores.

- **Diagrama de Clases de Diseño:** Se emplean para modelar la estructura estática de las clases en el sistema, sus tipos, sus contenidos y las relaciones que se establecen entre ellos. A través de este diagrama se definen las características de cada una de las clases, interfaces, colaboraciones y relaciones de dependencia y generalización (ver figura 2.7)**.**

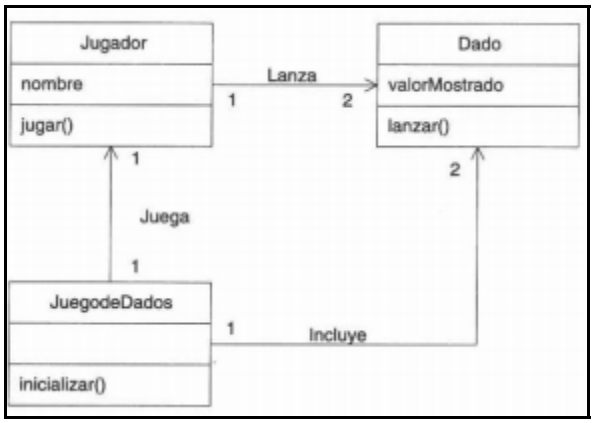

Figura 2.7: Representación del diagrama de clase de diseño

Fuente: Larman, C. (2001)

- **Diagramas de Actividad**: Son similares a los diagramas de flujo de otras metodologías Orientadas a Objetos (OO). En realidad se corresponden con un caso especial de los diagramas de estado donde los estados son estados de acción (estados con una acción interna y una o más transiciones que suceden al finalizar esta acción, o lo que es lo mismo, un paso en la ejecución de lo que será un procedimiento) y las transiciones vienen provocadas por la finalización de las acciones que tienen lugar en los estados de origen. Siempre van unidos a una clase o a la implementación de un caso de uso o de un método (que tiene el mismo significado que en cualquier otra metodología OO). Los diagramas de actividad se utilizan para mostrar el flujo de operaciones que se desencadenan en un procedimiento interno del sistema**.**

- **Diagramas de Implementación**: Se derivan de los diagramas de proceso y módulos de la metodología de Booch, aunque presentan algunas modificaciones. Los diagramas de implementación muestran los aspectos físicos del sistema. Incluyen la estructura del código fuente y la implementación, en tiempo de implementación. Existen dos tipos:

    - **Diagramas de componentes:** Muestra la dependencia entre los distintos componentes de software, incluyendo componentes de código fuente, binario y ejecutable. Un componente es un fragmento de código software (un fuente, binario o ejecutable) que se utiliza para mostrar dependencias en tiempo de compilación.

    - **Diagrama de plataformas o despliegue:** Muestra la configuración de los componentes hardware, los procesos, los elementos de procesamiento en tiempo de ejecución y los objetos que existen en tiempo de ejecución. En este tipo de diagramas intervienen nodos, asociaciones de comunicación, componentes dentro de los nodos y objetos que se encuentran a su vez dentro de los componentes. Un nodo es un objeto físico en tiempo de ejecución, es decir una máquina que se compone

habitualmente de, por lo menos, memoria y capacidad de procesamiento, a su vez puede estar formado por otros componentes.

- Modelo de Dominio*:* **es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción, en la tarea construcción del modelo de dominio, presentado como uno o más diagramas de clases y que contiene, no conceptos propios de un sistema de software sino de la propia realidad física.**

Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, ya sea de software o de otro tipo. Similares a los mapas mentales utilizados en el aprendizaje, el modelo de dominio es utilizado por el analista como un medio para comprender el sector industrial o de negocios al cual el sistema va a servir. (Ver figura 2.8)

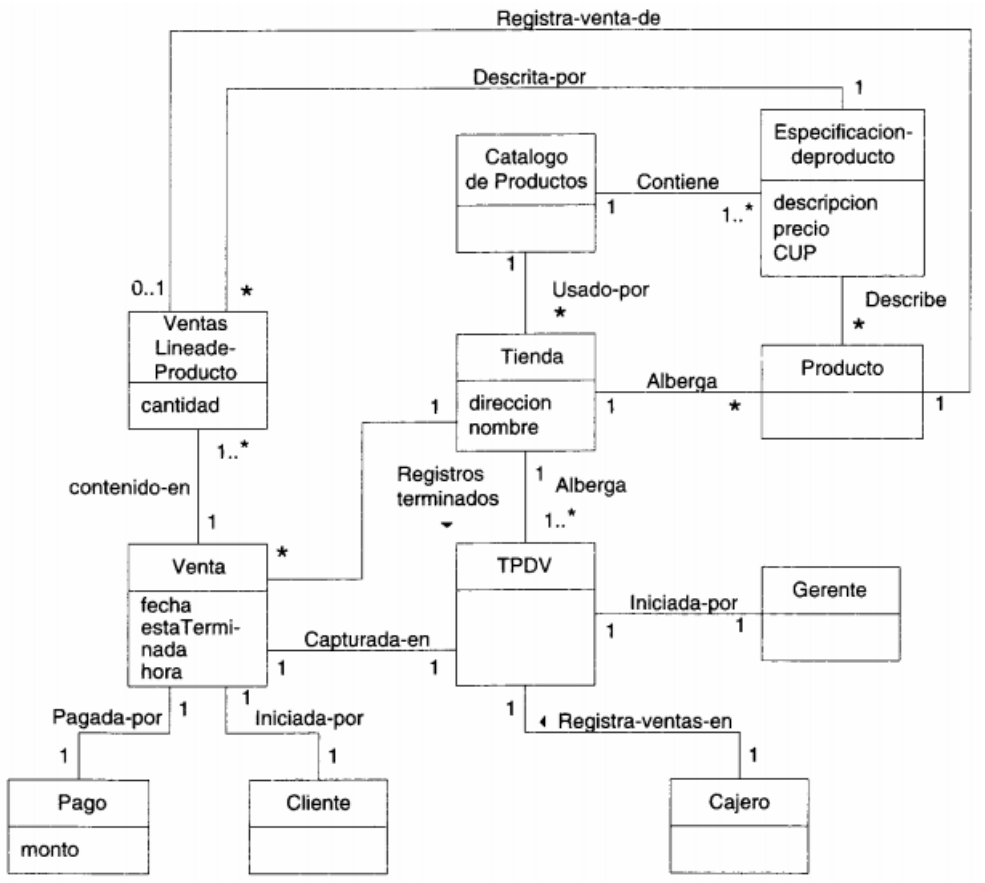

Figura 2.8: Representación de Modelo de dominio.

Fuente: Larman, C. (2001).

- **Diagrama de Paquetes**: Los diagramas de paquetes se usan para reflejar la organización de paquetes y sus elementos. Cuando se usan para representaciones, los diagramas de paquete de los elementos de clase se usan para proveer una visualización de los espacios de nombres. Los elementos contenidos en un paquete comparten el mismo espacio de nombre, el hecho de compartir espacios de nombres requiere que los elementos contenidos en un espacio de nombre específico tengan nombres únicos. Los paquetes se pueden construir para representar relaciones tanto físicas como lógicas (Ver figura 2.9).

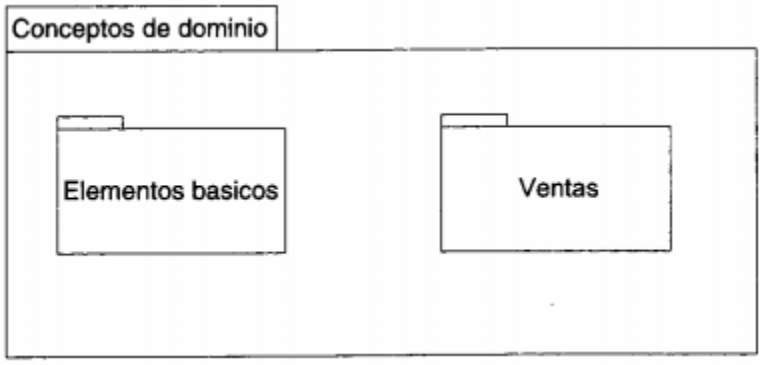

Figura 2.9: Representación de diagrama de paquetes
Fuente: Larman, C. (2001).

2.2.17 Java Media Framework

El framework de multimedia de Java (JMF), es una interfaz de programación de aplicaciones (API) utilizada para incorporar multimedia basada en tiempo a aplicaciones Java y Applets. El API JMF 1.0 (Conocido como API java reproductor de multimedia) permitía a los programadores desarrollar aplicaciones java con funciones de reproducción de multimedia basada en tiempo. El API JMF 2.0 extiende la funcionalidad del Framework para proveer soporte para la captura y almacenamiento de data multimedia, controlando el tipo de procesamiento que es realizado durante la reproducción, y realizando procesamientos personalizados a flujos de data multimedia. Adicionalmente, el API JMF 2.0 define un API de tipo “plug-in” que permite a los desarrolladores avanzados personalizar con mayor facilidad la funcionalidad del API JMF.

El Framework de multimedia de Java provee una arquitectura unificada y un protocolo de mensajería para la gestión de la adquisición, procesamiento y entrega de data multimedia basada en tiempo. JMF está diseñado para soportar la mayoria de los tipos de contenidos multimedia, tales como AIFF, AU, GSM, MIDI, MPEG, QuickTime y WAV.

Con JMF, es posible crear fácilmente applets y aplicaciones que muestren, capturen, manipulen y almacenen multimedia basado en tiempo. El Framework permite a los desarrolladores avanzados y suplidores de tecnología realizar procesamiento personalizado de data multimedia en bruto así como también extender el JMF para dar soporte a tipos de contenidos y formatos adicionales, optimizar el manejo de formatos soportados y crear nuevos mecanismos de presentación. [9]

2.2.18 Metas del diseño para el API JMF

El Framework de multimedia de java 2.0 soporta la captura de data multimedia y responde a la necesidad que presentan los desarrolladores de software que precisan mayor control sobre el procesamiento de la data multimedia. El API JMF 2.0 ha sido diseñado para [9]:

- Ser fácil de programar.
- Proveer soporte para la captura de data multimedia.
- Permitir el desarrollo de flujos de multimedia y aplicaciones de conferencias en Java.
- Permitir a los desarrolladores de software implementar soluciones a la medida basadas en el API existente así como también integrar fácilmente nuevas características al Framework ya establecido.
- Proveer acceso a data multimedia en estado bruto (sin procesar).
- Posibilitar el desarrollo de codecs, demultiplexores, procesadores de efectos y multiplexores descargables personalizados (plug-ins JMF).
- Mantener la compatibilidad con el API JMF 1.0.

2.2.19 Códigos de Barras

Los códigos de barras son las omnipresentes etiquetas con bandas negras y blancas que aparecen en casi todos los artículos al consumidor en la mayor parte del mundo moderno. Es una serie de barras negras, separadas por unos espacios blancos. Los anchos de las bandas, junto con sus propiedades reflectoras representan unos y ceros binarios que identifican las características del objeto. Además, pueden contener información acerca de la administración y control del inventario, acceso de seguridad, salidas y entradas, conteo de producción, procesamiento de documentos y pedidos, facturación automática y muchas otras aplicaciones. La figura muestra la distribución de los campos en un código característico de barras. El campo de inicio consiste en una secuencia exclusiva de barras y espacios, para identificar el inicio del campo de datos.

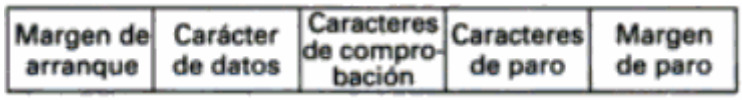

Figura 2.10: Distribución de los campos en un código de barras
Fuente: Tomasi, W. (2003)

Los caracteres de datos corresponden a la simbología o el formato del código de barras empleado. Los datos en serie, codificados en el campo de caracteres de datos se extraen de la tarjeta con un escáner o lector óptico. Este lector reproduce condiciones lógicas que corresponden a la diferencia de reflectividad de las barras impresas y los espacios en blanco. Para leer la información tan solo se recorre la barra impresa con un movimiento uniforme. Un fotodetector en el lector siente la luz reflejada y la convierte en señales eléctricas, unos y ceros, para su decodificación. [10]

Capítulo III: Fase de Inicio

La fase de inicio permite definir el ámbito del sistema para justificar la factibilidad del proyecto. Este capítulo plantea un modelo conceptual de su arquitectura de acuerdo a las especificaciones funcionales que se obtengan de la identificación de los requisitos exigidos. Se continúa con el análisis mediante el desarrollo del diagrama del modelo de actividades, aquí se refinan los casos de uso encontrados en el flujo precedente. Además se construirá el diagrama de paquetes de análisis, para encapsular los casos de uso que fueron definidos al realizar el análisis del sistema.

3.1 Contexto del sistema

Se entiende por contexto el entorno físico o de situación a partir del cual se considera un hecho. Esta constituido por un conjunto de circunstancias que ayudan a la comprensión de un mensaje. Haciendo uso de este concepto se determinó el contexto del sistema, iniciando con el análisis de los procedimientos que se emplean en el área de trabajo de la sala de lectura del departamento de computación y sistemas.

El análisis se realizó mediante entrevistas no estructuradas a los ayudantes de la sala de lectura, con el propósito de familiarizarse con las diversas actividades que se llevan a cabo e identificar las debilidades del sistema implantado actualmente, con la finalidad de realizar mejoras pertinentes según sea el caso.

Luego de establecer el contexto se constituyen los requerimientos que servirán de objetivos durante el desarrollo del sistema.

3.2 Lista de Requisitos o Características

La lista presentada a continuación es producto de la recopilación de información necesaria para el entendimiento del problema y los procesos que se desean automatizar y mejorar, con miras a generar un sistema más eficiente. Según lo expresado con anterioridad, el sistema debe cumplir con los siguientes lineamientos:

- El software debe permitir la inscripción de estudiantes y tesistas en la sala, adicionalmente debe proveer un mecanismo de almacenamiento persistente de esta data.
- El software debe facilitar la gestión de préstamos de libros y tesis.
- Debe tener la funcionalidad para agregar los datos de libros y tesis a una base de datos persistente.
- El software diseñado deberá tener la opción para generar un código de barras tanto para los ejemplares (libros y tesis) así como también para los estudiantes.
- El sistema debe contar con mecanismos para la generación de reportes detallados sobre las transacciones realizadas diariamente, así como también reportes al final de semestre.
- El software debe permitir hacer respaldos de la información almacenada en la base de datos, así como también restaurar la base de datos a partir de un respaldo hecho con anterioridad.
- El sistema debe contar con un manual de usuario claro que ilustre las diversas funcionalidades del software.
- El sistema debe contar con una interfaz gráfica clara e intuitiva que facilite el uso por parte de distintos usuarios.

3.3 Identificación de Riesgos

La identificación de riesgos tienen como principal objetivo definir la viabilidad del proyecto a futuro, ya que estos pueden causar pérdidas o daños, que pondrían en peligro el buen funcionamiento del sistema o en el peor de los casos que afectará la culminación del proyecto. En tal sentido, una vez identificados los posibles riesgos del sistema se priorizan y se define un plan de manejo para atenuarlos.

Una vez identificados pueden ser tratados de diversas maneras. Se cuenta fundamentalmente con cuatro elecciones: evitarlos, limitarlos, eliminarlos o controlarlos. Algunos riesgos pueden evitarse mediante un cambio en la planificación del proyecto o una modificación en los requisitos; otros deberían restringirse de modo que sólo afecten una parte del proyecto; otros mitigarse ejercitándolos y observándolos; aunque existen riesgos que no pueden mitigarse, sólamente pueden controlarse y observar si aparecen. Los principales riesgos se muestran en la tabla 3.1

Tabla 3.1: Lista de riesgos críticos

Riesgo	Contingencia
Errores en la manipulación de la data.	Verificar las secuencias de código encargadas de la gestión de la data.
Incapacidad de uso del sistema por parte de los usuarios finales.	Comprobar el correcto diseño de las interfaces gráficas y del manual de usuario.
Escogencia de herramientas inadecuadas para el desarrollo del sistema.	Indagar sobre el alcance de estas herramientas y sus posibles alternativas.

Fuente: Elaboración propia

3.4 Modelo de Dominio

El modelo de dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés, no de componentes software u objetos software con responsabilidades.

Su objetivo es contribuir a la comprensión del contexto del sistema y de los requerimientos que se desprenden de este contexto; es decir, contribuye a una comprensión del problema que el sistema resolverá en relación a su contexto.

En la figura 3.1 se muestra el modelo de dominio del sistema actual el cual refleja las entidades y sus relaciones dentro del contexto del mismo y que interactúa dentro del proceso a automatizar, el sistema tendrá el nombre de “SIGESLEC”.

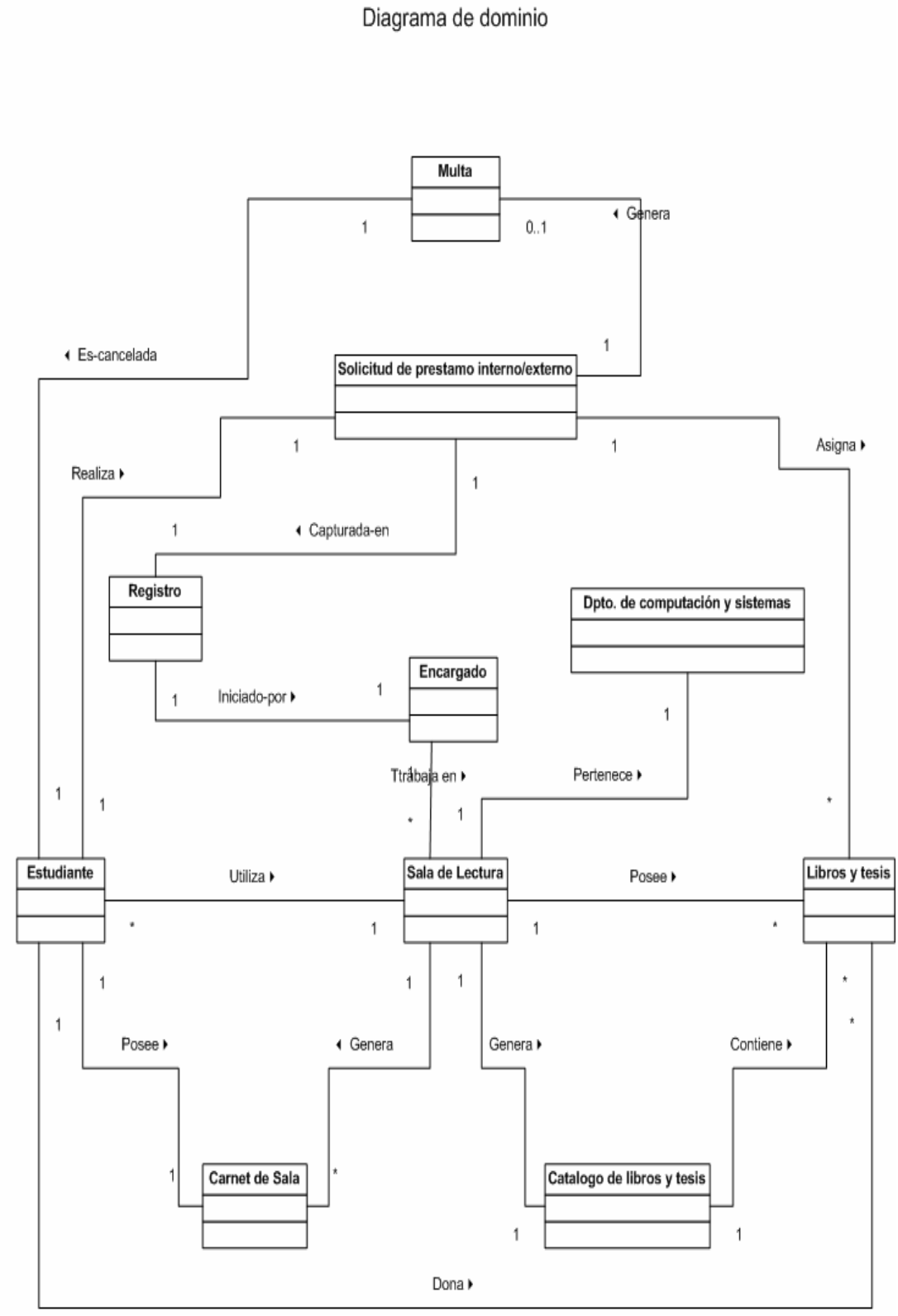

Figura 3.1 Modelo de Dominio del Sistema

Fuente: Elaboración propia

Las clases que se presentan en el modelo de dominio son las siguientes:

- **Dpto. de computación y sistemas:** Contiene aproximadamente 3000 estudiantes que son potenciales usuarios de la sala de lectura.

- **Sala de Lectura:** representa, como su nombre lo indica, a la sala de lectura adscrita al departamento de computación y sistemas de la Universidad de Oriente, núcleo Anzoátegui.

- **Estudiante:** cada uno de los usuarios de la sala que se inscriba y haga uso de la misma.

- **Libros y Tesis:** Diversos textos donados y trabajos de grados presentados con anterioridad que se encuentran en la sala.

- **Catalogo de libros y tesis:** contiene los nombre de los libros y tesis de grado ordenados por materia en el caso de los libros y cronológicamente en el caso de las tesis. para facilitar la búsqueda de ejemplares.

- **Carnet de sala:** identifica a los usuarios de la sala, y les permite solicitar los préstamos internos y circulantes de libros así como también de las tesis.

- **Encargado:** representa a un estudiante que labora como encargado de la sala en un periodo de tiempo definido en intervalos regulares. Se encarga de la gestión de préstamos y multas manualmente, así como también del orden de la sala.

- **Solicitud de préstamo interno/externo:** es realizada por un estudiante que desea obtener un libro de texto o una tesis de grado., y es registrada manualmente por el encargado de turno de la sala.

- **Multa:** es asignada a un estudiante que había solicitado un préstamo circulante y entrego el libro o tesis en un plazo mayor al establecido. El cálculo del costo de la multa, así como también la cantidad de tiempo de suspensión del carnet es calculado por el ayudante manualmente.

Vale la pena acotar que el procedimiento de préstamo y recepción no se pudo expresar apropiadamente en el diagrama, debido a que involucra una relación entre varios elementos.

El procedimiento de préstamo se inicia cuando un estudiante solicita el catálogo de tesis, en caso que desee una tesis de grado, o directamente solicita el libro por su nombre o autor. Seguidamente, el ayudante verifica visualmente si está en existencia el libro/tesis, en caso de que este disponible, procede a llenar la planilla de préstamo, anotando en ella los datos del estudiante y del ejemplar a prestar, seguidamente retiene el carnet del estudiante y hace entrega del ejemplar.

El proceso para recibir un ejemplar comienza cuando un estudiante regresa una tesis o libro a la sala, el cual es recibido por el ayudante que a continuación verifica manualmente si hubo un retraso en la entrega, en cuyo caso procede a calcular manualmente el costo de la multa y del tiempo de suspensión del carnet.

3.5 Modelo de Casos de Uso

El modelo de casos de uso se compone por varios objetos que intervienen en los procesos que un sistema es capaz de ejecutar, los cuales se describen mediante la identificación de casos de uso, identificación de actores y descripción de los casos de uso.

En la figura 3.2 se muestra el diagrama de casos de uso realizado para el sistema:

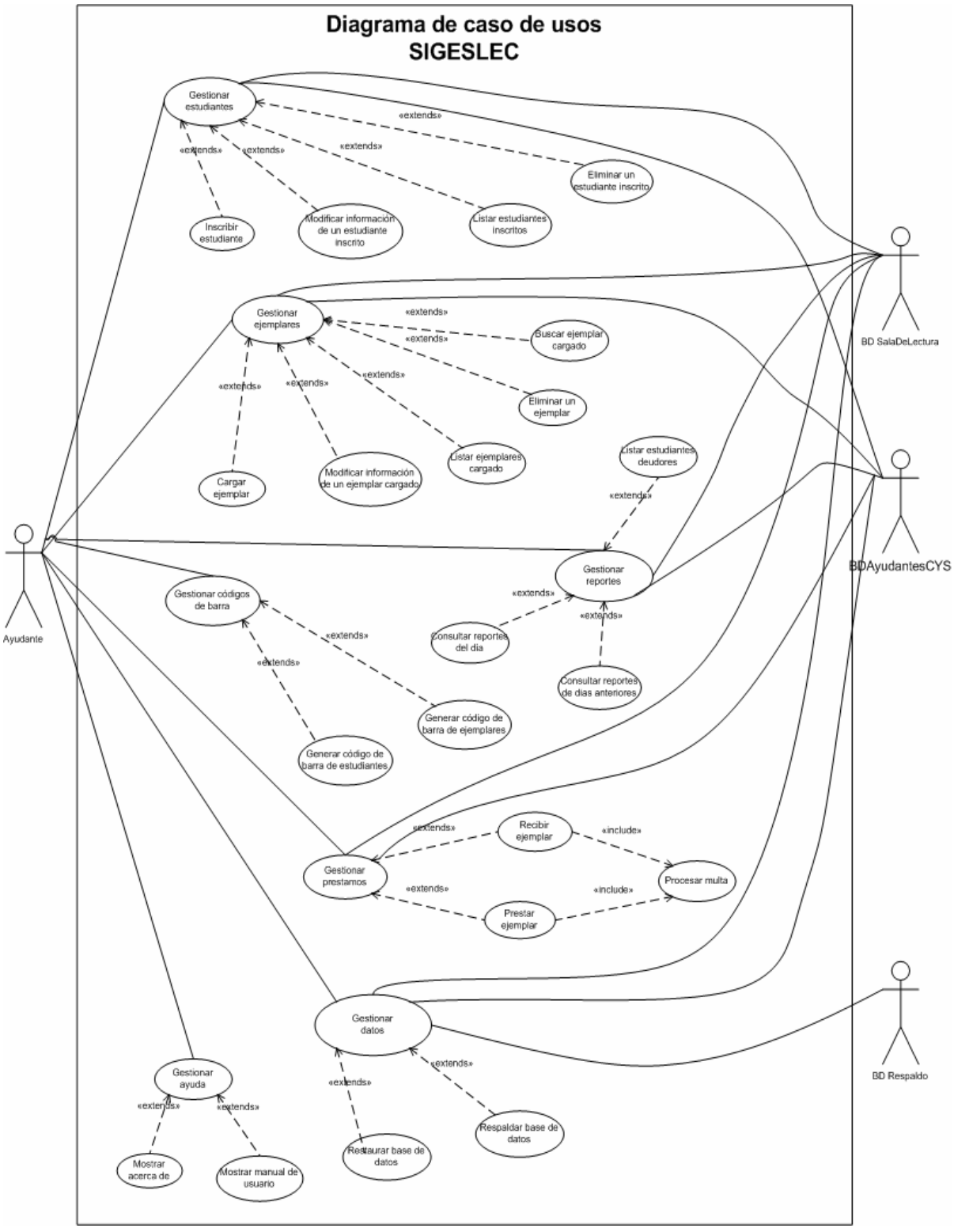

Figura 3.2: Diagrama de casos de uso para Sigeslec

Fuente: Elaboración propia

3.6 Identificación de Actores

Los actores son personas, sistemas o hardware externo que interactúa con el sistema en cuestión, es decir, representan terceros fuera del sistema que colaboran con éste. Los

actores pueden utilizar funciones propias del sistema, y de igual forma pueden proveer otras distintas al sistema, obteniendo o ingresando información en el mismo.

En la tabla 3.2 se muestran los principales actores que tendrían interacción con el software a desarrollar.

Tabla 3.2: Identificación de Actores

Actor	Descripción
Ayudante	Representa al usuario principal del sistema, el que interactuará directamente con el software. Es el encargado de utilizar el software para gestionar sus diversas funciones.
Base de datos Sala de Lectura	Se trata del conjunto de datos almacenados de manera persistente. Contiene la información de los estudiantes y de los ejemplares, préstamos, multas, así como también de los ayudantes.
Base de datos AyudantesCyS	Representa la base de datos de la aplicación externa que controla la asistencia de los ayudantes a la sala, con la cual el software interactúa para obtener la información de los ayudantes de turno.
Base de datos de Respaldo	Contiene un respaldo de la información de la base de datos utilizada por el software. Se realiza por medidas de seguridad para la preservación de la data.

Fuente: Elaboración propia.

3.7 Descripción de Casos de Uso

3.7.1 Gestionar Estudiantes

Este caso de uso se encarga de procesar todo lo relacionado con la información de inscripción de los estudiantes a la sala.

3.7.1.1 Inscribir estudiante

Aquí se procede a solicitar la información de cada estudiante a ser inscrito para almacenarla a la base de datos.

3.7.1.2 Modificar información de un estudiante inscrito

En este caso de uso se actualiza los datos de un estudiante que han sido almacenados con anterioridad.

3.7.1.3 Listar estudiantes inscritos

Aquí se genera un listado completo de todos los estudiantes presentes en la base de datos.

3.7.1.4 Eliminar un estudiante inscrito

Mediante este caso de uso se elimina a un estudiante del actor de almacenamiento.

3.7.2 Gestionar Ejemplares

En este caso de uso se engloban las distintas operaciones que se realizan con los diversos ejemplares (tanto libros como tesis) de la sala.

3.7.2.1 Cargar ejemplar

Aquí se introduce la información de un libro o tesis a la base de datos.

3.7.2.2 Modificar información de ejemplar cargado

En este caso de uso se actualizan los datos de una tesis o libro cargado con anterioridad.

3.7.2.3 Listar ejemplares cargados

Mediante este caso de uso se genera un listado de todos los ejemplares que se encuentran en el actor de almacenamiento persistente.

3.7.2.4 Eliminar ejemplar

Permite la eliminación de un ejemplar cargado con anterioridad.

3.7.2.5 Buscar ejemplar cargado

Este caso de uso permite ubicar un libro o tesis según varios criterios introducidos por el usuario.

3.7.3 Gestionar reportes

Este caso de uso permite la creación de reportes de las actividades diarias de préstamo.

3.7.3.1 Listar estudiantes deudores

Genera un listado de los estudiantes que habían solicitado un préstamo y realizaron la entrega en un tiempo mayor al establecido.

3.7.3.2 Consultar reporte del día

Permite realizar la revisión del reporte generado con las actividades más relevantes del día.

3.7.3.3 Consultar reportes de días anteriores

En este caso de uso se permite consultar reportes generados en días previos al actual.

3.7.4 Gestionar códigos de barras

Mediante este caso de uso se realizan las operaciones pertinentes al uso de los códigos de barras de estudiantes y ejemplares.

3.7.4.1 Generar código de barras de ejemplar

Crea una imagen del código de barras que representa a la cota del libro y/o tesis.

3.7.4.2 Generar código de barras de estudiante

Produce un archivo de imagen correspondiente al número de cédula de un estudiante inscrito en la sala.

3.7.5 Gestionar préstamos

Este caso de uso permite tramitar las operaciones relacionadas con los préstamos internos y circulantes tanto de libros como de trabajos de grado, así como también la recepción de los mismos.

3.7.5.1 Prestar ejemplar

Realiza el almacenamiento de los datos del préstamo, como son los datos del ayudante que realiza el prestamo, el estudiante que lo solicita y los datos del libro o tesis que se está prestando.

3.7.5.2 Recibir ejemplar

Mediante este caso de uso se almacena la recepción de un ejemplar por parte de un estudiante, registrándose en el actor de almacenamiento.

3.7.5.3 Procesar multa

Este caso de uso se encarga de la verificación del estatus de un estudiante, ya se si se encuentra solvente o si al entregar un libro o tesis lo hace con retraso, en cuyo caso procede a calcular el monto de la multa y el tiempo de suspensión del estudiante.

3.7.6 Gestionar datos

Mediante este caso de uso se realizan operaciones de respaldo y restauración de los datos guardados por el actor de almacenamiento.

3.7.6.1 Respaldar base de datos

Permite generar un archivo con la información archivada en la base de datos que posibilite su restauración y conservación.

3.7.6.2 Restaurar base de datos

Este caso de uso permite reestablecer la base de datos a un punto almacenado con anterioridad.

3.7.7 Gestionar ayuda

Con este caso de uso se ilustra el funcionamiento del sistema, suministrando información clara sobre sus funciones e utilización.

3.7.7.1 Mostrar manual de usuario

Despliega el manual de ayuda para los usuarios.

3.7.7.2 Mostrar acerca de

Proporciona información general sobre el software.

3.8 Modelo de Clase de Análisis

El modelo de clase de análisis tiene como finalidad definir la estructura básica de clases que conformarán el sistema. Está conformado por los actores involucrados, las interfaces, las entidades y los gestores. A continuación se muestran los diagramas para los casos de uso: Gestionar estudiantes, Gestionar ejemplares y Gestionar préstamos en las figuras 3.3, 3.4 y 3.5 respectivamente:

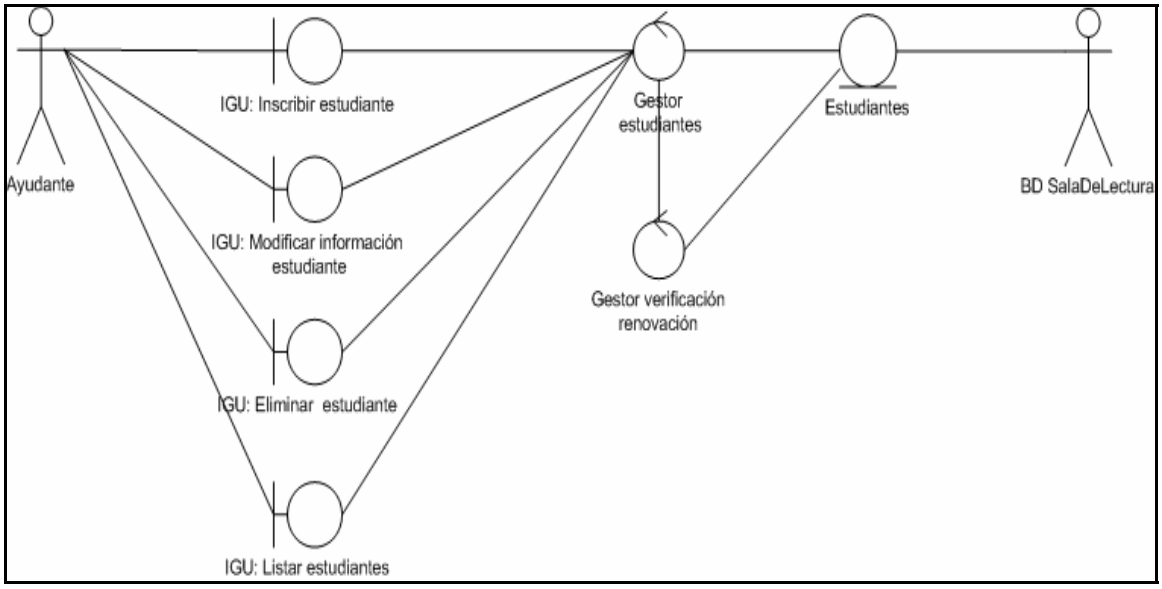

Figura 3.3: Diagrama de clase de análisis – Gestionar estudiantes.

Fuente: Elaboración propia.

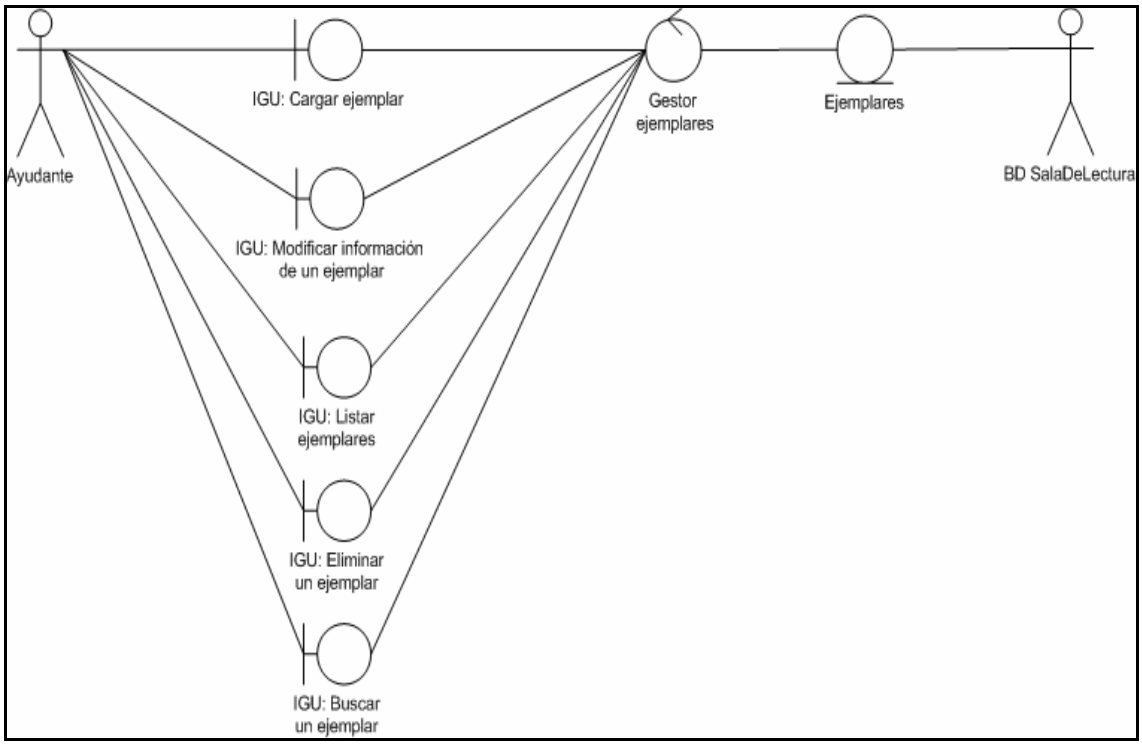

Figura 3.4: Diagrama de clase de análisis – Gestionar ejemplares.

Fuente: Elaboración propia.

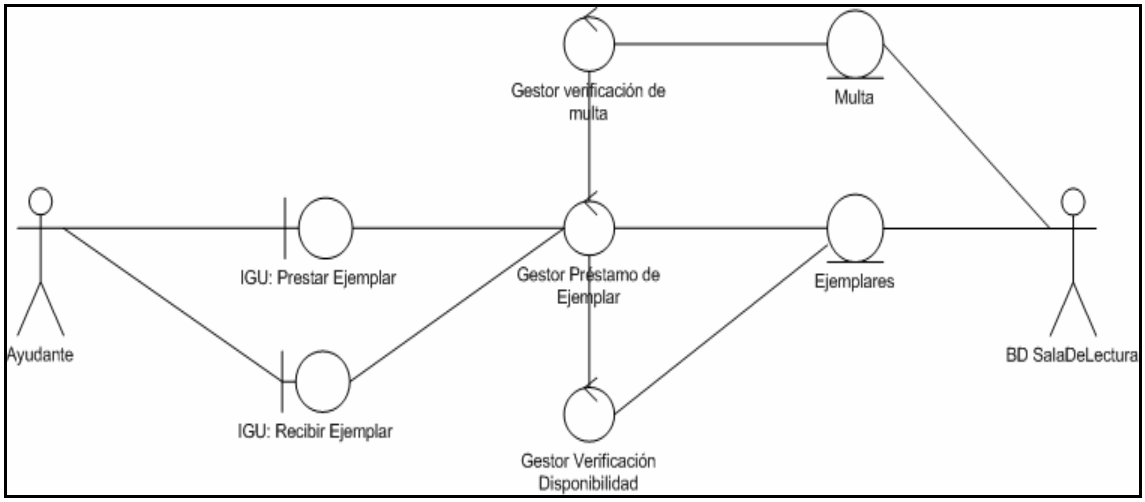

Figura 3.5: Diagrama de clase de análisis – Gestionar préstamos.

Fuente: Elaboración propia.

3.9 Diagrama de Colaboración

Un diagrama de colaboración posee la misma estructura que el diagrama de clase de análisis, solo que incluye los diversos mensajes que se intercambian entre los elementos que conforman el sistema. Seguidamente se muestran los diagramas de colaboración para los casos de uso Gestionar estudiantes, Gestionar ejemplares y Gestionar préstamos en las figuras 3.6, 3.7 y 3.8 respectivamente:

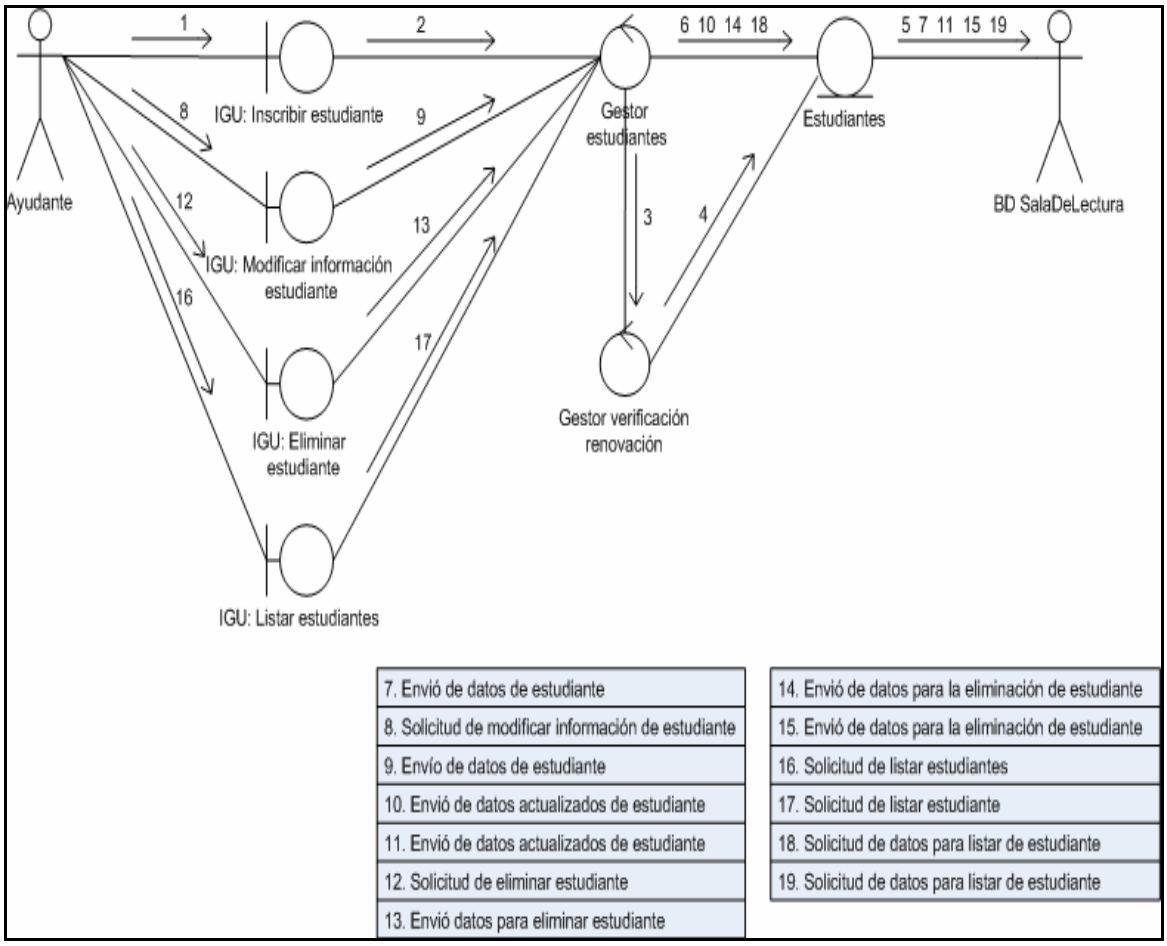

Figura 3.6: Diagrama de colaboración – Gestionar estudiantes

Fuente: Elaboración propia

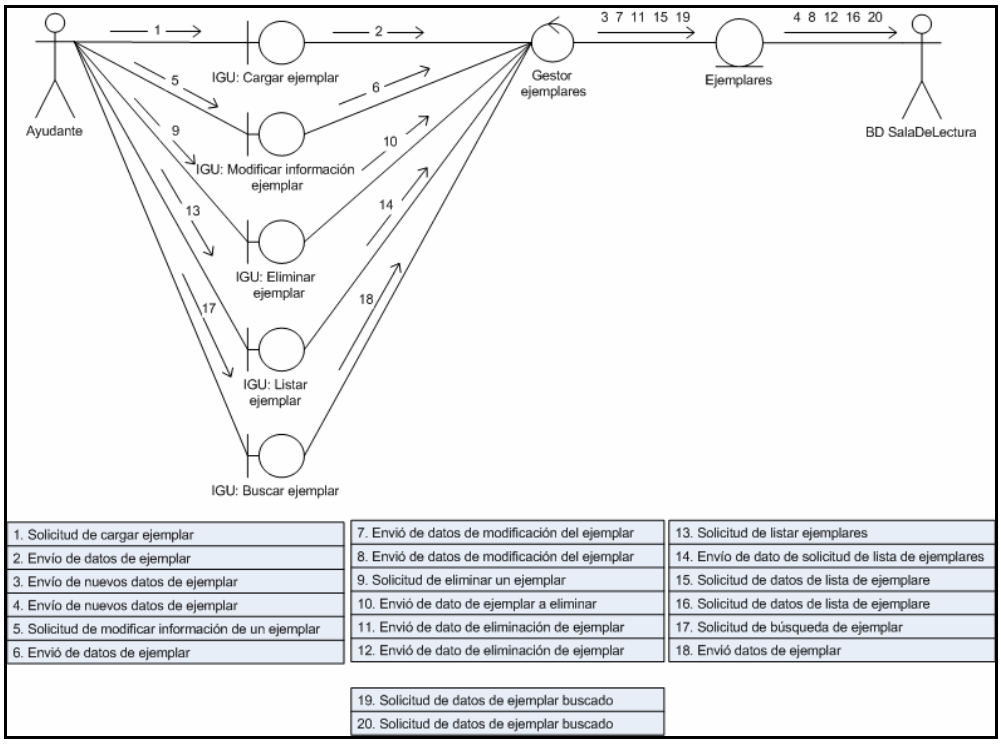

Figura 3.7: Diagrama de colaboración – Gestionar ejemplares

Fuente: Elaboración propia

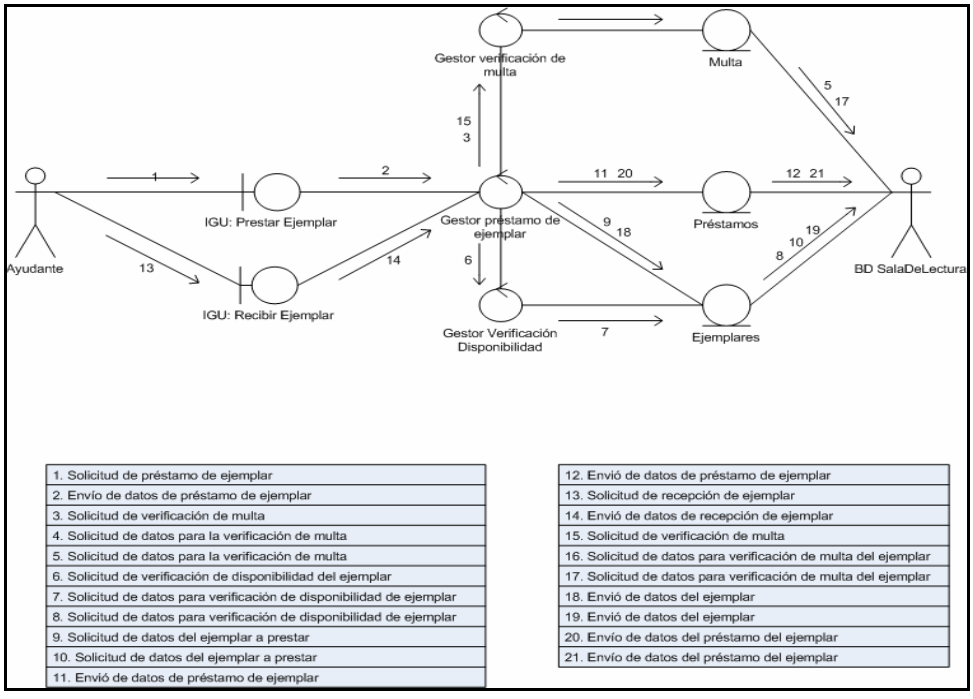

Figura 3.8: Diagrama de colaboración – Gestionar prestamos

Fuente: Elaboración propia

3.10 Diagrama de Paquetes de Análisis

Los paquetes de análisis proporcionan un medio para organizar los artefactos del modelo de análisis en piezas manejables. Un paquete de análisis puede constar de clases de análisis, de realizaciones de casos de uso, y de otros paquetes de análisis.

La identificación de paquetes de análisis permitirá organizar el modelo de análisis en unidades que, básicamente, englobarán los requerimientos funcionales del sistema definidos con anterioridad como casos de uso en el modelo de casos de uso.
A continuación se muestra el diagrama de paquetes del sistema en la figura 3.9:

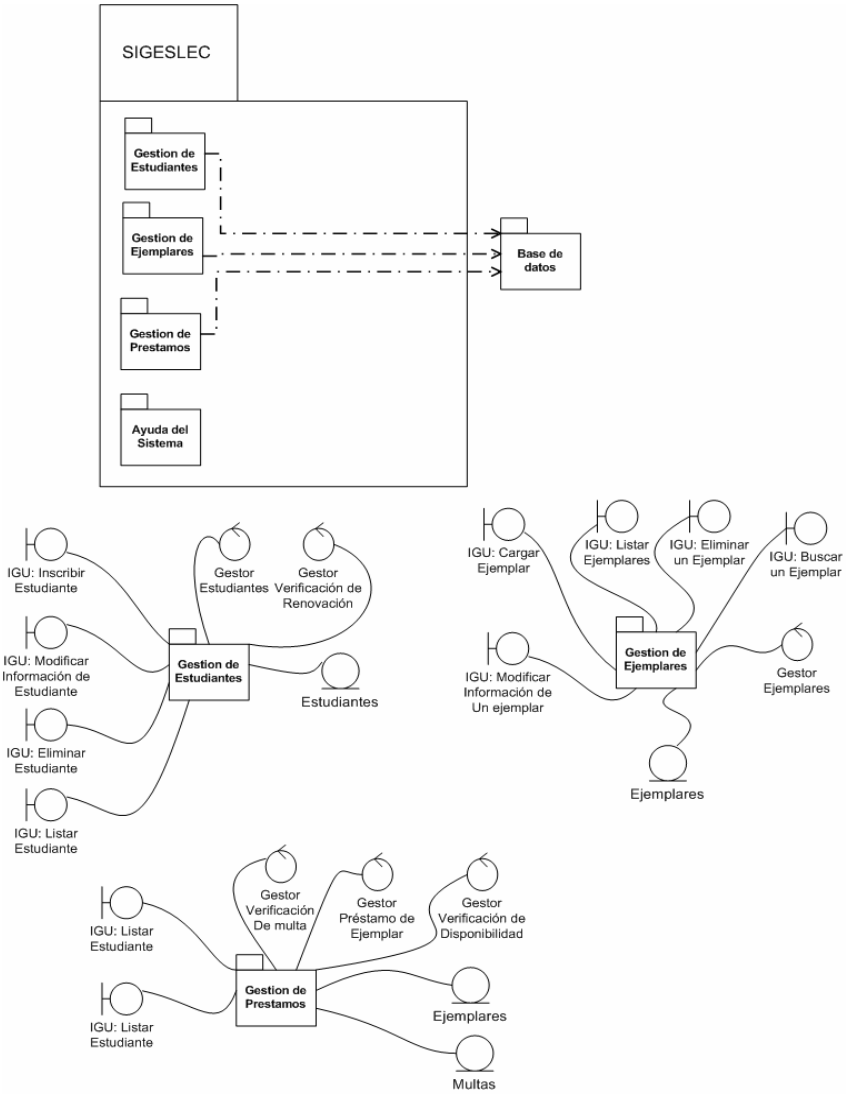

Figura 3.9: Diagrama de paquetes del sistema SIGESLEC

Fuente: Elaboración propia

3.11 Conclusión de la Fase de Início

En esta fase se realizó el análisis del procedimiento utilizado actualmente para el manejo de las diversas funciones ejercidas dentro del contexto del sistema, ejecutando un estudio de las necesidades de los usuarios, una lista de requisitos funcionales necesarios y un modelo de dominio. Seguidamente se formó una lista de riesgos críticos, los cuales se mitigaron a lo largo de esta fase y se continuarán atenuando en la fase de elaboración.

Como conclusión del análisis realizado a los procedimientos y normas de la sala de lectura se obtuvo que las diversas operaciónes realizadas en la sala tienen una alta propensión a fallas, debido a que todo es realizado de forma manual; bastaría con la perdida de un recibo que refleja una operación para que no exista constancia alguna de dicha operación.

Además, se generó el modelo de caso de usos para la captura de los requisitos funcionales, con la identificación de los actores que interactúan con el sistema, los casos de usos y las distintas relaciones existentes entre estos.

Despues de realizar la evaluación de los objetivos de la fase de inicio, contexto del sistema, riesgos críticos y arquitectura candidata se logró determinar la viabilidad del proyecto, por lo que se procede con el desarrollo del software.

Capítulo IV: Fase de Elaboración

4.1 Introducción

El propósito fundamental de la etapa de elaboración es crear la línea base del diseño para así disponer de una base sólida sobre la que se establecerá el diseño e implementación durante la fase de construcción. La arquitectura evoluciona considerando los requisitos más significativos y la evaluación de riesgos.

En esta fase serán modificados y depurados los modelos creados en la fase de inicio, generando así otro conjunto de modelos que irán perfilando una solución más cercana al mundo real, se construye la línea base de la arquitectura incluyendo técnicas de diseño globales. Se establecen los procesos, capas de software, paquetes, subsistemas, identificando sus responsabilidades y se efectúa la depuración de las interfaces internas y externas, refinándolas e incluyendo parámetros y valores de retorno.

4.2 Requisitos

En esta nueva iteración de la fase de elaboración no se identificaron nuevos actores, casos de uso, ni requisitos que se pudiesen añadir a los establecidos en la fase anterior, ya que hubo una completa comprensión del contexto en la fase de inicio, que permitió establecer los lineamientos que se van a utilizar durante el avance del proyecto.

4.3 Diseño

En esta fase se plantea un diseño a detalle, apuntando más a un modelo desarrollado de la arquitectura del sistema.

4.3.1 Diagrama de clase de diseño

El diagrama de clases es el principal diagrama de diseño para un sistema. Los diagramas de clases de diseño muestran la funcionalidad del sistema mediante clases relacionadas

entre sí por medio de líneas, haciendo la especificación de las instancias que participan en la relación.

Las clases de diseño definen los atributos y operaciones con que debe contar cada clase de análisis para llevar a cabo sus responsabilidades, así como las relaciones existentes entre ellas. Los atributos y operaciones se encuentran en su mayoría inmersos dentro de los diagramas y artefactos obtenidos en los diferentes flujos de trabajo, por lo que se necesita realizar un estudio exhaustivo de estos y agrupar las distintas responsabilidades de las clases para ofrecer soporte a todos sus roles. Los diagramas de clases muestran la estructura estática de los casos de uso.

La figura 4.1 representa de manera general el diagrama de clase para el sistema SIGESLEC.

4.3.2 Diagrama de clase de diseño para el caso de uso prestar ejemplar

A continuación se muestra en el diagrama 4.2 las diversas clases involucradas en la realización del caso de uso Prestar Ejemplar. Se puede apreciar la interacción entre las diversas clases, primeramente se muestra la GUIPrestarEjemplar, que es la interfaz gráfica para el caso de uso, las clases: GestorInfoEstudiante, GestorInfoEjemplar y GestorInfoPrestamo sirven como estructuras de datos para el almacenamiento de los diversos datos que se registran al momento de registrar un préstamo. La clase GestorPrestarEjemplar es la que se encarga de procesar los datos recibidos de la interfaz, así como también realizar el registro de la data, mediante la clase de conexión a la base de datos ConexionBD. La clase CellRendererPersonalizado se utiliza para dar características adicionales a los objetos gráficos que se muestran en la interfaz grafica. La clase GestorDeReportesHTML se encarga de la realización en tiempo real del reporte diario de sucesos del software.

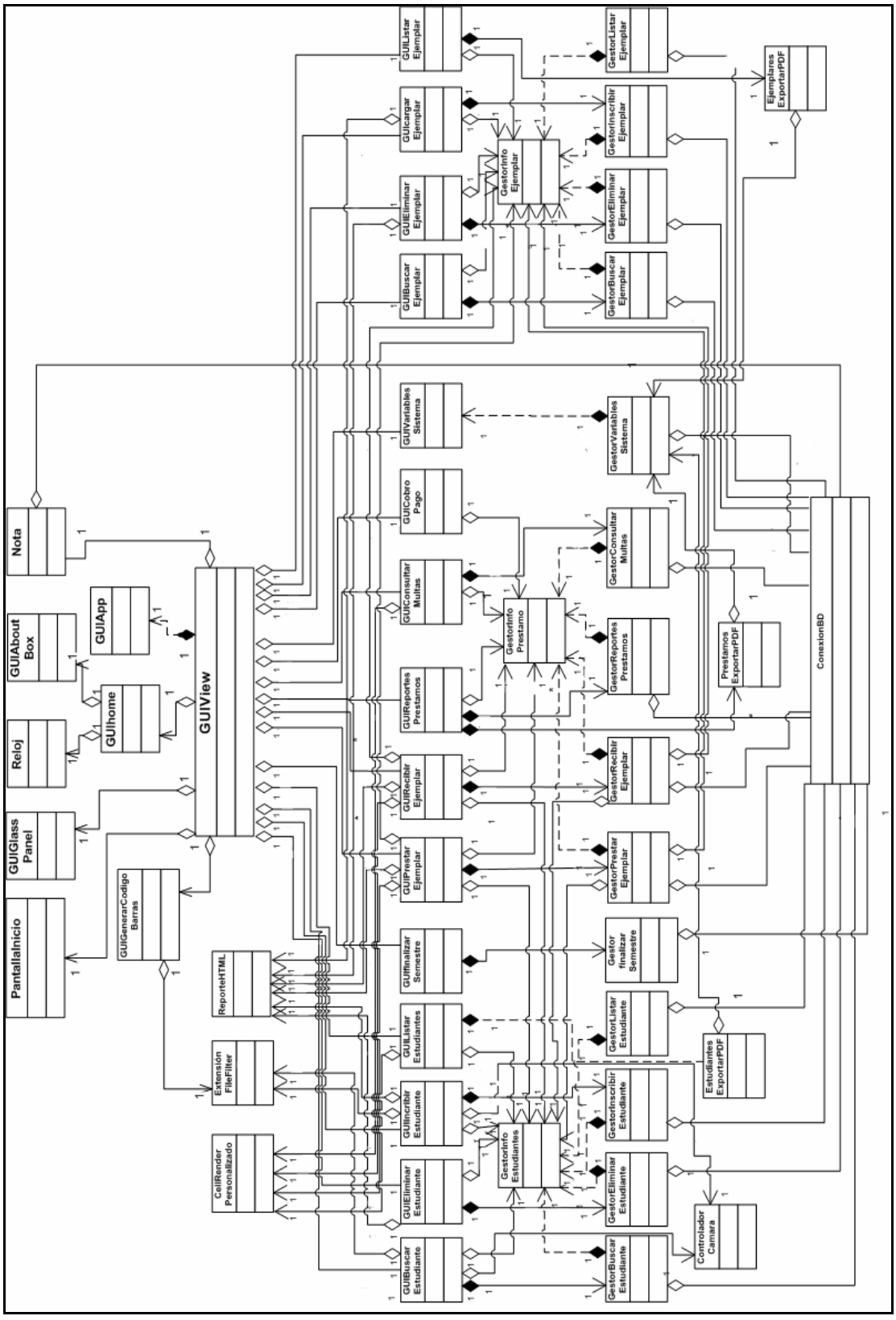

Figura 4.1: Diagrama de Clases de diseño del sistema SIGESLEC.

Fuente: Elaboración propia.

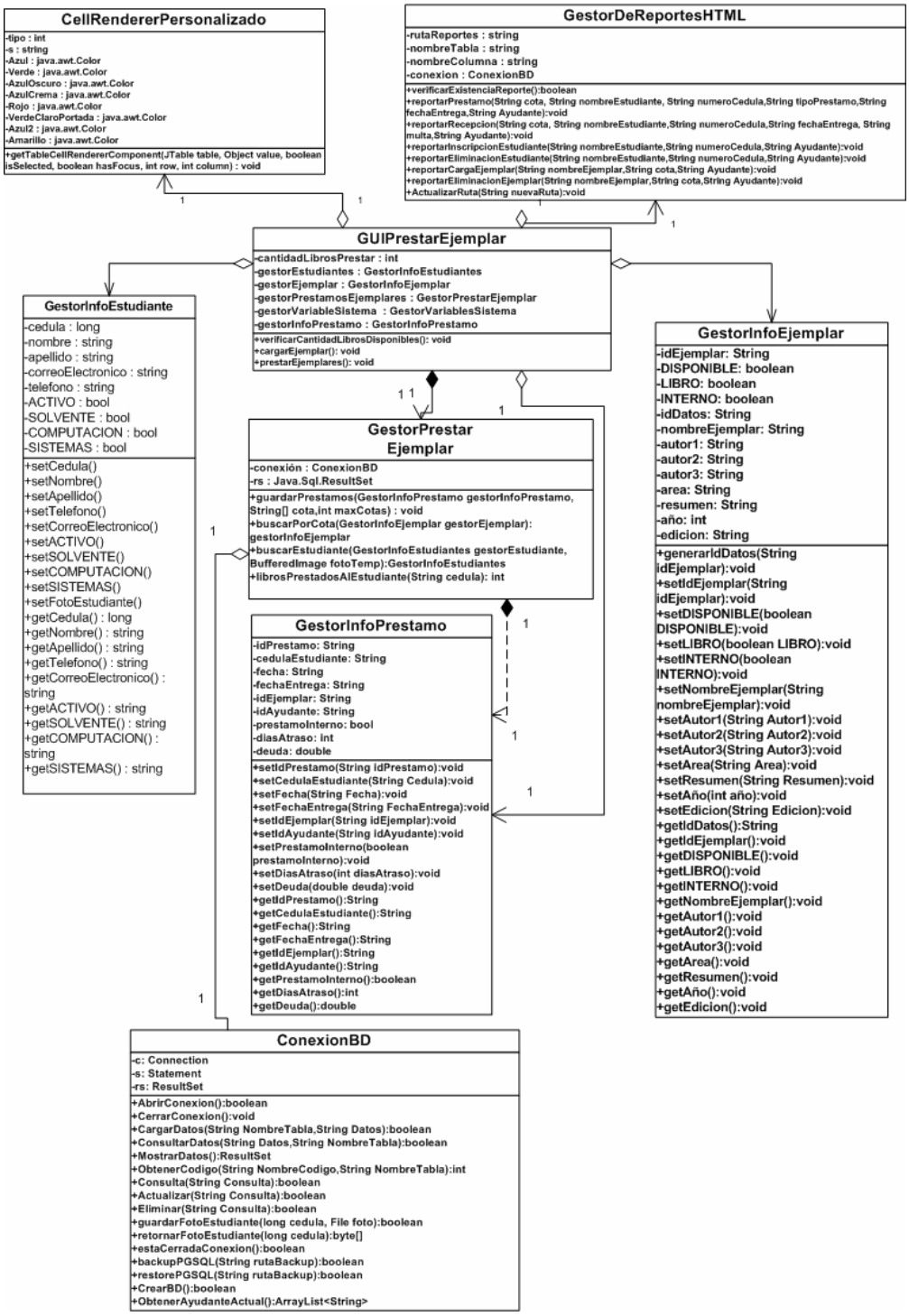

Figura 4.2: Diagrama de clase de diseño para el caso de uso prestar ejemplar.

Fuente: Elaboración propia.

4.3.3 Diagrama de secuencia

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada método de la clase; contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan

para implementar el escenario, y mensajes pasados entre los objetos. Modelan la secuencia lógica, a través del tiempo, de los mensajes entre las instancias.

A continuación se muestran los diagramas de secuencia de dos de los casos de usos más importantes del sistema SIGESLEC.

En la figura 4.3 se muestra el diagrama de secuencia para el caso de uso Inscribir Estudiante, en el se aprecia como es iniciado el caso de uso cuando el ayudante solicita la inscripción de un estudiante, seguidamente se activa la interfaz gráfica GUIinscribirEstudiante para la captura de los datos requeridos. Seguidamente, se realiza la verificación de la existencia del registro; esto se realiza ya que el proceso de inscripción en la sala de lectura debería realizarse más de una vez por un estudiante a lo largo de su carrera, su data no se elimina de la base de datos una vez finalizado el semestre. Así, si un estudiante está renovando su inscripción al inicio de un semestre simplemente se carga su información y se actualiza su estado a Activo. Una vez se recibe la data del registro (exista o no) se procede a solicitar la activación de la cámara Web a través de la clase ControladorCamara para capturar la foto del estudiante. Una vez capturada la foto, se envía la data de la interfaz gráfica al gestor de inscripción, GestorInscribirEstudiante, que realiza el procesamiento de la data y el registro físico de la misma a través de una instancia de la clase ConexionBD. El resto de los flujos corresponden a solicitudes para exportar la foto de un estudiante a formato .jpg, así como la restricción que el archivo a ser almacenado tenga esta extensión utilizando la clase ExtensionFileFilter.

En la figura 4.4 se muestra el diagrama de secuencia para el caso de uso Recibir Ejemplar. Se inicia cuando el ayudante solicita iniciar la recepción de un ejemplar, a continuación se activa la interfaz gráfica para el caso (GUIrecibirEjemplar), luego se envía la data recogida en la interfaz gráfica al gestor de información de ejemplar (GestorInfoEjemplar), de ahí se pasa la data al objeto gestor de recepción de interfaz (GestorRecibirEjemplar), el cual manipula la data del ejemplar para conseguir los datos del préstamo asociado al ejemplar, luego con esos datos calcula la multa (si es que la hay) y envía la información a la interfaz gráfica. De la interfaz se reciben los datos del pago de la multa y con estos datos se realiza la actualización de la información referente a la multa.

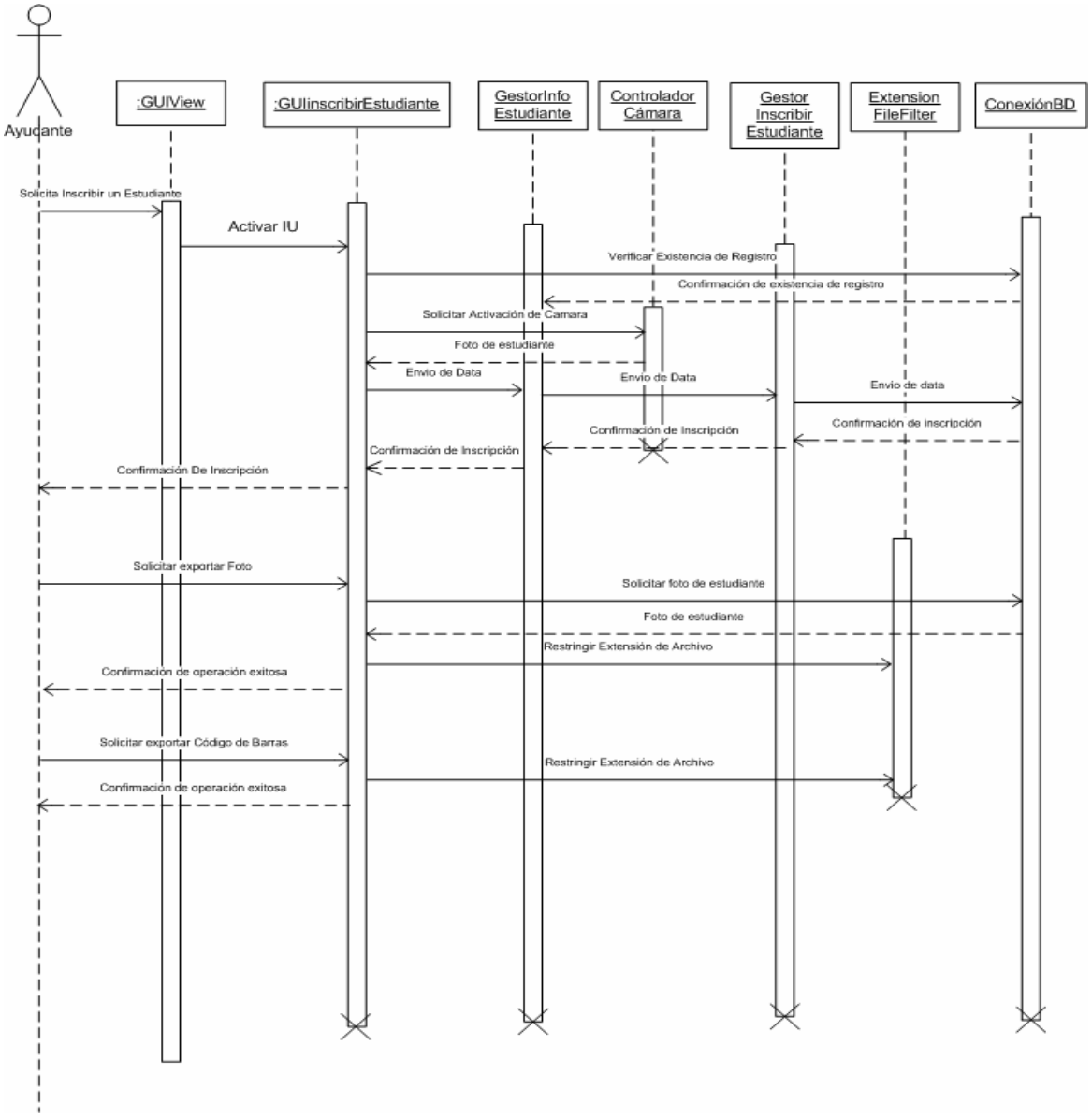

Figura 4.3: Diagrama de secuencia para caso de uso inscribir estudiante.

Fuente: elaboración propia.

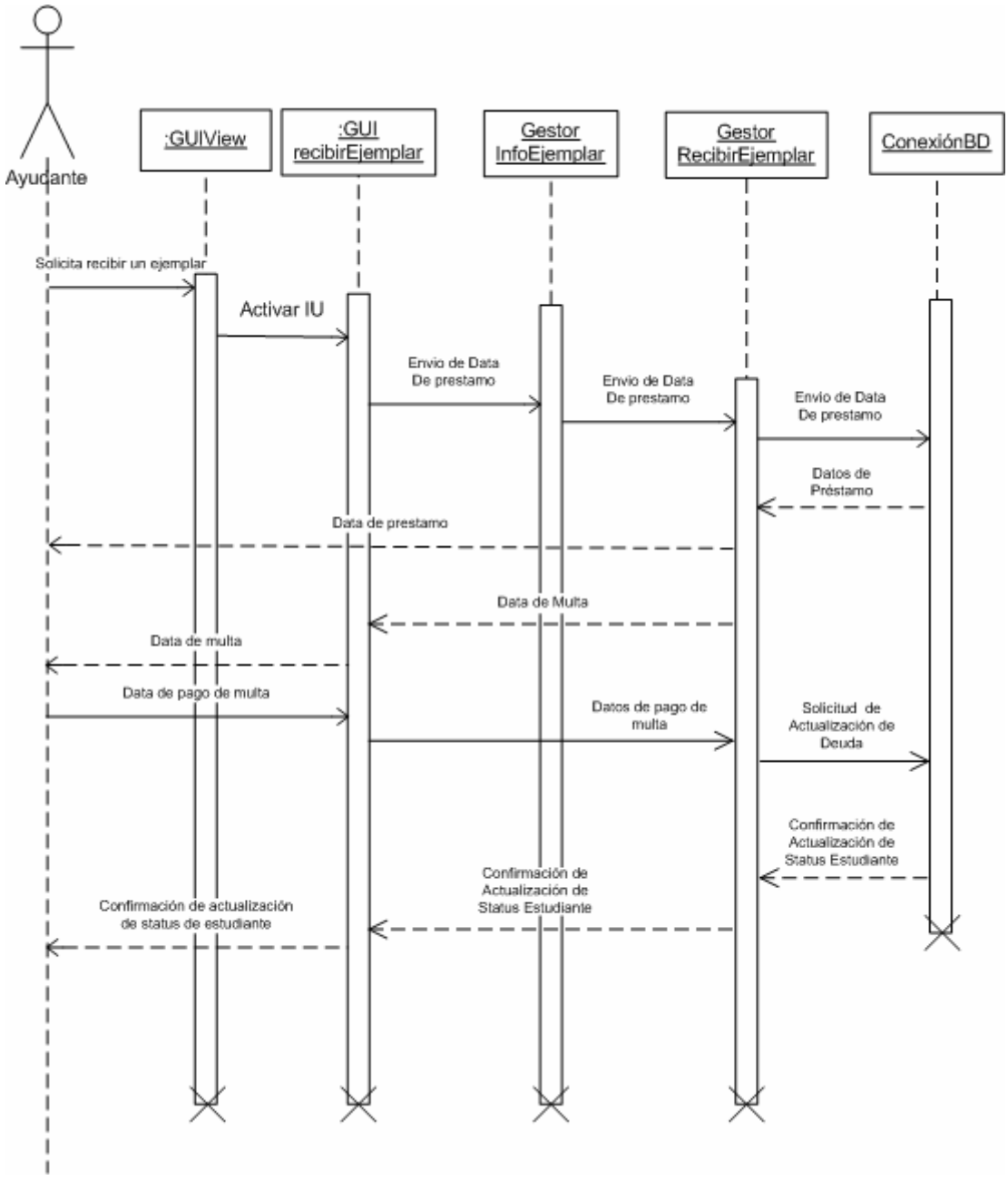

Figura 4.4: Diagrama de secuencia para el caso de uso recibir Ejemplar.

Fuente: Elaboración propia.

4.3.4 Diagrama de capas

El hardware y el software se complementan para definir el sistema informático. Las aplicaciones de software facilitan a los usuarios comunicar sus necesidades al computador sin tener que aprender a programar.

La Figura 4.5 muestra un diagrama de capas donde se describe el sistema informático en términos de varios niveles desde el mas externo (donde se encuentra el

usuario) hasta el mas interno (donde se encuentra el hardware). La información fluye entre las diferentes capas a nivel real y a nivel lógico.

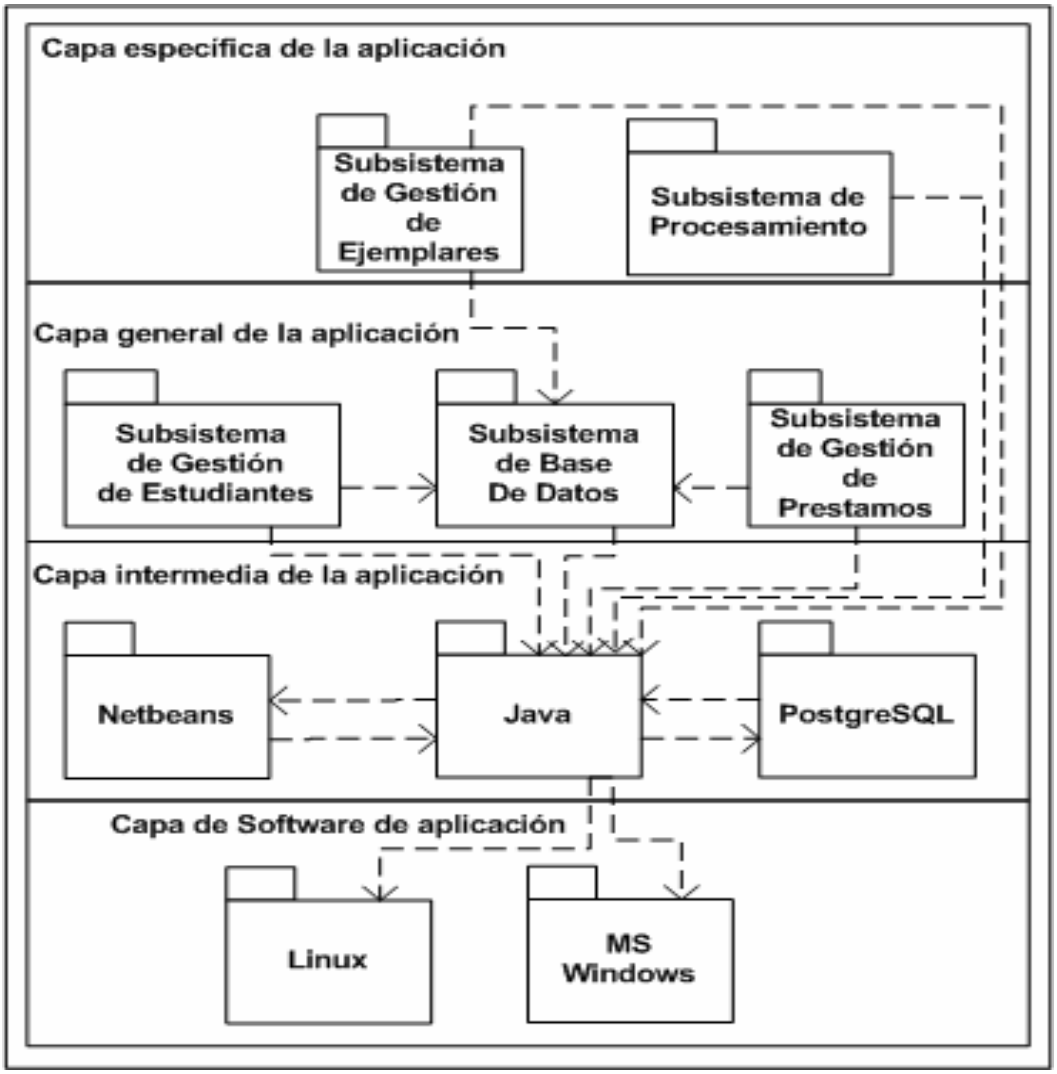

Figura 4.5: Diagrama de capas para el sistema SIGESLEC

Fuente: Elaboración propia.

4.3.5 Diagrama de Base de Datos

La base de datos relacional es el modelo de base de datos más difundido en la actualidad, en gran medida debido a que ofrecen sistemas simples y eficaces para representar y manipular los datos.

Se basan en el modelo relacional, con sólidas bases teóricas. Entre estas bases, se da la relación a nivel de filas (tuplas o registros) y columnas (atributos o campos) dentro de una misma tabla (entidad); cada “instancia” de la entidad encontrará sitio en

una tupla de la relación, mientras que los atributos de esta representaran las propiedades de la entidad. Adicionalmente, se da la relación entre entidades, cuando un atributo de una tabla hace referencia al atributo identificador de otra entidad.

Antes de realizar el diseño relacional de las tablas que conforman el sistema, se identificarán y explicarán cada una de ellas.

4.3.5.1 Identificación de las tablas

En esta etapa del diseño de la base de datos se han identificado un total de 8 tablas con las que contará el sistema. A continuación se muestran cada una de estas.

Estudiantes: esta tabla almacenará los datos referentes a cada estudiante que será inscrito en la sala de lectura. El campo *numerocedula* es la clave principal de esta tabla. Como se puede apreciar en la figura 4.6, se almacenan los datos básicos para cada estudiante, así como también una foto capturada desde el software. El campo booleano *activo* se utilizará para indicar si un estudiante ha renovado su inscripción para el presente semestre. Esto con la finalidad de almacenar la información de los estudiantes de manera tal que no se tenga que almacenar la misma información varias veces para un mismo estudiante. Este atributo se coloca en falso al final de cada semestre. El atributo *solvente* indica si un estudiante tiene alguna multa asociada a un préstamo entregado tardíamente.

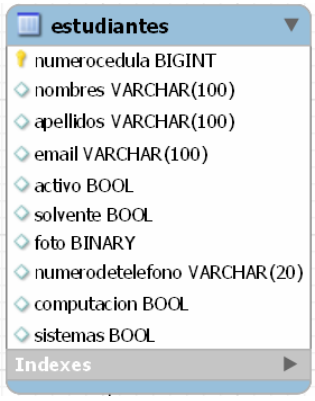

Figura 4.6: Tabla estudiantes

Fuente: Elaboración propia

Préstamo: en esta tabla se almacenarán lo campos clave de la información referente al préstamo de un ejemplar. De esta tabla se hace referencia a otras tablas para obtener la data que corresponde a un préstamo.

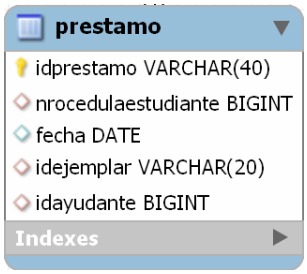

Figura 4.7: Tabla préstamo.

Fuente: Elaboración propia.

Ejemplar: esta tabla se utilizará para recopilar los datos de los ejemplares (tanto libros como tesis de grado) que se almacenan en la sala. Se identifica en ella si es libro ó tesis mediante el campo booleano *libro*. El campo *iddatos* se utiliza para acceder a los datos de un libro, que se almacenan en una tabla separada. Esto para minimizar la redundancia de información en los casos en que existan varios ejemplares del mismo libro.

Figura 4.8: Tabla Ejemplar.

Fuente: Elaboración propia.

Datosejemplares: esta tabla se utilizará para el almacenamiento de la data referente a un ejemplar. En ella se incluyen el área a la cual pertenece el ejemplar, así como un resumen del contenido del ejemplar. Esto con el propósito de realizar búsquedas desde el software con respecto a palabras claves del contenido en el ejemplar.

Figura 4.9: Tabla datosejemplares.

Fuente: Elaboración propia.

Multa: esta tabla almacenará la data que corresponde a una multa que se le genera a un estudiante cuando entrega un libro ó tesis fuera del tiempo establecido para el préstamo.

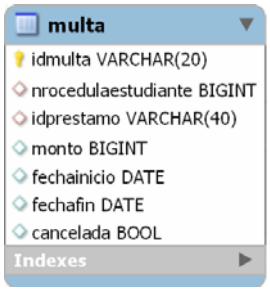

Figura 4.10: Tabla Multa.

Fuente: Elaboración propia.

Opcionessistema: esta tabla se utilizará para almacenar las características configurables del software, que pueden variar en el tiempo, tales como *cargo_por_un_dia*, *cargo_por_dia_adicional*, *duración préstamo*, *nroejemplaresporpersona*, etc. El atributo *prestamotesis* indica si es posible el préstamo circulante de trabajos de grado.

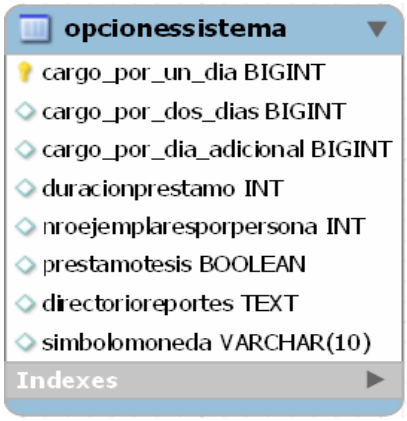

Figura 4.11: Tabla opcionessistema.

Fuente: Elaboración propia.

Nota: en esta tabla se almacenará un pequeño pedazo de texto que servirá a modo de comentario personal o recordatorio de algún evento por parte de un ayudante a otro.

El idnota es un campo entero auto incremental.

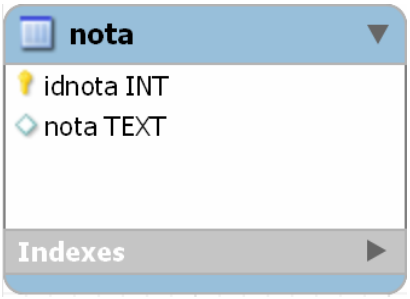

Figura 4.12: Tabla nota.

Fuente: Elaboración propia.

Ayudante: en esta tabla se recopilará los datos de los ayudantes de la sala de lectura. Dichos datos se obtendrán de la base de datos de un software externo, encargado del manejo de la asistencia de los ayudantes a las salas.

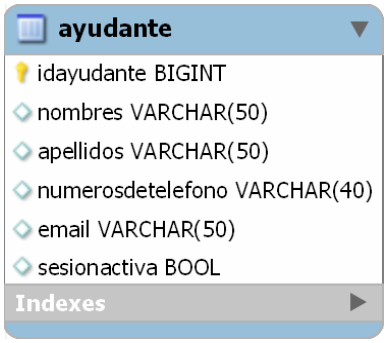

Figura 4.13: Tabla ayudante.

Fuente: Elaboración propia.

Como se mencionó anteriormente, la data de la asistencia de los ayudantes es manejada por un software externo, por lo tanto desde el sistema SIGESLEC se hace referencia a la base de datos de dicho software, que lleva por nombre AyudantesCYS. A continuación se muestran las tablas a las que se acceden.

Ayudante: esta tabla almacena el identificador único de cada ayudante, el tipo de ayudante y su contraseña.

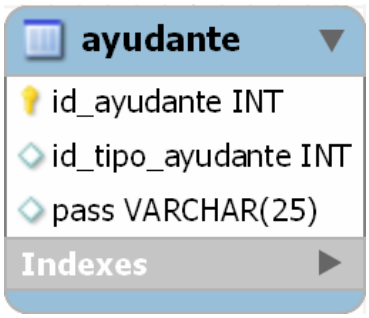

Figura 4.14: Tabla ayudante.

Fuente: Fermín, J., Ovando, N. [11]

Estudiante: esta tabla es utilizada para almacenar la data de los estudiantes que forman parte del conjunto de ayudantes de las diferentes salas (micro, navegación y lectura).

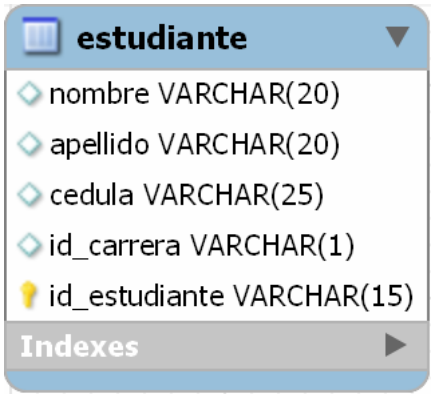

Figura 4.15: Tabla estudiante.

Fuente: Fermín, J., Ovando, N.

Horario_ayudante: esta tabla almacena el itinerario de los distintos estudiantes que cumplen la función de ayudante en las diferentes salas presentes en el departamento de Computación y Sistemas de la Universidad de Oriente Núcleo de Anzoátegui.

Figura 4.16: Tabla horario_ayudante.

Fuente: Fermín, J., Ovando, N.

4.3.5.2 Representación del diagrama relacional de base de datos

Luego de identificadas las tablas, se procedió a la elaboración del diagrama del modelo relacional. En la figura 4.17 se muestra el diagrama del modelo relacional de la base de datos.

4.3.6 Diseño de prototipos de interfaces gráficas de usuario

El objetivo del diseño de la interfaz es definir un conjunto de objetos y acciones de interfaz que posibiliten al usuario llevar a cabo todas las tareas de administración de contenido definidas, de forma que cumplan todos los objetivos determinados para el sistema.

Una vez elaborados los casos de usos y definidas las operaciones que realizan, es el momento indicado para la esquematización de los prototipos de interfaces.

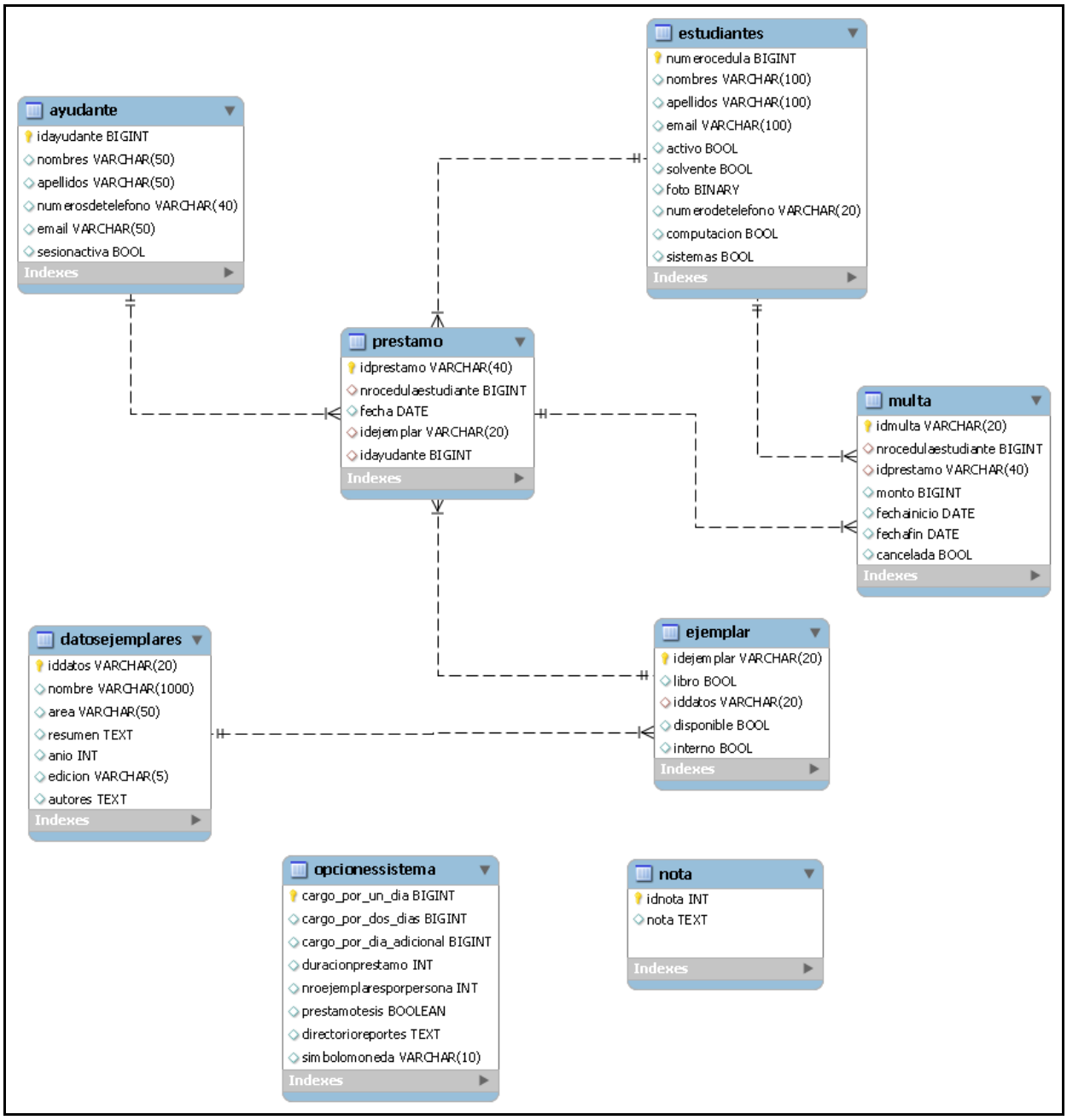

Figura 4.17: Diagrama de modelo relacional de la base de datos del sistema.

Fuente: Elaboración propia

4.3.6.1 Interfaz de inicio de SIGESLEC

Esta es la interfaz que verán los usuarios inmediatamente al ejecutar la aplicación. Proporciona un tiempo adicional de carga para establecer la configuración de inicio del sistema. Se muestra en la figura 4.18.

Figura 4.18: Interfaz de inicio del sistema SIGESLEC

Fuente: Elaboración propia.

4.3.6.2 Interfaz principal: Pantalla de Inicio

La pestaña de inicio le presenta al usuario un acceso directo a dos operaciones primordiales que realiza el software, así como también los gadgets de nota y un reloj que refleja la hora actual del sistema operativo. Adicionalmente muestra un banner lateral con el nombre del sistema y la barra de menú y de accesos (ver figura 4.19).

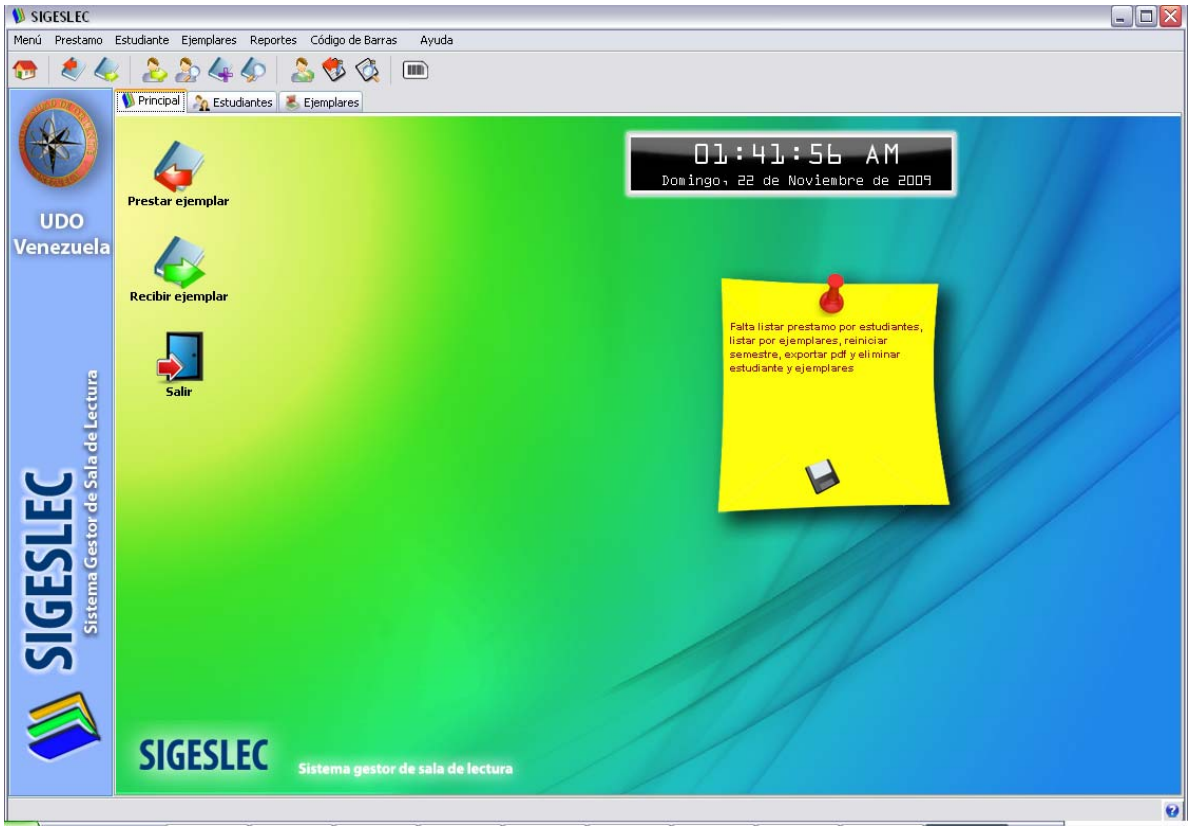

Figura 4.19: Interfaz Principal del sistema SIGESLEC

Fuente: Elaboración propia.

4.3.6.3 Interfaz Principal: Pantalla de Estudiantes

La segunda pestaña en la pantalla principal presenta un listado de todos los estudiantes inscritos en la sala y su estatus, ya sea solvente, no solvente y activo ó inactivo (ver figura 4.20).

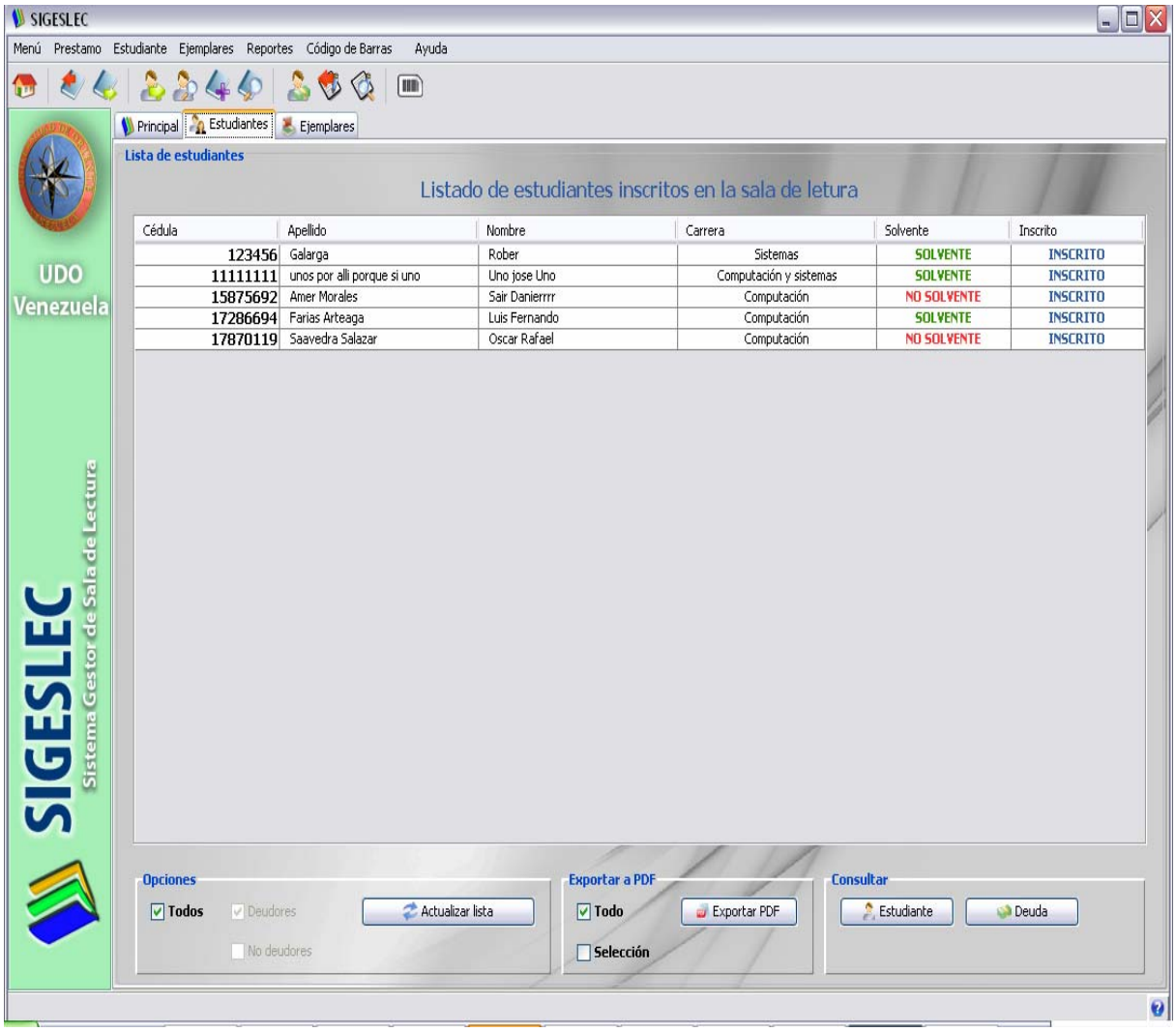

Figura 4.20: Interfaz Principal, Pestaña Estudiantes.

Fuente: Elaboración propia.

4.3.6.4 Interfaz principal (pantalla de ejemplares)

La tercera pestaña de la interfaz de inicio presenta un listado de todos los ejemplares cargados, muestra su disponibilidad y si esta disponible para préstamo interno o circulante (ver figura 4.21).

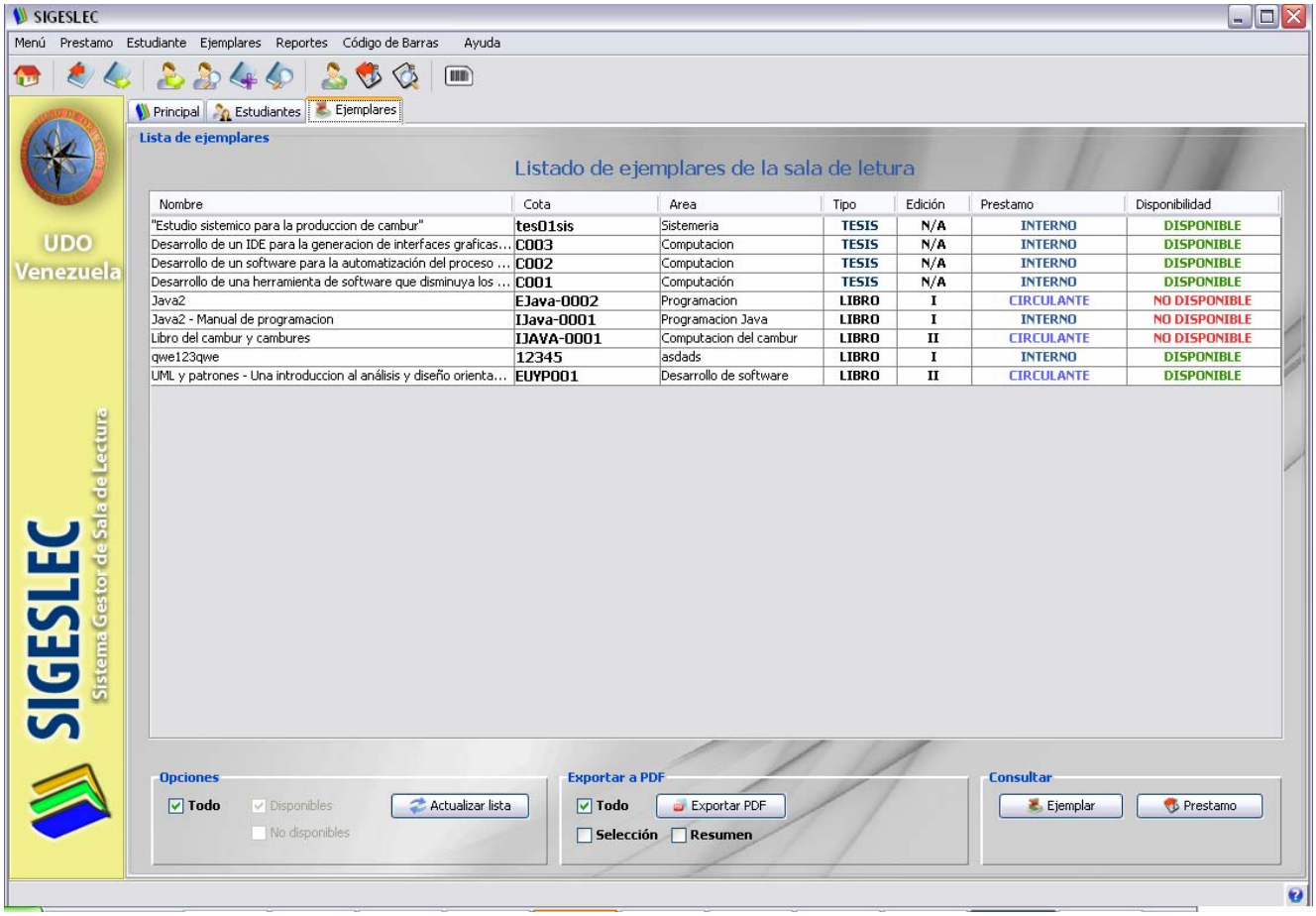

Figura 4.21: Interfaz Principal, Pestaña Ejemplares.

Fuente: Elaboración propia.

4.4 Implementación

En la implementación se partirá de los resultados del diseño y se implementará el sistema en términos de componentes. El propósito principal de esta iteración es desarrollar la arquitectura y el sistema con una visión global.

4.4.1 Diagrama de componentes

Para la fase de elaboración se ha estructurado en componentes al sistema, estableciendo las dependencias entre ellos. A continuación se representan las dependencias entre los componentes completos del sistema, incluyendo los detalles del caso de uso prestar ejemplar (ver figura 4.22).

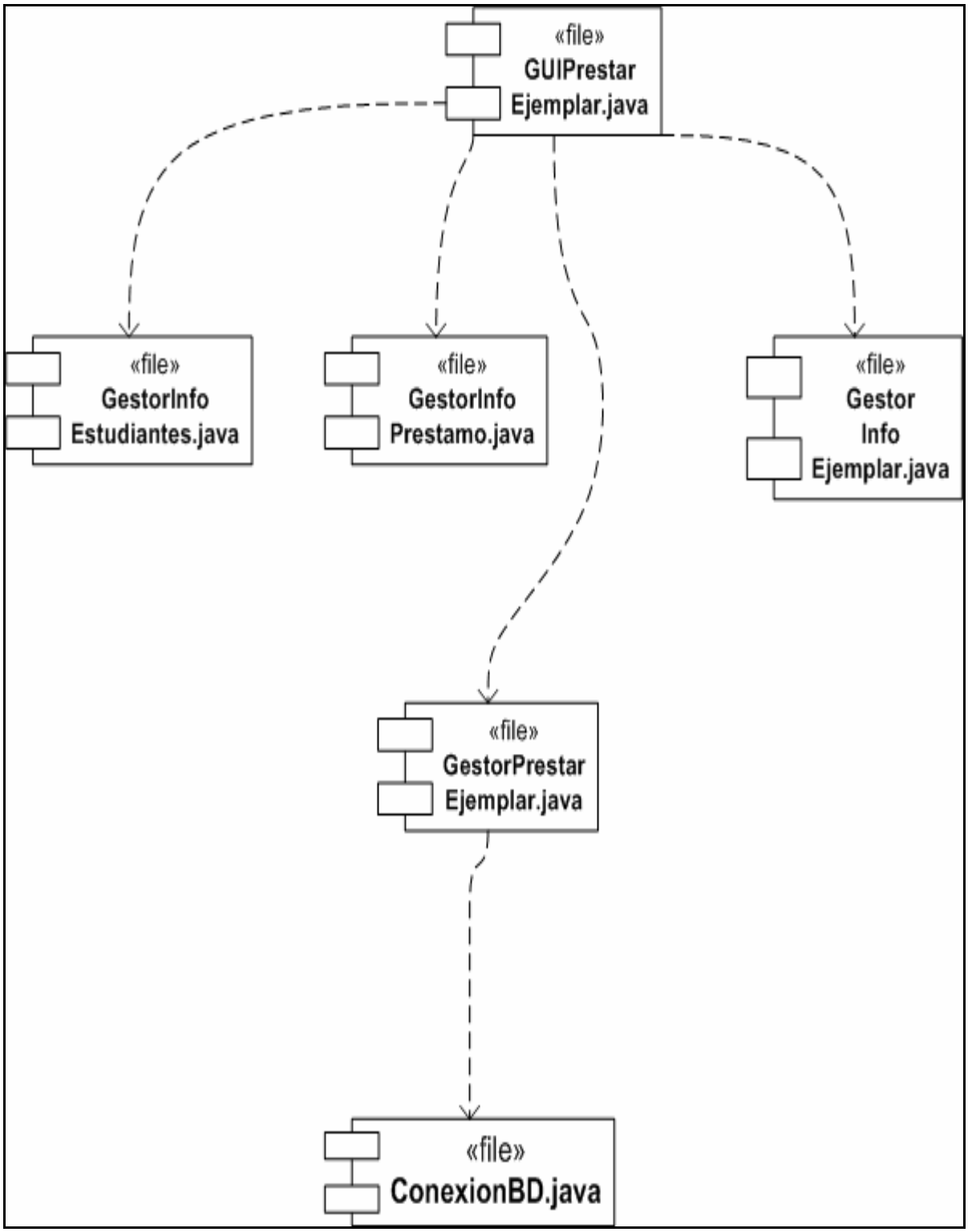

Figura 4.22: Diagrama de componentes, caso de uso Prestar Ejemplar

Fuente: Elaboración propia

4.4.2 Implementación de los componentes asociados al caso de uso prestar ejemplar

```
package GestionPrestamos;

import BaseDeDatos.ConexionBD;
import GestionEjemplares.GestorInfoEjemplar;
import GestionEstudiantes.GestorInfoEstudiantes;
import Principal.GUIView;
import java.io.IOException;
import java.sql.ResultSet;
import java.awt.image.BufferedImage;
import java.io.ByteArrayInputStream;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.util.regex.Pattern;
import javax.imageio.ImageIO;

// Autores: Oscar Saavedra y Sair Amer
//      Universidad de Oriente
//    Barcelona - Venezuela - 2009

public class GestorPrestarEjemplar {
    private GUIView pantallaPrincipal;
    private ConexionBD conexionBD;
    private ResultSet rs = null;

GestorPrestarEjemplar(GUIView pantallaPrincipal) {
        this.pantallaPrincipal = pantallaPrincipal;
    }


    // Guarda los datos de un préstamo en la base de datos
    // recibe un GestorInfoPrestamo con la información
    // el préstamo, un String[] con las cotas a cargar y un
    // int que determina la cantidad de libros que se van a cargar
    public void guardarPrestamo(GestorInfoPrestamo
gestorInfoPrestamo,String[] cota,int maxCotas){
```

```
// limpia los mensajes de la barra de estado de la pantalla
//principal
pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexión con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        int i;
        for (i=0;i<maxCotas;i++)
            conexionBD.Actualizar("UPDATE ejemplar SET
disponible='false' WHERE idejemplar='"+cota[i]+"'");

        for (i=0;i<maxCotas;i++){
            gestorInfoPrestamo.setIdEjemplar(cota[i]);

conexionBD.CargarDatos("prestamo(nrocedulaestudiante,fecha,idejemplar,
idayudante,interno,fechaentrega,recibido)",
                                    "
'"+gestorInfoPrestamo.getCedulaEstudiante()+"',"+
                                    "
'"+gestorInfoPrestamo.getFecha()+"',"+
                                    "
'"+gestorInfoPrestamo.getIdEjemplar()+"',"+
                                    "
'"+gestorInfoPrestamo.getIdAyudante()+"',"+
                                    "
'"+gestorInfoPrestamo.getPrestamoINTERNO()+"',"+
                                    "
'"+gestorInfoPrestamo.getFechaEntrega()+"',"+
                                    " 'false'");
        }
        conexionBD.CerrarConexion();
    }

// Método que busca un ejemplar en las tablas ejemplar y
//datosejemplares usando la cota del mismo
// @ SI consigue la cota retorna el objeto gestorEjemplar con la
//información del ejemplar  @ Si se produce un error o no se consigue
le ejemplar retorna null
    public GestorInfoEjemplar buscarPorCota(GestorInfoEjemplar
gestorEjemplar) {
        // limpia los mensajes de la barra de estado de la pantalla
```

```
//principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexión con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();

        conexionBD.Consulta("SELECT * FROM datosejemplares, ejemplar
WHERE datosejemplares.iddatos = ejemplar.iddatos AND
ejemplar.idejemplar = '"+gestorEjemplar.getIdEjemplar()+"'");
        rs = conexionBD.MostrarDatosConsulta();


try {
            if (rs.next()!=false){

gestorEjemplar.setNombreEjemplar(rs.getString("nombre"));
                gestorEjemplar.setArea(rs.getString("area"));
                gestorEjemplar.setLibro(rs.getBoolean("libro"));
                gestorEjemplar.setEdicion(rs.getString("edicion"));
                gestorEjemplar.setInterno(rs.getBoolean("interno"));

gestorEjemplar.setDisonible(rs.getBoolean("disponible"));
                gestorEjemplar.setAÃ±o(rs.getInt("anio"));
                gestorEjemplar.setResumen(rs.getString("resumen"));

                String autores = rs.getString("autores");
                String REGEX = "!!!";
                Pattern p = Pattern.compile(REGEX);
                String a[] = p.split(autores) ;
                gestorEjemplar.setAutores(a[0].trim(), a[1].trim(),
a[2].trim());
conexionBD.CerrarConexion();
                return gestorEjemplar;
            }
            else{
                conexionBD.CerrarConexion();
                pantallaPrincipal.advertenciaBarraEstado("La cota #
"+gestorEjemplar.getIdEjemplar()+" no se consigue en el sistema.");
                return null;
            }
        } catch ( SQLException ex) {
```

```
            conexionBD.CerrarConexion();
            pantallaPrincipal.errorBarraEstado("Error al buscar en la
base de datos.");

      Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level
      .SEVERE, null, ex);
                  return null;
            }
        }



        // Consulta la cedula en la base de datos, si la cedula
      //existe devuelve toda la información del estudiante a través
      //del objeto gestorEstudiante  @ Si hay un error durante la
      //lectura el objeto gestorEstudiante se devuelve con una cedula
      //cuyo valor es 0  @ Si no se consigue la cedula que se solicito
      //se devuelve el objeto gestorEstudiante con un valor de 1 en la
      cedula.
          public GestorInfoEstudiantes
      buscarEstudiante(GestorInfoEstudiantes gestorEstudiante,
      BufferedImage fotoTemp) {
            // Limpia la barra de estado de la pantalla principal
            pantallaPrincipal.borrarBarraEstado();

// Inicia y abre la conexión con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();

        if (conexionBD.Consulta("SELECT * FROM estudiantes WHERE
numerocedula = '"+gestorEstudiante.getCedula()+"'")){
            rs = conexionBD.MostrarDatosConsulta();
                try {
                    rs.next();
                    if (rs.getRow()!=0) {
gestorEstudiante.setNombre(rs.getString("nombres"));

gestorEstudiante.setApellido(rs.getString("apellidos"));

gestorEstudiante.setCorreoElectronico(rs.getString("email"));
```

```
gestorEstudiante.setACTIVO(rs.getBoolean("activo"));

gestorEstudiante.setSOLVETE(rs.getBoolean("solvente"));

gestorEstudiante.setCOMPUTACION(rs.getBoolean("computacion"));

gestorEstudiante.setSISTEMAS(rs.getBoolean("sistemas"));
                        byte[] imgBytes =
conexionBD.retornarFotoEstudiante(gestorEstudiante.getCedula());
                        BufferedImage image = null;
                        if (imgBytes!=null)
                             image = ArrayBufferedImage(imgBytes);
                        gestorEstudiante.setFotoEstudiante(image);

gestorEstudiante.setTelefono(rs.getString("numerodetelefono"));
                    }
                    else {
                        conexionBD.CerrarConexion();
                        gestorEstudiante.setCedula((long)1);
                        return gestorEstudiante;
                    }
                    }   catch (SQLException e) {
                            conexionBD.CerrarConexion();
                            System.out.println("Error "+e);
                            pantallaPrincipal.errorBarraEstado("Error
al acceder a la base de datos");
                            gestorEstudiante.setCedula((long)0);
                            return gestorEstudiante;
                    }
                conexionBD.CerrarConexion();
                return gestorEstudiante;
        } // end if
        else {
            conexionBD.CerrarConexion();
            gestorEstudiante.setCedula((long)1);
            return gestorEstudiante;
        } // end else
    }

//  Método que revisa toda la tabla prestamo buscando las
//coincidencias con la cedula de un estudiante para determinar cuantos
```

```
//libros tiene en su poder un estudiante.
    public int librosPrestadosAlEstudiante (String cedula){
        try {
            // Limpia la barra de estado de la pantalla principal
            pantallaPrincipal.borrarBarraEstado();
            // Inicia y abre la conexión con la base de datos
            conexionBD = new ConexionBD(pantallaPrincipal);
            conexionBD.AbrirConexion();
            conexionBD.Consulta("SELECT COUNT(*) FROM prestamo WHERE
nrocedulaestudiante ="+cedula+" AND recibido='false'");
            rs = conexionBD.MostrarDatosConsulta();
            rs.next();
            conexionBD.CerrarConexion();
            return rs.getInt("COUNT");
        } catch (SQLException ex) {

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            pantallaPrincipal.errorBarraEstado("Error al acceder a la
base de datos.");
            conexionBD.CerrarConexion();
            return -1;
        }
    }
    // Método que trasforma un array de byte en un BufferedImage y lo
///retorna @ Recibe: byte[] imageData
    // @ Retorna: BufferedImage
    public BufferedImage ArrayBufferedImage (byte[] imageData ) {
        try {
            ByteArrayInputStream in = new
ByteArrayInputStream(imageData);
            return ImageIO.read(in);
        } catch (IOException ex) {

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            return null;
        }
    }

}
```

4.5 Conclusión de la Fase de Elaboración

En esta fase se alcanzó un alto nivel de detalle en el análisis de requerimientos y diseño del sistema, así como un adelanto en la implementación. Se obtuvo una visión sólida de la arquitectura del sistema y los principales elementos de riesgo han sido abordados y resueltos, se han obtenido todos los elementos necesarios para completar la implementación del sistema en la siguiente fase.

Capítulo V: Fase de Construcción

5.1 Introducción

La finalidad principal de la fase de construcción es alcanzar la capacidad operacional del producto. Durante esta fase todos los componentes, características y requisitos identificados y detallados en las fases anteriores deben ser adecuados a las posibilidades de la herramienta de desarrollo que se va a usar, para luego ser implementados, integrados y probados en su totalidad, obteniendo una versión aceptable del sistema.

En esta fase se hace ahínco en las iteraciones de implementación y prueba del flujo de trabajo normal, debido a que su objetivo es lograr el desarrollo del software con calidad de producción, mediante la implementación de toda la funcionalidad y ejecución de las pruebas.

5.1.1 Escogencia del lenguaje de programación

Para el desarrollo del sistema SIGESLEC se debe contar con un lenguaje orientado a objetos, debido a que este paradigma permite una mayor modularidad y escalabilidad del software, así como un mayor nivel de abstracción entre los diferentes módulos que conforman el sistema. Por lo tanto se escogió el lenguaje de programación Java, de Sun Microsystems, debido a que es simple,totalmente orientado a objetos, distribuido, interpretado, robusto, seguro, portable, de altas prestaciones, multitarea y dinámico.

Adicionalmente, presenta una enorme cantidad de interfaces de programación de aplicaciones (API) que le añaden funcionalidades deseables al sistema.

Como entorno de desarrollo integrado (IDE) se escogió Netbeans 6.5. Esta herramienta le proporciona al usuario una interfaz para el diseño de interfaces graficas, auto corrección del código java mientras se realiza la codificación, capacidades de compilación y generación automática de archivos ejecutables de java (.JAR), entre otras.

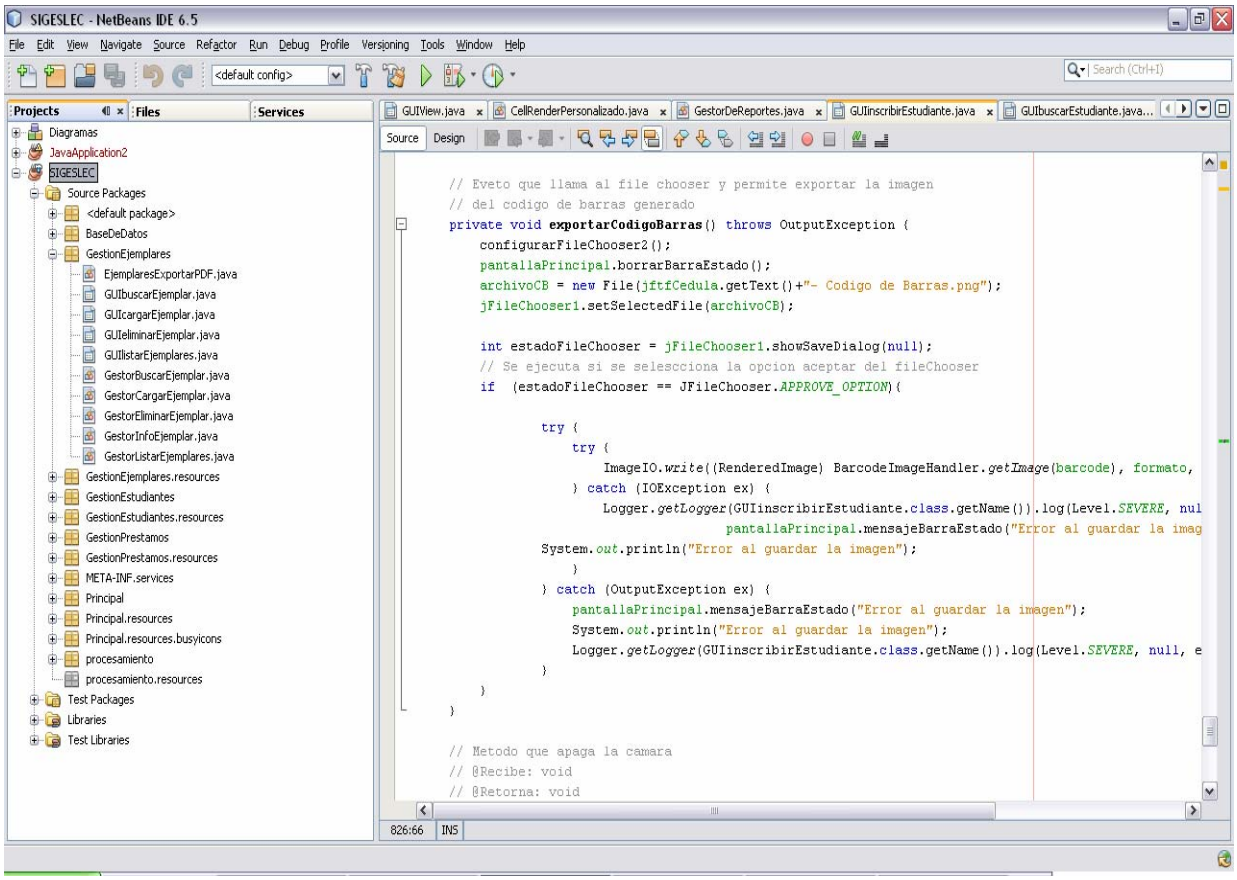

Figura 5.1: Interfaz del entorno de programación Java (Netbeans)

Fuente: Elaboración propia.

5.1.2 Escogencia del sistema gestor de base de datos

Para el desarrollo de la base de datos para el sistema SIGESLEC se utilizó PostgreSQL, particularmente su herramienta PGAdmin III, que es un sistema gestor de base de datos objeto-relacional. Entre sus características se encuentran que es software libre, multihilo, multiusuario y altamente extensible.

En la figura 5.2 se muestra el entorno en que se desarrollo la base de datos para el sistema SIGESLEC.

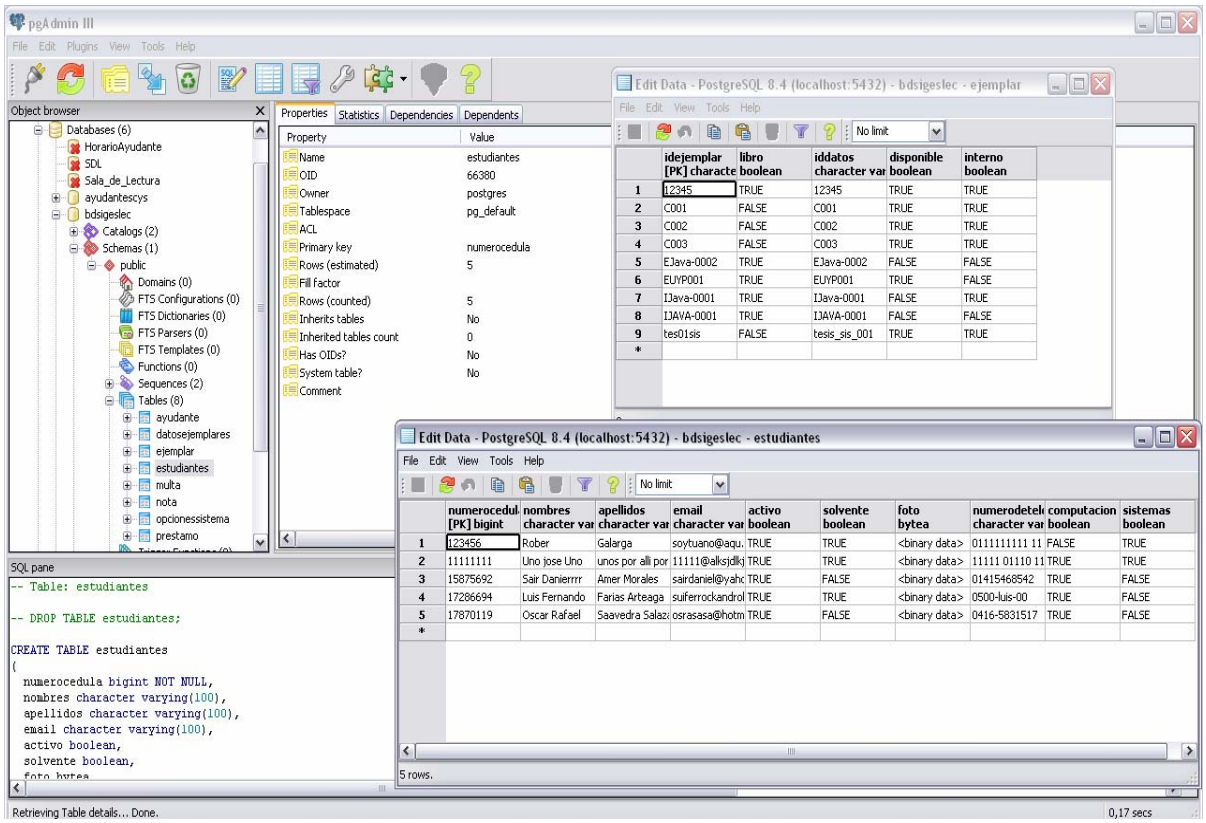

Figura 5.2: Interfaz del entorno PGAdmin III.

Fuente: Elaboración propia.

5.2 Implementación

La implementación trata al sistema en términos de desarrollo de componentes y codificación del software, su relación con la base de datos, integración de módulos y la explicación acerca de las funcionalidades del sistema y su correcto uso, revelando así toda la estructura en forma de código abierto e identificando de las actividades que este puede realizar.

5.2.1 Diagrama de Componentes Totales

En la figura 5.3 se muestra el diagrama de componentes totales, indicando la totalidad de los componentes y las clases requeridas para su funcionamiento.

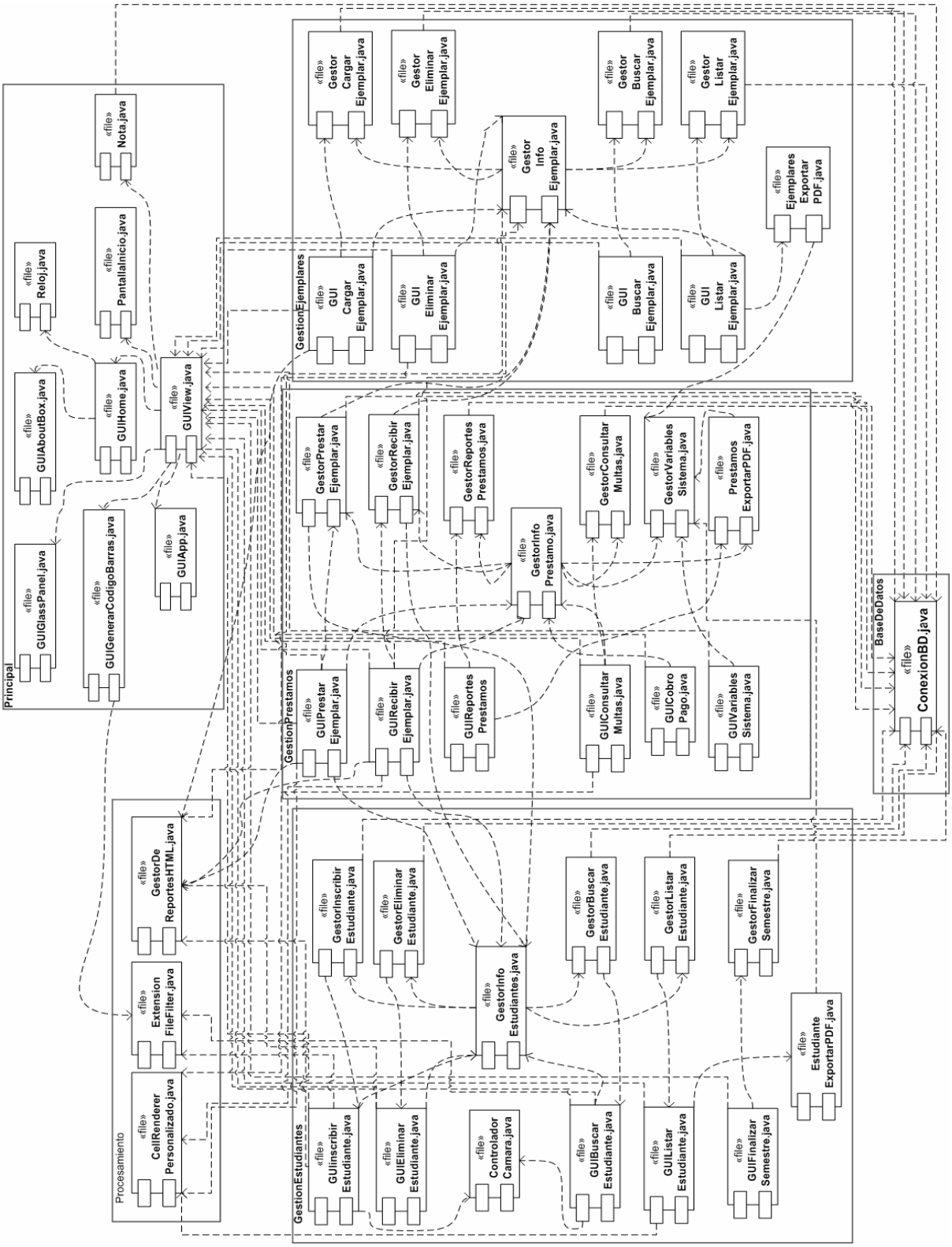

Figura 5.3: Diagrama de componentes totales para el sistema SIGESLEC.

Fuente: Elaboración propia.

5.3 Pruebas

Las pruebas son la herramienta que permite validar y comprobar el software, es decir, son los procesos que determinan si el software satisface los requisitos y trabaja de la manera establecida. Estas pruebas tienen como finalidad verificar la interacción entre los objetos, al igual que la integración adecuada de componentes, confirmar que se satisfagan los requerimientos, identificar las fallas y corregirlas antes de la instalación.

5.3.1 Pruebas por unidad

Las pruebas por unidad se aplicaron mediante la aplicación de la prueba de la caja negra sobre los diversos componentes del sistema. Para la realización de este tipo de pruebas, se identifican un conjunto de valores que pueden ser introducidos por un actor, y se expresan como clases de equivalencia para poder abarcar la totalidad de las ocurrencias de un evento de inserción de datos.

A continuación en la tabla 5.1, se representan las clases de equivalencia del componente Gestión Préstamos, el mismo se encarga de las actividades relacionadas al préstamo de un libro ó una tesis de grado.

Tabla 5.1: Clases de equivalencia del componente GUIPrestarEjemplar

Número	Campo	Clase de equivalencia	Válido	Inválido
1	Cédula	Cadena de caracteres		*
2	Cédula	Cadena alfanumérica		*
3	Cédula	Caracteres especiales		*
4	Cédula	Cadena numérica longitud <=8	*	
5	Cédula	Cadena vacía		*
6	Cota	Cadena de caracteres	*	
7	Cota	Cadena alfanumérica	*	
8	Cota	Cadena vacía		*

Fuente: Elaboración propia.

A continuación en la tabla 5.2 se muestran las clases de equivalencia para el componente GUIRecibirEjemplar, que tiene como función la recepción de un libro ó tesis de grado.

Tabla 5.2: Clases de equivalencia para el componente GUIEliminarEjemplar

Numero	Campo	Clase de equivalencia	Valido	Invalido
1	Cota	Cadena alfanumérica	*	
2	Cota	Cadena numérica	*	
3	Cota	Cadena vacía		*
4	Cota	Cadena con caracteres especiales	*	
5	Cota	Carácter Espacio (“ ”)		*

Fuente: Elaboración propia

En la tabla 5.3 y 5.4 se presentan algunos casos de prueba de caja negra, donde se incluirán datos que serán cotejados por las clases de equivalencia referenciadas anteriormente. Si se cumplen todas las clases involucra con dicho dato, entonces la salida será validada.

Tabla 5.3: Casos de prueba para el componente GUIPrestarEjemplar

Campo	Caso de prueba	Salida	Clases cubiertas
Cédula	“”	Invalido	5
Cédula	¡!qwe123	Invalido	2,3
Cédula	Hola amigo	Invalido	1
Cédula	16997256	Valido	4
Cota	“”	Invalido	8
Cota	Sis-007	Valido	6,7
Cota	875/660	Valido	6,7
Cota	Ejemplar01	Valido	6,7

Fuente: Elaboración propia.

Tabla 5.4: Casos de prueba para el componente GUIEliminarEjemplar

Fuente: Elaboración propia.

Campo	Caso de prueba	Salida	Clases Cubiertas
Cota	Caracter espacio (“ ”)	Invalido	5
Cota	Carácter vacío (“”)	Invalido	3
Cota	Qwerty12345	Valido	1
Cota	18999000’’’	Valido	1,3
Cota	¡”·$%&	Valido	3

5.3.2 Pruebas de Integración

El objetivo general de las pruebas de integración es detectar las fallas de interacción entre las distintas clases que conforman al sistema. Debido a que cada clase examinada por separado se inserta de manera gradual dentro de la estructura, las pruebas de integración son realmente un mecanismo para verificar el correcto ensamblaje del sistema como un todo. Al efectuar la integración de los módulos, se concentra el esfuerzo en la búsqueda de fallas que puedan generar excepciones arrojadas por los métodos; el empleo de operaciones equivocadas, e invocación incorrecta de los métodos.

Luego de verificar la calidad de los componentes, se procede a comprobar la eficiencia de un conjunto de componentes integrados por fase, estas se pueden observar distinguidas en la figura 5.4, en donde se muestra el diagrama de componentes por fase de integración del sistema.

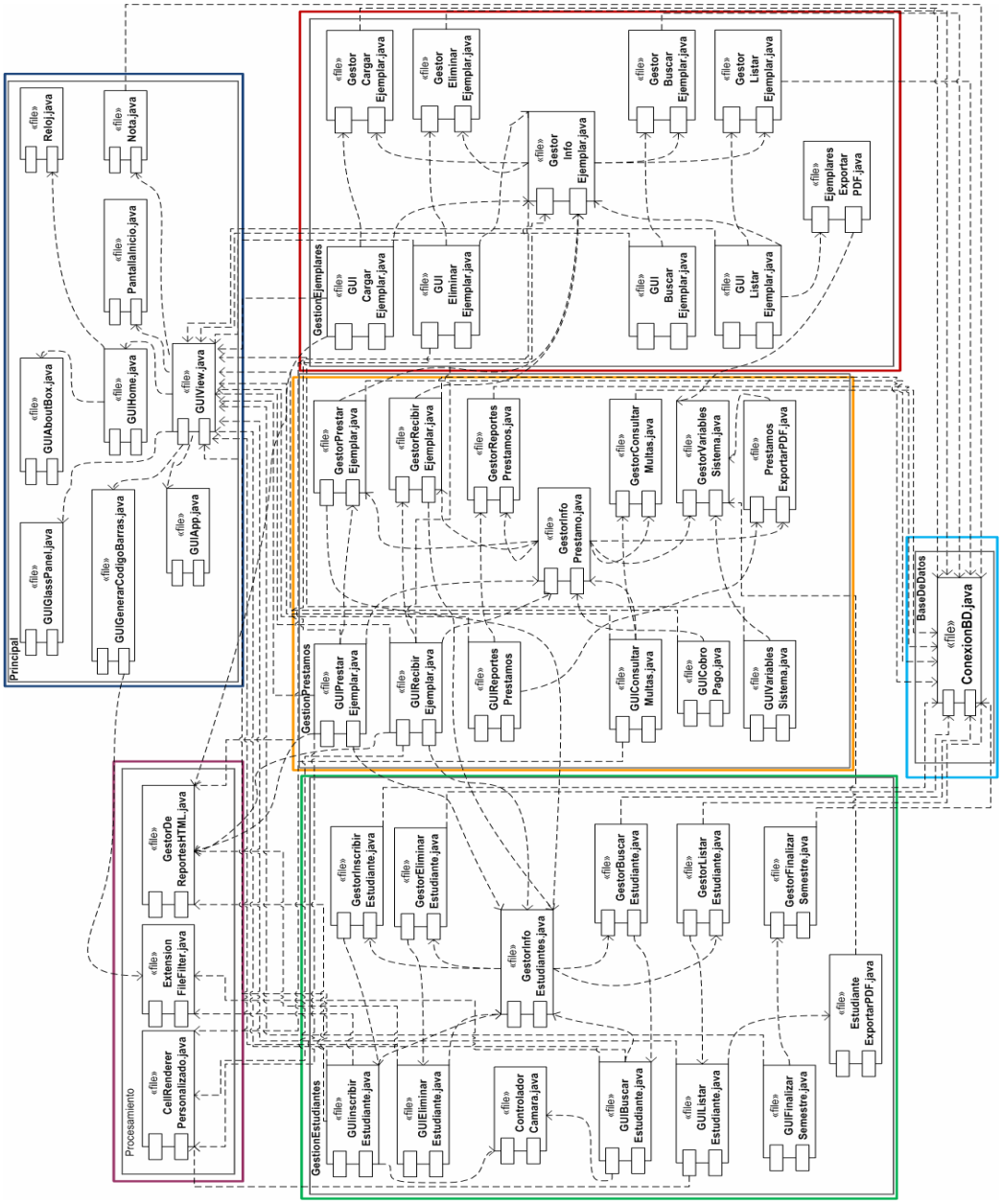

Figura 5.4: Diagrama de componentes por fase de integración.

Fuente: Elaboración propia

Tabla 5.5: Leyenda de colores para fases del diagrama de componentes totales

Color	Fase	Color	Fase
	1		4
	2		5
	3		6

Fuente: Elaboración propia.

5.3.2.1 Integración de Gestión Prestamos

Este caso de prueba verifica la funcionalidad de la implementación de la fase 5.

Tabla 5.6: Caso de prueba para la fase 5

Campo	Datos
Cédula	15875692
Cota	EUYP0001

Fuente: Elaboración propia

5.3.2.1.1 Resultados

- Se encontró necesaria la creación de una rutina para limitar la longitud del campo Cota para evitar errores en su ingreso al sistema por parte de los usuarios.
-Luego de esta corrección no se hallaron más errores en esta fase.

5.3.2.1.2 Procedimiento de prueba

- Activar la interfaz principal.
- Acceder al menú préstamo y seleccionar la opción prestar Ejemplar.
- Llenar el campo “Cédula” con el dato contemplado en la tabla 5.6.
- Presionar el botón buscar.
- Se activa la parte de la interfaz relativa al préstamo de ejemplares.
- Llenar el campo “Cota” con el dato contemplado en la tabla 5.6.
- Presionar el botón Añadir.
-Finalmente, presionar el botón Prestar.

A continuación se muestra un conjunto de imágenes que ilustran el procedimiento:

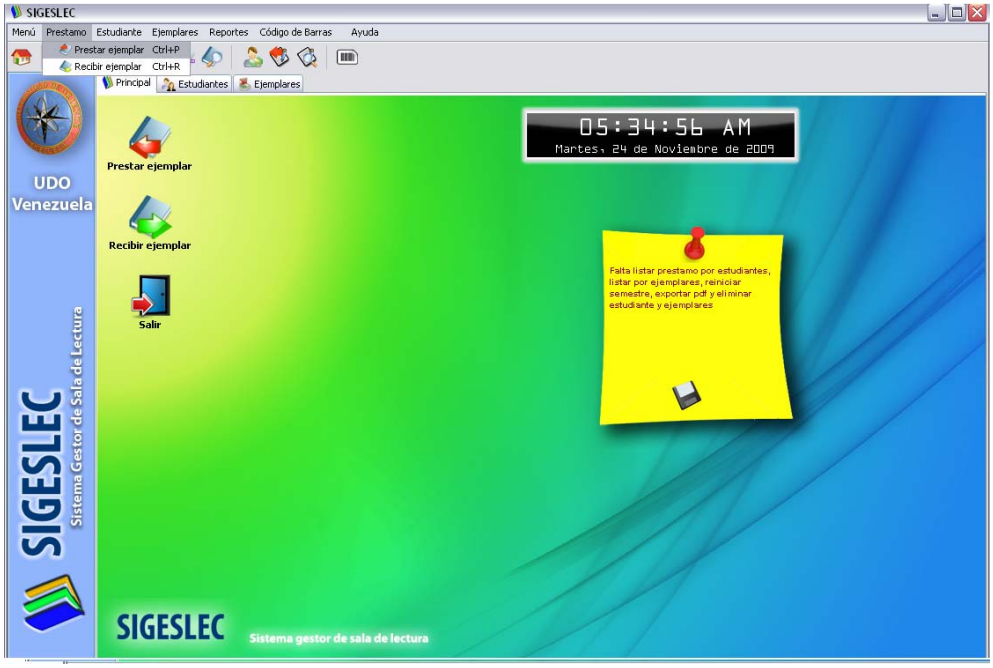

Figura 5.5: Componente principal recibiendo la petición de préstamo

Fuente: Elaboración propia.

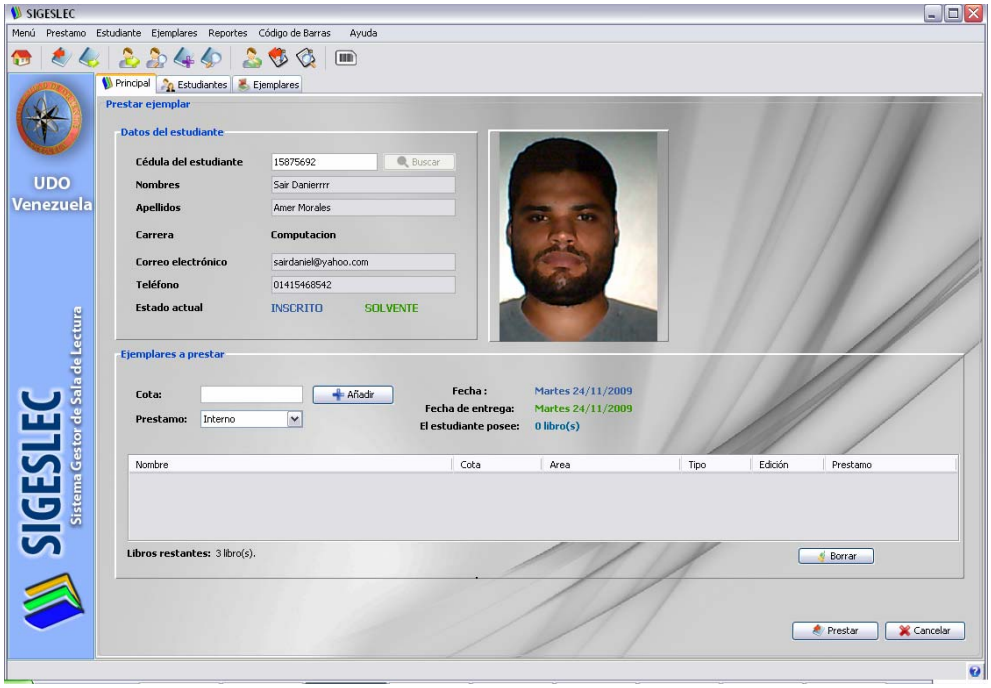

Figura 5.6: Componente GUIPrestarEjemplar recibiendo el número de cédula

Fuente: Elaboración propia.

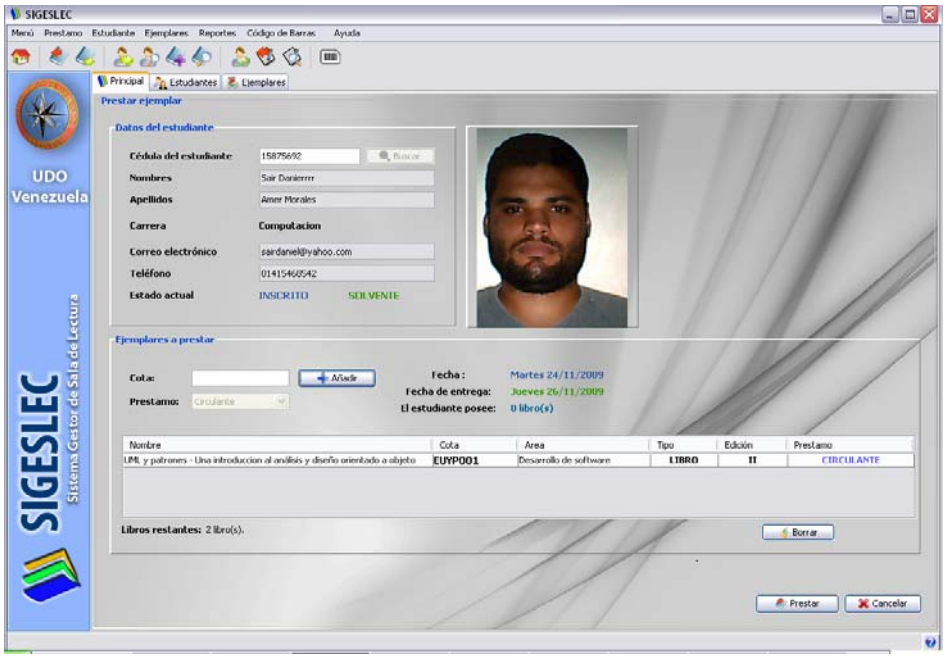

Figura 5.7: Componente GUIPrestarEjemplar recibiendo el dato Cota.

Fuente: Elaboración propia.

5.3.2.2 Código de componentes en la fase de Gestión de préstamos

5.3.2.2.1 GUIPrestarEjemplar

```
package GestionPrestamos;

import GestionEjemplares.GestorInfoEjemplar;
import GestionEstudiantes.GestorInfoEstudiantes;
import Principal.GUIView;
import java.awt.Color;
import java.awt.Cursor;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.event.KeyEvent;
import java.awt.image.BufferedImage;
import java.util.Calendar;
import javax.swing.ImageIcon;
import javax.swing.JFormattedTextField;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableColumn;
import procesamiento.CellRenderPersonalizado;

public class GUIprestarEjemplar extends javax.swing.JPanel {
    private GUIView pantallaPrincipal;
    private Color Azul;
    private Color Verde;
    private Color Rojo;
```

```
    private boolean prestamoActivo;
    private int cantidadLibrosPrestar;
    private TableColumn column;
    private String path;
    private ImageIcon imagenFondo;

    public GUIprestarEjemplar(GUIView pantallaPrincipal) {
        initComponents();
        this.pantallaPrincipal = pantallaPrincipal;
        this.setVisible(true);
        // Incializar colores
        Azul = new Color(51,94,168);
        Verde = new Color (51,153,0);
        Rojo = new Color (255, 51, 51);
        // Configura el FormattedTextField
        jftfCedula.setFocusLostBehavior(JFormattedTextField.PERSIST);
        // Inicializa gestores de información de préstamo y  de
        // variables del sistema
        gestorVariableSistema = new
GestorVariablesSistema(pantallaPrincipal);
        gestorInfoPrestamo = new GestorInfoPrestamo();

        // Personaliza la apariencia y ajusta el tamaño de la columnas
        // de la tabla que contiene la lista de los ejemplares

jtListaEjemplares.setDefaultRenderer(jtListaEjemplares.getColumnClass(
0),new CellRenderPersonalizado(3));
        ajustarAnchoTabla ();
        cargarIconos();
    }

    private void cargarIconos() {
        imagenFondo = new ImageIcon
(GUIView.class.getResource("resources/fondo.png"));
    }

    // Método para dibujar una imagen de fondo en el jPanel del
formulario
    // que se esta desplegando

    public void paintComponent(Graphics g)
     {
        Graphics2D g2 = (Graphics2D)g;

g2.drawImage(imagenFondo.getImage(),0,0,this.getWidth(),this.getHeight
(),this);

    }
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        buttonGroup1 = new javax.swing.ButtonGroup();
        jPopupMenu1 = new javax.swing.JPopupMenu();
        jMenuItem1 = new javax.swing.JMenuItem();
        jMenuItem2 = new javax.swing.JMenuItem();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jPanel2 = new javax.swing.JPanel();
        jPanel3 = new javax.swing.JPanel();
        jLabel7 = new javax.swing.JLabel();
```

```
        jLabel2 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jPanel5 = new javax.swing.JPanel();
        jtfNombres = new javax.swing.JTextField();
        jtfApellidos = new javax.swing.JTextField();
        jtfCorreoElectronico = new javax.swing.JTextField();
        jtfTelefono = new javax.swing.JTextField();
        botonConsultar = new javax.swing.JButton();
        jlCarrera = new javax.swing.JLabel();
        jlActivo = new javax.swing.JLabel();
        jlSolvente = new javax.swing.JLabel();
        jftfCedula = new javax.swing.JFormattedTextField();
        jPanel6 = new javax.swing.JPanel();
        jPanel7 = new javax.swing.JPanel();
        jtfCota = new javax.swing.JTextField();
        jLabel6 = new javax.swing.JLabel();
        botonAñadir = new javax.swing.JButton();
        jcbPrestamo = new javax.swing.JComboBox();
        jLabel9 = new javax.swing.JLabel();
        jScrollPane2 = new javax.swing.JScrollPane();
        jtListaEjemplares = new javax.swing.JTable();
        jPanel8 = new javax.swing.JPanel();
        jLabel10 = new javax.swing.JLabel();
        jlFecha = new javax.swing.JLabel();
        jLabel12 = new javax.swing.JLabel();
        jlFechaEntrega = new javax.swing.JLabel();
        jLabel13 = new javax.swing.JLabel();
        jlLibrosPrestados = new javax.swing.JLabel();
        jLabel11 = new javax.swing.JLabel();
        jlLibrosRestantes = new javax.swing.JLabel();
        jButton3 = new javax.swing.JButton();
        panelCamara = new javax.swing.JPanel();
        labelFoto = new javax.swing.JLabel();

        jPopupMenu1.setName("jPopupMenu1"); // NOI18N

        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(Principal.GUIApp.clas
s).getContext().getResourceMap(GUIprestarEjemplar.class);
        jMenuItem1.setText(resourceMap.getString("jMenuItem1.text"));
// NOI18N
        jMenuItem1.setName("jMenuItem1"); // NOI18N
        jMenuItem1.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jMenuItem1ActionPerformed(evt);
            }
        });
        jPopupMenu1.add(jMenuItem1);

        jMenuItem2.setText(resourceMap.getString("jMenuItem2.text"));
// NOI18N
        jMenuItem2.setName("jMenuItem2"); // NOI18N
        jMenuItem2.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
```

```
evt) {
                jMenuItem2ActionPerformed(evt);
            }
        });
        jPopupMenu1.add(jMenuItem2);

        setBorder(javax.swing.BorderFactory.createTitledBorder(null,
resourceMap.getString("Form.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("Form.border.titleFont"))); // NOI18N
        setToolTipText(resourceMap.getString("Form.toolTipText")); //
NOI18N
        setName("Form"); // NOI18N
        setPreferredSize(new java.awt.Dimension(700, 600));

        jButton1.setIcon(resourceMap.getIcon("jButton1.icon")); //
NOI18N
        jButton1.setText(resourceMap.getString("jButton1.text")); //
NOI18N
        jButton1.setName("jButton1"); // NOI18N

        jButton2.setIcon(resourceMap.getIcon("jButton2.icon")); //
NOI18N
        jButton2.setText(resourceMap.getString("jButton2.text")); //
NOI18N
        jButton2.setName("jButton2"); // NOI18N
        jButton2.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton2ActionPerformed(evt);
            }
        });

        jPanel2.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel2.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel2.border.titleFont"))); // NOI18N
        jPanel2.setMaximumSize(new java.awt.Dimension(500, 147));
        jPanel2.setMinimumSize(new java.awt.Dimension(500, 147));
        jPanel2.setName("jPanel2"); // NOI18N
        jPanel2.setOpaque(false);

        jPanel3.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel3.setName("jPanel3"); // NOI18N
        jPanel3.setOpaque(false);

        jLabel7.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N
        jLabel7.setText(resourceMap.getString("jLabel7.text")); //
NOI18N
        jLabel7.setName("jLabel7"); // NOI18N

        jLabel2.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N
        jLabel2.setText(resourceMap.getString("jLabel2.text")); //
```

```
NOI18N
        jLabel2.setName("jLabel2"); // NOI18N

        jLabel4.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N
        jLabel4.setText(resourceMap.getString("jLabel4.text")); //
NOI18N
        jLabel4.setName("jLabel4"); // NOI18N

        jLabel3.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N
        jLabel3.setText(resourceMap.getString("jLabel3.text")); //
NOI18N
        jLabel3.setName("jLabel3"); // NOI18N

        jLabel5.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N
        jLabel5.setText(resourceMap.getString("jLabel5.text")); //
NOI18N
        jLabel5.setName("jLabel5"); // NOI18N

        jLabel1.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N
        jLabel1.setText(resourceMap.getString("jLabel1.text")); //
NOI18N
        jLabel1.setName("jLabel1"); // NOI18N

        jLabel8.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N
        jLabel8.setText(resourceMap.getString("jLabel8.text")); //
NOI18N
        jLabel8.setName("jLabel8"); // NOI18N

        javax.swing.GroupLayout jPanel3Layout = new
javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING, false)
                    .addComponent(jLabel8,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jLabel5,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jLabel3,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jLabel2,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                    .addComponent(jLabel4,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jLabel7,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
        );
        jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel7,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel4,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(11, 11, 11)
                .addComponent(jLabel3,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jLabel5,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        jPanel5.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel5.setName("jPanel5"); // NOI18N
        jPanel5.setOpaque(false);

        jtfNombres.setEditable(false);
```

```
jtfNombres.setToolTipText(resourceMap.getString("jtfNombres.toolTipTex
t")); // NOI18N
        jtfNombres.setName("jtfNombres"); // NOI18N

        jtfApellidos.setEditable(false);

jtfApellidos.setToolTipText(resourceMap.getString("jtfApellidos.toolTi
pText")); // NOI18N
        jtfApellidos.setName("jtfApellidos"); // NOI18N

        jtfCorreoElectronico.setEditable(false);

jtfCorreoElectronico.setToolTipText(resourceMap.getString("jtfCorreoEl
ectronico.toolTipText")); // NOI18N
        jtfCorreoElectronico.setName("jtfCorreoElectronico"); //
NOI18N

        jtfTelefono.setEditable(false);

jtfTelefono.setToolTipText(resourceMap.getString("jtfTelefono.toolTipT
ext")); // NOI18N
        jtfTelefono.setName("jtfTelefono"); // NOI18N


botonConsultar.setIcon(resourceMap.getIcon("botonConsultar.icon")); //
NOI18N

botonConsultar.setText(resourceMap.getString("botonConsultar.text"));
// NOI18N

botonConsultar.setToolTipText(resourceMap.getString("botonConsultar.to
olTipText")); // NOI18N
        botonConsultar.setName("botonConsultar"); // NOI18N
        botonConsultar.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                botonConsultarActionPerformed(evt);
            }
        });

        jlCarrera.setFont(resourceMap.getFont("jlCarrera.font")); //
NOI18N
        jlCarrera.setText(resourceMap.getString("jlCarrera.text")); //
NOI18N

jlCarrera.setToolTipText(resourceMap.getString("jlCarrera.toolTipText"
)); // NOI18N

jlCarrera.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER)
;
        jlCarrera.setName("jlCarrera"); // NOI18N

        jlActivo.setFont(resourceMap.getFont("jlActivo.font")); //
NOI18N

jlActivo.setToolTipText(resourceMap.getString("jlActivo.toolTipText"))
; // NOI18N

jlActivo.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
```

```
        jlActivo.setName("jlActivo"); // NOI18N
        jlActivo.setPreferredSize(new java.awt.Dimension(0, 20));

        jlSolvente.setFont(resourceMap.getFont("jlSolvente.font")); //
NOI18N

jlSolvente.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER
);
        jlSolvente.setName("jlSolvente"); // NOI18N
        jlSolvente.setPreferredSize(new java.awt.Dimension(0, 20));

        try {
            jftfCedula.setFormatterFactory(new
javax.swing.text.DefaultFormatterFactory(new
javax.swing.text.MaskFormatter("#########")));
        } catch (java.text.ParseException ex) {
            ex.printStackTrace();
        }
        jftfCedula.setText(resourceMap.getString("jftfCedula.text"));
// NOI18N

jftfCedula.setToolTipText(resourceMap.getString("jftfCedula.toolTipTex
t")); // NOI18N

jftfCedula.setFocusLostBehavior(javax.swing.JFormattedTextField.PERSIS
T);
        jftfCedula.setName("jftfCedula"); // NOI18N
        jftfCedula.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jftfCedulaActionPerformed(evt);
            }
        });
        jftfCedula.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyPressed(java.awt.event.KeyEvent evt) {
                jftfCedulaKeyPressed(evt);
            }
        });

        javax.swing.GroupLayout jPanel5Layout = new
javax.swing.GroupLayout(jPanel5);
        jPanel5.setLayout(jPanel5Layout);
        jPanel5Layout.setHorizontalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jtfApellidos,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 210, Short.MAX_VALUE)
                    .addComponent(jtfNombres,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 210, Short.MAX_VALUE)
                    .addGroup(jPanel5Layout.createSequentialGroup()
                        .addComponent(jftfCedula,
javax.swing.GroupLayout.PREFERRED_SIZE, 121,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(botonConsultar,
javax.swing.GroupLayout.DEFAULT_SIZE, 83, Short.MAX_VALUE))
                    .addComponent(jtfTelefono,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 210, Short.MAX_VALUE)
                    .addComponent(jlCarrera,
javax.swing.GroupLayout.DEFAULT_SIZE, 210, Short.MAX_VALUE)
                    .addComponent(jtfCorreoElectronico,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 210, Short.MAX_VALUE)
                    .addGroup(jPanel5Layout.createSequentialGroup()
                        .addComponent(jlActivo,
javax.swing.GroupLayout.PREFERRED_SIZE, 101,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jlSolvente,
javax.swing.GroupLayout.PREFERRED_SIZE, 101,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap())
        );
        jPanel5Layout.setVerticalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(botonConsultar,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jftfCedula,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jtfNombres,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jtfApellidos,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jlCarrera,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jtfCorreoElectronico,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jtfTelefono,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jlActivo,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jlSolvente,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel3,
javax.swing.GroupLayout.PREFERRED_SIZE, 148,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(6, 6, 6)
                .addComponent(jPanel5,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING, false)
                    .addComponent(jPanel3,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel5,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
```

```
Short.MAX_VALUE))
        );

        jPanel6.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel6.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel6.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel6.border.titleFont"))); // NOI18N
        jPanel6.setName("jPanel6"); // NOI18N
        jPanel6.setOpaque(false);

        jPanel7.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel7.setName("jPanel7"); // NOI18N
        jPanel7.setOpaque(false);

        jtfCota.setText(resourceMap.getString("jtfCota.text")); //
NOI18N

jtfCota.setToolTipText(resourceMap.getString("jtfCota.toolTipText"));
// NOI18N
        jtfCota.setEnabled(false);
        jtfCota.setName("jtfCota"); // NOI18N
        jtfCota.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusGained(java.awt.event.FocusEvent evt) {
                jtfCotaFocusGained(evt);
            }
        });
        jtfCota.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyPressed(java.awt.event.KeyEvent evt) {
                jtfCotaKeyPressed(evt);
            }
        });

        jLabel6.setFont(resourceMap.getFont("jLabel6.font")); //
NOI18N
        jLabel6.setText(resourceMap.getString("jLabel6.text")); //
NOI18N
        jLabel6.setName("jLabel6"); // NOI18N

        botonAñadir.setIcon(resourceMap.getIcon("botonAñadir.icon"));
// NOI18N

botonAñadir.setText(resourceMap.getString("botonAñadir.text")); //
NOI18N

botonAñadir.setToolTipText(resourceMap.getString("botonAñadir.toolTipT
ext")); // NOI18N
        botonAñadir.setEnabled(false);
        botonAñadir.setName("botonAñadir"); // NOI18N
        botonAñadir.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                botonAñadirActionPerformed(evt);
            }
        });

        jcbPrestamo.setModel(new javax.swing.DefaultComboBoxModel(new
```

```
String[] { "Interno", "Circulante" }));

jcbPrestamo.setToolTipText(resourceMap.getString("jcbPrestamo.toolTipT
ext")); // NOI18N
        jcbPrestamo.setEnabled(false);
        jcbPrestamo.setName("jcbPrestamo"); // NOI18N
        jcbPrestamo.addItemListener(new java.awt.event.ItemListener()
{
            public void itemStateChanged(java.awt.event.ItemEvent evt)
{
                jcbPrestamoItemStateChanged(evt);
            }
        });

        jLabel9.setFont(resourceMap.getFont("jLabel9.font")); //
NOI18N
        jLabel9.setText(resourceMap.getString("jLabel9.text")); //
NOI18N
        jLabel9.setName("jLabel9"); // NOI18N

        javax.swing.GroupLayout jPanel7Layout = new
javax.swing.GroupLayout(jPanel7);
        jPanel7.setLayout(jPanel7Layout);
        jPanel7Layout.setHorizontalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jLabel6,
javax.swing.GroupLayout.PREFERRED_SIZE, 69,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel9,
javax.swing.GroupLayout.PREFERRED_SIZE, 69,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING)
                    .addComponent(jtfCota,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.PREFERRED_SIZE, 117,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jcbPrestamo,
javax.swing.GroupLayout.PREFERRED_SIZE, 117,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(botonAñadir,
javax.swing.GroupLayout.DEFAULT_SIZE, 94, Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel7Layout.setVerticalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel7Layout.createSequentialGroup()
```

```
                .addContainerGap()

.addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(jLabel6,
javax.swing.GroupLayout.PREFERRED_SIZE, 21,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jtfCota,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(botonAñadir))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(jcbPrestamo,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel9,
javax.swing.GroupLayout.PREFERRED_SIZE, 21,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(15, Short.MAX_VALUE))
        );


jScrollPane2.setBorder(javax.swing.BorderFactory.createEtchedBorder())
;
        jScrollPane2.setName("jScrollPane2"); // NOI18N

        jtListaEjemplares.setAutoCreateRowSorter(true);
        jtListaEjemplares.setModel(new
javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Nombre", "Cota", "Area", "Tipo", "Edición",
"Prestamo"
            }
        ) {
            Class[] types = new Class [] {
                java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class
            };
            boolean[] canEdit = new boolean [] {
                false, false, false, false, false, false
            };

            public Class getColumnClass(int columnIndex) {
                return types [columnIndex];
            }

            public boolean isCellEditable(int rowIndex, int
columnIndex) {
                return canEdit [columnIndex];
            }
        });
```

```
jtListaEjemplares.setToolTipText(resourceMap.getString("jtListaEjempla
res.toolTipText")); // NOI18N

jtListaEjemplares.setAutoResizeMode(javax.swing.JTable.AUTO_RESIZE_ALL
_COLUMNS);

jtListaEjemplares.setDebugGraphicsOptions(javax.swing.DebugGraphics.NO
NE_OPTION);
        jtListaEjemplares.setName("jtListaEjemplares"); // NOI18N

jtListaEjemplares.setSelectionMode(javax.swing.ListSelectionModel.SING
LE_SELECTION);

jtListaEjemplares.getTableHeader().setReorderingAllowed(false);
        jScrollPane2.setViewportView(jtListaEjemplares);

        jPanel8.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel8.setName("jPanel8"); // NOI18N
        jPanel8.setOpaque(false);

        jLabel10.setFont(resourceMap.getFont("jLabel10.font")); //
NOI18N

jLabel10.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel10.setText(resourceMap.getString("jLabel10.text")); //
NOI18N
        jLabel10.setName("jLabel10"); // NOI18N

        jlFecha.setFont(resourceMap.getFont("jlFecha.font")); //
NOI18N

jlFecha.setForeground(resourceMap.getColor("jlFecha.foreground")); //
NOI18N

jlFecha.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
        jlFecha.setText(resourceMap.getString("jlFecha.text")); //
NOI18N
        jlFecha.setName("jlFecha"); // NOI18N

        jLabel12.setFont(resourceMap.getFont("jLabel12.font")); //
NOI18N

jLabel12.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel12.setText(resourceMap.getString("jLabel12.text")); //
NOI18N
        jLabel12.setName("jLabel12"); // NOI18N

        jlFechaEntrega.setFont(resourceMap.getFont("jlFecha.font"));
// NOI18N

jlFechaEntrega.setForeground(resourceMap.getColor("jlFecha.foreground"
)); // NOI18N

jlFechaEntrega.setHorizontalAlignment(javax.swing.SwingConstants.LEFT)
;

jlFechaEntrega.setText(resourceMap.getString("jlFechaEntrega.text"));
// NOI18N
        jlFechaEntrega.setName("jlFechaEntrega"); // NOI18N
```

```
        jLabel13.setFont(resourceMap.getFont("jLabel13.font")); //
NOI18N

jLabel13.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel13.setText(resourceMap.getString("jLabel13.text")); //
NOI18N
        jLabel13.setName("jLabel13"); // NOI18N


jlLibrosPrestados.setFont(resourceMap.getFont("jlLibrosPrestados.font"
)); // NOI18N

jlLibrosPrestados.setForeground(resourceMap.getColor("jlLibrosPrestado
s.foreground")); // NOI18N

jlLibrosPrestados.setText(resourceMap.getString("jlLibrosPrestados.tex
t")); // NOI18N
        jlLibrosPrestados.setName("jlLibrosPrestados"); // NOI18N

        javax.swing.GroupLayout jPanel8Layout = new
javax.swing.GroupLayout(jPanel8);
        jPanel8.setLayout(jPanel8Layout);
        jPanel8Layout.setHorizontalGroup(

jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel8Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING, false)
                    .addComponent(jLabel13,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jLabel10,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jLabel12,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addGap(18, 18, 18)

.addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)

.addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING, false)
                        .addComponent(jlFechaEntrega,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jlFecha,
javax.swing.GroupLayout.PREFERRED_SIZE, 142,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jlLibrosPrestados,
javax.swing.GroupLayout.PREFERRED_SIZE, 87,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(33, Short.MAX_VALUE))
        );
        jPanel8Layout.setVerticalGroup(
```

```
jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel8Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addGroup(jPanel8Layout.createSequentialGroup()
                        .addComponent(jlFecha)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jlFechaEntrega)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jlLibrosPrestados))
                    .addGroup(jPanel8Layout.createSequentialGroup()
                        .addComponent(jLabel10)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel12)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel13,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                .addContainerGap())
        );

        jLabel11.setFont(resourceMap.getFont("jLabel11.font")); //
NOI18N
        jLabel11.setText(resourceMap.getString("jLabel11.text")); //
NOI18N
        jLabel11.setName("jLabel11"); // NOI18N


jlLibrosRestantes.setText(resourceMap.getString("jlLibrosRestantes.tex
t")); // NOI18N

jlLibrosRestantes.setToolTipText(resourceMap.getString("jlLibrosRestan
tes.toolTipText")); // NOI18N
        jlLibrosRestantes.setName("jlLibrosRestantes"); // NOI18N

        jButton3.setIcon(resourceMap.getIcon("jButton3.icon")); //
NOI18N
        jButton3.setText(resourceMap.getString("jButton3.text")); //
NOI18N

jButton3.setToolTipText(resourceMap.getString("jButton3.toolTipText"))
; // NOI18N
        jButton3.setName("jButton3"); // NOI18N
        jButton3.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton3ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel6Layout = new
javax.swing.GroupLayout(jPanel6);
        jPanel6.setLayout(jPanel6Layout);
```

```
        jPanel6Layout.setHorizontalGroup(

jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel6Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel6Layout.createSequentialGroup()

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                            .addComponent(jScrollPane2,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 640, Short.MAX_VALUE)

.addGroup(jPanel6Layout.createSequentialGroup()
                                .addComponent(jPanel7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                                .addComponent(jPanel8,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel6Layout.createSequentialGroup()
                        .addComponent(jLabel11)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jlLibrosRestantes,
javax.swing.GroupLayout.PREFERRED_SIZE, 71,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
283, Short.MAX_VALUE)
                        .addComponent(jButton3,
javax.swing.GroupLayout.PREFERRED_SIZE, 88,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(97, 97, 97)))
                .addGap(0, 0, 0))
        );
        jPanel6Layout.setVerticalGroup(

jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel6Layout.createSequentialGroup()
                .addGap(11, 11, 11)

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jPanel8,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                    .addComponent(jPanel7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(14, 14, 14)
                .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jButton3,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                        .addComponent(jLabel11)
                        .addComponent(jlLibrosRestantes)))
                .addGap(11, 11, 11))
        );

        panelCamara.setBackground(new java.awt.Color(0x00ffffff, true
));
        panelCamara.setBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED));
        panelCamara.setMaximumSize(new java.awt.Dimension(320, 240));
        panelCamara.setName("panelCamara"); // NOI18N
        panelCamara.setPreferredSize(new java.awt.Dimension(320,
240));


labelFoto.setBackground(resourceMap.getColor("labelFoto.background"));
// NOI18N
        labelFoto.setIcon(resourceMap.getIcon("labelFoto.icon")); //
NOI18N
        labelFoto.setName("labelFoto"); // NOI18N

        javax.swing.GroupLayout panelCamaraLayout = new
javax.swing.GroupLayout(panelCamara);
        panelCamara.setLayout(panelCamaraLayout);
        panelCamaraLayout.setHorizontalGroup(

panelCamaraLayout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
            .addGroup(panelCamaraLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(labelFoto)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        panelCamaraLayout.setVerticalGroup(

panelCamaraLayout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
            .addComponent(labelFoto,
javax.swing.GroupLayout.DEFAULT_SIZE, 240, Short.MAX_VALUE)
        );
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                    .addComponent(jPanel6,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 418,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(panelCamara,
javax.swing.GroupLayout.PREFERRED_SIZE, 206,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                        .addComponent(jButton2,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jButton1)))
                .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.TRAILING)
                    .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(panelCamara,
javax.swing.GroupLayout.PREFERRED_SIZE, 242,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel6,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
```

```
.BASELINE)
                        .addComponent(jButton1)
                        .addComponent(jButton2))
                    .addContainerGap())
        );
    }// </editor-fold>


    private void
botonConsultarActionPerformed(java.awt.event.ActionEvent evt) {
        consultarCedula();
}

    private void jftfCedulaActionPerformed(java.awt.event.ActionEvent
evt) {
        consultarCedula();
}

    private void jftfCedulaKeyPressed(java.awt.event.KeyEvent evt) {
        if (evt.getKeyCode()==evt.VK_ENTER)
            consultarCedula();
}

    private void jcbPrestamoItemStateChanged(java.awt.event.ItemEvent
evt) {
        activarPrestamo ();
    }

    private void botonAñadirActionPerformed(java.awt.event.ActionEvent
evt) {
        cargarEjemplar();
    }

    private void jtfCotaKeyPressed(java.awt.event.KeyEvent evt) {
        if (evt.getKeyCode()== KeyEvent.VK_ENTER)
            cargarEjemplar();
    }

    private void jtfCotaFocusGained(java.awt.event.FocusEvent evt) {
        jtfCota.setBackground(Color.white);
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent
evt) {

        jPopupMenu1.setVisible(true);

jPopupMenu1.setLocation((int)jButton3.getLocationOnScreen().getX()+jBu
tton3.getWidth(),(int)jButton3.getLocationOnScreen().getY());

    }

    private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent
evt) {
        borrarPrestamos();
    }

    private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent
evt) {
        borrarTodo();
    }
```

```
    private void jButton2ActionPerformed(java.awt.event.ActionEvent
evt) {

this.setCursor(Cursor.getPredefinedCursor(Cursor.WAIT_CURSOR));
        prestarEjemplares();

this.setCursor(Cursor.getPredefinedCursor(Cursor.DEFAULT_CURSOR));
    }


    // Variables declaration - do not modify
    private javax.swing.JButton botonAñadir;
    private javax.swing.JButton botonConsultar;
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JMenuItem jMenuItem2;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel5;
    private javax.swing.JPanel jPanel6;
    private javax.swing.JPanel jPanel7;
    private javax.swing.JPanel jPanel8;
    private javax.swing.JPopupMenu jPopupMenu1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JComboBox jcbPrestamo;
    private javax.swing.JFormattedTextField jftfCedula;
    private javax.swing.JLabel jlActivo;
    private javax.swing.JLabel jlCarrera;
    private javax.swing.JLabel jlFecha;
    private javax.swing.JLabel jlFechaEntrega;
    private javax.swing.JLabel jlLibrosPrestados;
    private javax.swing.JLabel jlLibrosRestantes;
    private javax.swing.JLabel jlSolvente;
    private javax.swing.JTable jtListaEjemplares;
    private javax.swing.JTextField jtfApellidos;
    private javax.swing.JTextField jtfCorreoElectronico;
    private javax.swing.JTextField jtfCota;
    private javax.swing.JTextField jtfNombres;
    private javax.swing.JTextField jtfTelefono;
    private javax.swing.JLabel labelFoto;
    private javax.swing.JPanel panelCamara;
    // End of variables declaration

    private GestorInfoEstudiantes gestorEstudiantes;
```

```
    private GestorInfoEjemplar gestorEjemplar;
    private GestorPrestarEjemplar gestorPrestamosEjemplares;
    private GestorVariablesSistema gestorVariableSistema;
    private GestorInfoPrestamo gestorInfoPrestamo;
    private BufferedImage fotoTemp;


    // Busca la cedula en la base de datos y despliega la informacion
    // basica del estudiante verificacdo ademas que este solvente e
    // inscrito
    private void consultarCedula (){
        // Limpia la barra de estado y verifica si el capo cedula esta
        // vacio, en caso de que asi sea retornara vacio y finalizara
        // el metodo
        pantallaPrincipal.borrarBarraEstado();
        limpiarFormularioEstudiante();

        if (jftfCedula.getText().trim().equals("")){
            pantallaPrincipal.advertenciaBarraEstado("La cedula debe
contener al menos un caracter numérico.");
            jftfCedula.setBackground(Color.pink);
            return ;
        }

        // Inicializa el gestorEstudiantes y gestorInscribirEstudiante
        gestorEstudiantes = new GestorInfoEstudiantes ();
        gestorPrestamosEjemplares = new GestorPrestarEjemplar
(pantallaPrincipal);

        // Limpia Gestor que almacena la informacion de un prestamo

        // Captura la cedula del formattedTextField y la almacena en
la estructura
        // temporal gestorEstudiantes

gestorEstudiantes.setCedula(Long.parseLong(jftfCedula.getText().trim()
));
        gestorEstudiantes =
gestorPrestamosEjemplares.buscarEstudiante(gestorEstudiantes,fotoTemp)
;

        if (gestorEstudiantes.getCedula()!= (long)0){
            if (gestorEstudiantes.getCedula()!=1){
                jftfCedula.setBackground(Color.WHITE);
                jtfNombres.setText(gestorEstudiantes.getNombre());
                jtfApellidos.setText(gestorEstudiantes.getApellido());

jtfCorreoElectronico.setText(gestorEstudiantes.getCorreoElectronico())
;
                jtfTelefono.setText(gestorEstudiantes.getTelefono());
                if (gestorEstudiantes.getCOMPUTACION()){
                   jlCarrera.setText("Computacion");
                }
                if (gestorEstudiantes.getSISTEMAS()){
                   jlCarrera.setText("Sistemas");
                }
                if
(gestorEstudiantes.getSISTEMAS()&&gestorEstudiantes.getCOMPUTACION()){
                   jlCarrera.setText("Computacion y sistemas.");
                }
```

```
                if (gestorEstudiantes.getFotoEstudiante()!=null){

fotoTemp=gestorEstudiantes.getFotoEstudiante();
                        labelFoto.setIcon(new
javax.swing.ImageIcon(gestorEstudiantes.getFotoEstudiante()));
                       panelCamara.add(labelFoto);
                        labelFoto.setVisible(true);
                        labelFoto.updateUI();
                } // end if
                 else {
                    labelFoto.setIcon(new
javax.swing.ImageIcon("src/gui/resources/sinFotoCam.png"));
                    panelCamara.add(labelFoto);
                    labelFoto.setVisible(true);
                    labelFoto.updateUI();
                 }

                if (gestorEstudiantes.getACTIVO()){
                    jlActivo.setForeground(Azul);
                    jlActivo.setText("INSCRITO");
                }
                else{
                    jlActivo.setForeground(Color.DARK_GRAY);
                    jlActivo.setText("NO INSCRITO");
                    pantallaPrincipal.errorBarraEstado("El estudiante
no se encuentra inscrito y no puede retirar nigun ejemplar de la
sala");
                    return;
                }

                if (gestorEstudiantes.getSOLVENTE()){
                    jlSolvente.setForeground(Verde);
                    jlSolvente.setText("SOLVENTE");
                }
                else {
                    jlSolvente.setForeground(Rojo);
                    jlSolvente.setText("NO SOLVENTE");
                    pantallaPrincipal.errorBarraEstado("El estudiante
no esta solvente, no puede retirar nignun ejemplar de la sala.");
                    return;
                }

gestorInfoPrestamo.setCeduladEstudiante(String.valueOf(gestorEstudiant
es.getCedula()));
//                prestamoActivo=true;
                activarPrestamo ();
            } // end if
            else {
                pantallaPrincipal.advertenciaBarraEstado("El
estudiante con la cédula "+jftfCedula.getText().trim()+" no existe en
la base de datos.");
                limpiarFormularioEstudiante();
                jftfCedula.setBackground(Color.pink);
            } // end else
        } // end if
    }

    // Activa el JTextField jtfCota y el JComboBox jcbPrestamo
    // para que se pueda cargar un ejemplar al estudiante
    private void activarPrestamo (){
        botonConsultar.setEnabled(false);
```

```
        // Activa textField y comboBox
        jtfCota.setEnabled(true);
        jcbPrestamo.setEnabled(true);
        botonAñadir.setEnabled(true);
        // Obtiene una instancia de la fecha, la almacena en el
gestorInfoPRestamo
        // y la dezpliega en el la pantalla
        Calendar fecha = Calendar.getInstance();
        String diaNumero =
String.valueOf(fecha.get(Calendar.DAY_OF_MONTH));
        String diaLetras =
convertirDia(fecha.get(Calendar.DAY_OF_WEEK));
        String mes = String.valueOf(fecha.get(Calendar.MONTH)+1);
        String año = String.valueOf(fecha.get(Calendar.YEAR));
        jlFecha.setForeground(Azul);
        jlFecha.setText(diaLetras+" "+diaNumero+"/"+mes+"/"+año);
        gestorInfoPrestamo.setFecha(año+"-"+mes+"-"+diaNumero);

        // Obtiene otra intancia de la fecha y la compara con el dia
        // y con el tipo de prestamo determinanado que dia tiene que
devolver
        // el ejemplar
        if (jcbPrestamo.getSelectedIndex()!=0){

fecha.add(Calendar.DATE,gestorVariableSistema.getDuracionDeUnPrestamo(
));
            if (fecha.get(Calendar.DAY_OF_WEEK)==Calendar.SUNDAY){
                fecha.add(Calendar.DATE,1);
            }
            else
                if
((fecha.get(Calendar.DAY_OF_WEEK)==Calendar.SATURDAY))
                    fecha.add(Calendar.DATE,2);
        diaNumero = String.valueOf(fecha.get(Calendar.DAY_OF_MONTH));
        diaLetras = convertirDia(fecha.get(Calendar.DAY_OF_WEEK));
        mes = String.valueOf(fecha.get(Calendar.MONTH)+1);
        año = String.valueOf(fecha.get(Calendar.YEAR));
        } // end if
        jlFechaEntrega.setForeground(Verde);
        jlFechaEntrega.setText(diaLetras+"
"+diaNumero+"/"+mes+"/"+año);
        gestorInfoPrestamo.setFechaEntrega(año+"-"+mes+"-"+diaNumero);
        verificarCantidadLibrosDisponibles ();
    }


    // Metodo que verifica la cantidad de libros que tiene prestado
    // el estudiante en ese momento y ademas usando el valor de la
    // cantidad maxima de ejemplares que se puede prestar (variables
    // de las opciones del sistema) determina cuantos libros se le
    // pueden prestar al estudiante
    private void verificarCantidadLibrosDisponibles (){
        int librosPrestados =
gestorPrestamosEjemplares.librosPrestadosAlEstudiante(gestorInfoPresta
mo.getCedulaEstudiante());
        jlLibrosPrestados.setText(librosPrestados+" libro(s)");
        if
(librosPrestados>=gestorVariableSistema.getMaxEjemplaresPorPersona()){
            pantallaPrincipal.advertenciaBarraEstado("El estudiante ha
alcanzado el máximo número de ejemplares que puede retirar de la
sala.");
```

```
            jtfCota.setEnabled(false);
            jcbPrestamo.setEnabled(false);
            botonAñadir.setEnabled(false);
            return;
        }
        cantidadLibrosPrestar =
gestorVariableSistema.getMaxEjemplaresPorPersona() - librosPrestados;
        jlLibrosRestantes.setText(cantidadLibrosPrestar+" libro(s).");
    }

    // Revisa una cota en la base de datos y verifica que este
disponibles y
    // que cumpla con los requisitos para ser prestada dependiendo del
tipo
    // de prestamo que seleccione el usuario
    private void cargarEjemplar(){
        // Limpia la pantalla y colorea de blanco el TextField de la
cota
        pantallaPrincipal.borrarBarraEstado();
        jtfCota.setBackground(Color.white);
        if (cantidadLibrosPrestar<=0){
            pantallaPrincipal.advertenciaBarraEstado("Se ha alcanzado
el limite de libros que se puede prestar a un estudiante.");
            return;
        }

        String cota = jtfCota.getText().trim();
        if (cota.trim().isEmpty()){
            pantallaPrincipal.errorBarraEstado("La cota debe contener
al menos un caracter.");
            return;
        }
        gestorPrestamosEjemplares = new
GestorPrestarEjemplar(pantallaPrincipal);
        gestorEjemplar = new GestorInfoEjemplar();

        gestorEjemplar.setIdEjemplar(cota);
        gestorEjemplar
=gestorPrestamosEjemplares.buscarPorCota(gestorEjemplar);

        // Verifica si se consiguio el ejemplar
        if (gestorEjemplar==null){
            jtfCota.setBackground(Color.PINK);
            return;
        }

        // Verifica si el ejemplar esta disponible
        if (!gestorEjemplar.getDISPONIBLE()){
            jtfCota.setBackground(Color.PINK);
            pantallaPrincipal.advertenciaBarraEstado("El ejemplar
"+gestorEjemplar.getIdEjemplar()+" no se encuentra disponible.");
            return;
        }

        // Verifica si el libro y el modo de prestamo seleccionado por
el
        // usuario son compatibles

        // Verifica si desea presta una tesis para prestamo externo
        if ( (!gestorEjemplar.getLIBRO()) &&
jcbPrestamo.getSelectedIndex()==1 ){
```

```
            pantallaPrincipal.advertenciaBarraEstado("Para el prestamo
externo de una tesis, consulte con el profesor encargado.");
            if (!gestorVariableSistema.getTesisPrestamoExterno()){
                JOptionPane.showMessageDialog(this, "    ERROR: las
tesis son solo para prestamo interno.  \n" +
                                                    "    Si  desea
activar el prestamo externo  de  las   \n" +
                                                    "    mismas,
revise  la  opciones  del  programa  y   \n" +
                                                    "    consulte con
el profesor encargado de la sala.  ","Prestamo externo de tesis
desactivado",JOptionPane.WARNING_MESSAGE);
                return;
            }// end if
        } // end if
        else {
            // Verifica si se desea presta un libro Externo para un
prestamo interno
            if (gestorEjemplar.getINTERNO() &&
jcbPrestamo.getSelectedIndex()==1){
                pantallaPrincipal.advertenciaBarraEstado("El libro que
selecciono es solo para prestamo interno.");
                return;
            }
            else{
                // Verifica si se desea prestar un libre externo para
un prestamo
                // interno
                if ( (!gestorEjemplar.getINTERNO()) &&
jcbPrestamo.getSelectedIndex()==0 ){
                    pantallaPrincipal.advertenciaBarraEstado("El libro
que selecciono es solo para prestamo externo.");
                    return;
                }
            }
        }
        for (int
i=0;i<jtListaEjemplares.getModel().getRowCount();i++){
            if
(jtListaEjemplares.getValueAt(i,1).toString().trim().equals(gestorEjem
plar.getIdEjemplar())){
                pantallaPrincipal.advertenciaBarraEstado("La cota
"+gestorEjemplar.getIdEjemplar()+" ya fue cargada.");
                jtfCota.setBackground(Color.PINK);
                return;
            }

        }
        if (jcbPrestamo.getSelectedIndex()==0)
            gestorInfoPrestamo.setPrestamoINTERNO(true);
        else
            gestorInfoPrestamo.setPrestamoINTERNO(false);

        // OJO POR TERMINAR
        // OJO POR TERMINAR
        // OJO POR TERMINAR
        // OJO POR TERMINAR
        gestorInfoPrestamo.setIdAyudante("00000000"); //
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
        // OJO POR TERMINAR
        // OJO POR TERMINAR
```

```
        // OJO POR TERMINAR
        // OJO POR TERMINAR
        jcbPrestamo.setEnabled(false);
        jtfCota.setText("");
        añadirInfoTabla (gestorEjemplar);
        --cantidadLibrosPrestar;
        jlLibrosRestantes.setText(cantidadLibrosPrestar+" libro(s).");

    }

    // Añade la informacion de un ejemplar dado por el objeto
getorEjemplar
    // a la tabla que muestra la lista de los ejemplares a prestar
    private void  añadirInfoTabla (GestorInfoEjemplar
gestorEjemplares){
        DefaultTableModel tablaTemporal = (DefaultTableModel)
jtListaEjemplares.getModel();
        String interno;

        String libro;
        if (gestorEjemplar.getINTERNO())
                interno="INTERNO";
            else
                interno= "CIRCULANTE";

            if (gestorEjemplar.getLIBRO())
                libro = "LIBRO";
            else
                libro= "TESIS";

            Object[] nuevo = {gestorEjemplar.getNombreEjemplar(),
                              gestorEjemplar.getIdEjemplar(),
                              gestorEjemplar.getArea(),
                              libro,
                              gestorEjemplar.getEdicion(),
                              interno};
             tablaTemporal.addRow(nuevo);
    }

    private void prestarEjemplares (){
        String cotas[] = new
String[gestorVariableSistema.getMaxEjemplaresPorPersona()];
        int i;
        for (i=0;i<jtListaEjemplares.getRowCount();i++){

cotas[i]=jtListaEjemplares.getValueAt(i,1).toString().trim();
        }



gestorPrestamosEjemplares.guardarPrestamo(gestorInfoPrestamo,cotas,i);
        borrarTodo();

    }


    // Metodo que borra y reinicia todos los componentes
    // necesarios en la pantalla
    private void borrarTodo(){
        jftfCedula.setValue("");
        limpiarFormularioEstudiante();
```

```
        limpiarComponentesPrestamo ();
        limpiarTabla ();
    }

    private void borrarPrestamos (){
       limpiarTabla();
       verificarCantidadLibrosDisponibles ();
       jtfCota.setEnabled(true);
       jcbPrestamo.setEnabled(true);
    }

    // Metodo que limpia los valores en los formularios y los deja en
el
    // estado activo en el que se entoncontraban cuando se inicio la
    // la interfaz grafica
    private void limpiarFormularioEstudiante(){
        jtfNombres.setText("");
        jtfApellidos.setText("");
        jtfCorreoElectronico.setText("");
        jtfTelefono.setText("");
        jlCarrera.setText(" ");
        jlActivo.setText(" ");
        jlSolvente.setText(" ");

        labelFoto.setIcon(new ImageIcon
(GUIView.class.getResource("resources/sinFotoCam.png")));
        panelCamara.add(labelFoto);
        labelFoto.setVisible(true);
        labelFoto.updateUI();
        jlLibrosPrestados.setText("");
        jlLibrosRestantes.setText("");
        botonConsultar.setEnabled(true);
        limpiarComponentesPrestamo ();
    }

    // Metodo para limpiar volver a iniciar los
    // valores de la zona de prestamo
    private void limpiarComponentesPrestamo (){
        // Componentes del prestamo
        botonAñadir.setEnabled(false);
        jtfCota.setEnabled(false);
        jcbPrestamo.setEnabled(false);
        jlFecha.setText("");
        jlFechaEntrega.setText("");
    }


    // Limpia la tabla eliminando todas la filas cargadas.
    private void limpiarTabla (){
        int columnas =
((DefaultTableModel)jtListaEjemplares.getModel()).getRowCount();
        if (columnas >0){
            for (int i=0;i<columnas;i++)

((DefaultTableModel)jtListaEjemplares.getModel()).removeRow(0);
        }
    }

    // Convierte un dia en formato numero en un String en español
    // p/ejemplo- int = 0 String =Lunes;
    private String convertirDia (int diaNum){
```

```
        String dia;
        switch (diaNum) {
            case Calendar.SUNDAY:
                dia= "Domingo";
                break;
            case Calendar.MONDAY:
                dia= "Lunes";
                break;
            case Calendar.TUESDAY:
                dia= "Martes";
                break;
            case Calendar.WEDNESDAY:
                dia= "Miercoles";
                break;
            case Calendar.THURSDAY:
                dia= "Jueves";
                break;
            case Calendar.FRIDAY:
                dia= "Viernes";
                break;
            case Calendar.SATURDAY:
                dia= "Sabado";
                break;
            default:
                dia=String.valueOf(diaNum);
                break;
        }
        return dia;
    }

     // Metodo para ajustar el tamaño de la tabla que lista los
estudiantes
    private void ajustarAnchoTabla (){
        for (int i = 0; i < 6; i++) {
            column = jtListaEjemplares.getColumnModel().getColumn(i);
            if(i == 0)
                column.setPreferredWidth(338);
            else
                if (i==1)
                            column.setPreferredWidth(70);
                else
                    if(i == 2)
                        column.setPreferredWidth(126);
                    else
                        if(i == 3)
                            column.setPreferredWidth(49);

                            if (i==4)
                                column.setPreferredWidth(51);
                            else
                                if (i==5)
                                column.setPreferredWidth(119);
                                else
                                    if (i==6)
                                        column.setPreferredWidth(119);
        }
    }
}
```

5.3.2.2.2 GestorPrestarEjemplar

```
package GestionPrestamos;

import BaseDeDatos.ConexionBD;
import GestionEjemplares.GestorInfoEjemplar;
import GestionEstudiantes.GestorInfoEstudiantes;
import Principal.GUIView;
import java.io.IOException;
import java.sql.ResultSet;
import java.awt.image.BufferedImage;
import java.io.ByteArrayInputStream;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.util.regex.Pattern;
import javax.imageio.ImageIO;


// Autores: Oscar Saavedra y Sair Amer
//      Universidad de Oriente
//    Barcelona - Venezuela - 2009

public class GestorPrestarEjemplar {
    private GUIView pantallaPrincipal;
    private ConexionBD conexionBD;
    private ResultSet rs = null;


    GestorPrestarEjemplar(GUIView pantallaPrincipal) {
```

```
        this.pantallaPrincipal = pantallaPrincipal;
    }


    // Guarda los datos de un prestamo en la base de datos
    // recibe un GestorInfoPrestamo con la informacion
    // el prestamo, un String[] con las cotas a cargar y un
    // int que determina la cantidad de libros que se van a cargar
    public void guardarPrestamo(GestorInfoPrestamo
gestorInfoPrestamo,String[] cota,int maxCotas){
        // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        int i;
        for (i=0;i<maxCotas;i++)
            conexionBD.Actualizar("UPDATE ejemplar SET
disponible='false' WHERE idejemplar='"+cota[i]+"'");

        for (i=0;i<maxCotas;i++){
            gestorInfoPrestamo.setIdEjemplar(cota[i]);

conexionBD.CargarDatos("prestamo(nrocedulaestudiante,fecha,idejemplar,
idayudante,interno,fechaentrega,recibido)",
                                    "
'"+gestorInfoPrestamo.getCedulaEstudiante()+"',"+
                                    "
'"+gestorInfoPrestamo.getFecha()+"',"+
                                    "
'"+gestorInfoPrestamo.getIdEjemplar()+"',"+
                                    "
'"+gestorInfoPrestamo.getIdAyudante()+"',"+
                                    "
'"+gestorInfoPrestamo.getPrestamoINTERNO()+"',"+
                                    "
'"+gestorInfoPrestamo.getFechaEntrega()+"',"+
                                    " 'false'");
        }
        conexionBD.CerrarConexion();
    }


    // Metodo que busca un ejemplar en las tables ejemplar y
datosejemplares
    // usando la cota del mismo
    // @ SI consigue la cota retorna el objeto getorEjemplar con la
informacion
    // del ejemplar
    // @ Si se produce un error o no se consigue le ejemplar retorna
null
    public GestorInfoEjemplar buscarPorCota(GestorInfoEjemplar
gestorEjemplar) {
        // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
```

```
        conexionBD.Consulta("SELECT * FROM datosejemplares, ejemplar
WHERE datosejemplares.iddatos = ejemplar.iddatos AND
ejemplar.idejemplar = '"+gestorEjemplar.getIdEjemplar()+"'");
        rs = conexionBD.MostrarDatosConsulta();

        try {
            if (rs.next()!=false){

gestorEjemplar.setNombreEjemplar(rs.getString("nombre"));
                gestorEjemplar.setArea(rs.getString("area"));
                gestorEjemplar.setLibro(rs.getBoolean("libro"));
                gestorEjemplar.setEdicion(rs.getString("edicion"));
                gestorEjemplar.setInterno(rs.getBoolean("interno"));

gestorEjemplar.setDisonible(rs.getBoolean("disponible"));
                gestorEjemplar.setAño(rs.getInt("anio"));
                gestorEjemplar.setResumen(rs.getString("resumen"));

                String autores = rs.getString("autores");
                String REGEX = "!!!";
                Pattern p = Pattern.compile(REGEX);
                String a[] = p.split(autores) ;
                gestorEjemplar.setAutores(a[0].trim(), a[1].trim(),
a[2].trim());
                conexionBD.CerrarConexion();
                return gestorEjemplar;
            }
            else{
                conexionBD.CerrarConexion();
                pantallaPrincipal.advertenciaBarraEstado("La cota #
"+gestorEjemplar.getIdEjemplar()+" no se consigue en el sistema.");
                return null;
            }
        } catch ( SQLException ex) {
            conexionBD.CerrarConexion();
            pantallaPrincipal.errorBarraEstado("Error al buscar en la
base de datos.");

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            return null;
        }
    }



    // Consulta la cedula en la base de datos, si la cedula exite
devuelve
    // toda la informacion del estudiante a travez del objeto
gestorEstudiante
    // @ Si hay un error durante la lectura el objeto gestorEstudiante
se
    // devuelve con una cedula cuyo valor es 0
    // @ Si no se consigue la cedula que se solicito se devuelve el
objeto
    // gestorEstudiante con un valor de 1 en la cedula.
    public GestorInfoEstudiantes
buscarEstudiante(GestorInfoEstudiantes gestorEstudiante, BufferedImage
fotoTemp) {
        // Limpia la barra de estado de la pantalla principal
        pantallaPrincipal.borrarBarraEstado();
```

```
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();

        if (conexionBD.Consulta("SELECT * FROM estudiantes WHERE
numerocedula = '"+gestorEstudiante.getCedula()+"'")){
            rs = conexionBD.MostrarDatosConsulta();
                try {
                    rs.next();
                    if (rs.getRow()!=0) {

gestorEstudiante.setNombre(rs.getString("nombres"));

gestorEstudiante.setApellido(rs.getString("apellidos"));

gestorEstudiante.setCorreoElectronico(rs.getString("email"));

gestorEstudiante.setACTIVO(rs.getBoolean("activo"));

gestorEstudiante.setSOLVETE(rs.getBoolean("solvente"));

gestorEstudiante.setCOMPUTACION(rs.getBoolean("computacion"));

gestorEstudiante.setSISTEMAS(rs.getBoolean("sistemas"));
                        byte[] imgBytes =
conexionBD.retornarFotoEstudiante(gestorEstudiante.getCedula());
                        BufferedImage image = null;
                        if (imgBytes!=null)
                             image = ArrayBufferedImage(imgBytes);
                        gestorEstudiante.setFotoEstudiante(image);

gestorEstudiante.setTelefono(rs.getString("numerodetelefono"));
                    }
                    else {
                        conexionBD.CerrarConexion();
                        gestorEstudiante.setCedula((long)1);
                        return gestorEstudiante;
                    }
                    }   catch (SQLException e) {
                            conexionBD.CerrarConexion();
                            System.out.println("Error "+e);
                            pantallaPrincipal.errorBarraEstado("Error
al acceder a la base de datos");
                            gestorEstudiante.setCedula((long)0);
                            return gestorEstudiante;
                    }
                conexionBD.CerrarConexion();
                return gestorEstudiante;
        } // end if
        else {
            conexionBD.CerrarConexion();
            gestorEstudiante.setCedula((long)1);
            return gestorEstudiante;
        } // end else
    }


    //  Metodo que revisa toda la tabla prestamo buscando las
coincidencias
    // con la cedula de un estudiante para determinar cuantos libros
```

```
tiene
    // en su poder un estudiante.
    public int librosPrestadosAlEstudiante (String cedula){
        try {
            // Limpia la barra de estado de la pantalla principal
            pantallaPrincipal.borrarBarraEstado();
            // Inicia y abre la conexion con la base de datos
            conexionBD = new ConexionBD(pantallaPrincipal);
            conexionBD.AbrirConexion();
            conexionBD.Consulta("SELECT COUNT(*) FROM prestamo WHERE
nrocedulaestudiante ="+cedula+" AND recibido='false'");
            rs = conexionBD.MostrarDatosConsulta();
            rs.next();
            conexionBD.CerrarConexion();
            return rs.getInt("COUNT");

        } catch (SQLException ex) {

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            pantallaPrincipal.errorBarraEstado("Error al acceder a la
base de datos.");
            conexionBD.CerrarConexion();
            return -1;
        }
    }

    // Metodo que tranforma un array de byte en un BufferedImage y lo
retorna
    // @ Recibe: byte[] imageData
    // @ Retorna: BufferedImage
    public BufferedImage ArrayBufferedImage (byte[] imageData ) {
        try {
            ByteArrayInputStream in = new
ByteArrayInputStream(imageData);
            return ImageIO.read(in);
        } catch (IOException ex) {

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            return null;
        }
    }

}
```

5.3.2.2.3 GUIRecibirEjemplar

```
package GestionPrestamos;

import GestionEjemplares.GestorInfoEjemplar;
import GestionEstudiantes.GestorInfoEstudiantes;
import Principal.GUIView;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.event.KeyEvent;
import java.text.DecimalFormat;
```

```
import java.text.ParseException;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.ImageIcon;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableColumn;
import procesamiento.CellRenderPersonalizado;

public class GUIrecibirEjemplar extends javax.swing.JPanel {
    private GUIView pantallaPrincipal;
    private Color Azul;
    private Color Verde;
    private Color Rojo;
    private TableColumn column;
    private String path;
    private ImageIcon imagenFondo;


    public GUIrecibirEjemplar(GUIView pantallaPrincipal) {
        initComponents();
        this.pantallaPrincipal = pantallaPrincipal;

jtListaEjemplares.setDefaultRenderer(jtListaEjemplares.getColumnClass(
0), new CellRenderPersonalizado(4));
        gestorVariablesSistema = new
GestorVariablesSistema(pantallaPrincipal);
        listaGestorInfoPrestamo = new ArrayList();

        jlMoneda.setText(gestorVariablesSistema.getMoneda());
        jlMoneda2.setText(gestorVariablesSistema.getMoneda());
        // Define colores a usar
        Azul = new Color(51,94,168);
        Verde = new Color (51,153,0);
        Rojo = new Color (255, 51, 51);
        ajustarAnchoTabla ();
         cargarIconos();
    }

    private void cargarIconos() {
        imagenFondo = new ImageIcon
(GUIView.class.getResource("resources/fondo.png"));
    }

    // Metodo para dibujar una imagen de fondo en el jPanel del
formulario
    // que se esta desplegando
    @Override
    public void paintComponent(Graphics g)
     {
        Graphics2D g2 = (Graphics2D)g;

g2.drawImage(imagenFondo.getImage(),0,0,this.getWidth(),this.getHeight
(),this);

    }
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {
```

```
        jDialog1 = new javax.swing.JDialog(){

            public void paintComponent(Graphics g)
            {
                Graphics2D g2 = (Graphics2D)g;
                g2.drawImage(new
ImageIcon("src\\gui\\resources\\fondoPago.png").getImage(),0,0,this.ge
tWidth(),this.getHeight(),this);

            }
        }
        ;
        jPanel1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        jLabel2 = new javax.swing.JLabel();
        jtfTotal = new javax.swing.JFormattedTextField();
        jLabel3 = new javax.swing.JLabel();
        jftfMontoCancelado = new javax.swing.JFormattedTextField();
        jSeparator1 = new javax.swing.JSeparator();
        jPanel10 = new javax.swing.JPanel();
        jlDiferencia = new javax.swing.JLabel();
        jtfDiferencia = new javax.swing.JFormattedTextField();
        botonCobrar = new javax.swing.JButton();
        jlError = new javax.swing.JLabel();
        jPanel3 = new javax.swing.JPanel();
        jPanel4 = new javax.swing.JPanel();
        jLabel19 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel20 = new javax.swing.JLabel();
        jLabel21 = new javax.swing.JLabel();
        jPanel5 = new javax.swing.JPanel();
        jtfNombres = new javax.swing.JTextField();
        jtfApellidos = new javax.swing.JTextField();
        jlCarrera = new javax.swing.JLabel();
        jftfCedula = new javax.swing.JFormattedTextField();
        jPanel6 = new javax.swing.JPanel();
        jLabel28 = new javax.swing.JLabel();
        jLabel29 = new javax.swing.JLabel();
        jLabel30 = new javax.swing.JLabel();
        jPanel7 = new javax.swing.JPanel();
        jlActivo = new javax.swing.JLabel();
        jtfTelefono = new javax.swing.JTextField();
        jtfCorreoElectronico = new javax.swing.JTextField();
        jPanel8 = new javax.swing.JPanel();
        jtfCota = new javax.swing.JTextField();
        botonBuscar = new javax.swing.JButton();
        jPanel9 = new javax.swing.JPanel();
        jScrollPane2 = new javax.swing.JScrollPane();
        jtListaEjemplares = new javax.swing.JTable();
        jPanel11 = new javax.swing.JPanel();
        jLabel10 = new javax.swing.JLabel();
        jlFechaSalida = new javax.swing.JLabel();
        jLabel12 = new javax.swing.JLabel();
        jlFechaEntrega = new javax.swing.JLabel();
        jLabel13 = new javax.swing.JLabel();
        jlLibrosPrestados = new javax.swing.JLabel();
        jPanel12 = new javax.swing.JPanel();
        jLabel14 = new javax.swing.JLabel();
        jlFecha = new javax.swing.JLabel();
        jLabel15 = new javax.swing.JLabel();
        jlRetraso = new javax.swing.JLabel();
```

```
        jLabel16 = new javax.swing.JLabel();
        jlDeuda = new javax.swing.JLabel();
        jlMoneda2 = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        jlDeudaTotal = new javax.swing.JLabel();
        jlMoneda = new javax.swing.JLabel();
        jbBotonLimpiar = new javax.swing.JButton();
        botonRecibirEjemplares = new javax.swing.JButton();
        botonCancelar = new javax.swing.JButton();

        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(Principal.GUIApp.clas
s).getContext().getResourceMap(GUIrecibirEjemplar.class);
        jDialog1.setTitle(resourceMap.getString("jDialog1.title")); //
NOI18N
        jDialog1.setModal(true);
        jDialog1.setName("jDialog1"); // NOI18N
        jDialog1.setResizable(false);

        jPanel1.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel1.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel1.border.titleFont"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        jPanel2.setName("jPanel2"); // NOI18N
        jPanel2.setLayout(new java.awt.GridLayout(2, 2, 5, 5));

        jLabel2.setFont(resourceMap.getFont("jLabel2.font")); //
NOI18N
        jLabel2.setText(resourceMap.getString("jLabel2.text")); //
NOI18N
        jLabel2.setName("jLabel2"); // NOI18N
        jPanel2.add(jLabel2);

        jtfTotal.setEditable(false);
        jtfTotal.setFormatterFactory(new
javax.swing.text.DefaultFormatterFactory(new
javax.swing.text.NumberFormatter(new
java.text.DecimalFormat("#0.00"))));
        jtfTotal.setHorizontalAlignment(javax.swing.JTextField.RIGHT);

jtfTotal.setFocusLostBehavior(javax.swing.JFormattedTextField.PERSIST)
;
        jtfTotal.setFont(resourceMap.getFont("jtfTotal.font")); //
NOI18N
        jtfTotal.setName("jtfTotal"); // NOI18N
        jtfTotal.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                jtfTotalKeyTyped(evt);
            }
        });
        jPanel2.add(jtfTotal);

        jLabel3.setFont(resourceMap.getFont("jLabel2.font")); //
NOI18N
        jLabel3.setText(resourceMap.getString("jLabel3.text")); //
```

```
NOI18N
        jLabel3.setName("jLabel3"); // NOI18N
        jPanel2.add(jLabel3);

        jftfMontoCancelado.setFormatterFactory(new
javax.swing.text.DefaultFormatterFactory(new
javax.swing.text.NumberFormatter(new
java.text.DecimalFormat("#0.00"))));

jftfMontoCancelado.setHorizontalAlignment(javax.swing.JTextField.RIGHT
);

jftfMontoCancelado.setFocusLostBehavior(javax.swing.JFormattedTextFiel
d.PERSIST);

jftfMontoCancelado.setFont(resourceMap.getFont("jtfTotal.font")); //
NOI18N
        jftfMontoCancelado.setName("jftfMontoCancelado"); // NOI18N
        jftfMontoCancelado.addKeyListener(new
java.awt.event.KeyAdapter() {
            public void keyPressed(java.awt.event.KeyEvent evt) {
                jftfMontoCanceladoKeyPressed(evt);
            }
            public void keyTyped(java.awt.event.KeyEvent evt) {
                jftfMontoCanceladoKeyTyped(evt);
            }
        });
        jPanel2.add(jftfMontoCancelado);

        jSeparator1.setName("jSeparator1"); // NOI18N

        jPanel10.setName("jPanel10"); // NOI18N
        jPanel10.setLayout(new java.awt.GridLayout(1, 2, 5, 0));


jlDiferencia.setFont(resourceMap.getFont("jlDiferencia.font")); //
NOI18N

jlDiferencia.setText(resourceMap.getString("jlDiferencia.text")); //
NOI18N
        jlDiferencia.setName("jlDiferencia"); // NOI18N
        jPanel10.add(jlDiferencia);

        jtfDiferencia.setEditable(false);
        jtfDiferencia.setFormatterFactory(new
javax.swing.text.DefaultFormatterFactory(new
javax.swing.text.NumberFormatter(new
java.text.DecimalFormat("#0.00"))));

jtfDiferencia.setHorizontalAlignment(javax.swing.JTextField.RIGHT);

jtfDiferencia.setFocusLostBehavior(javax.swing.JFormattedTextField.PER
SIST);
        jtfDiferencia.setFont(resourceMap.getFont("jtfTotal.font"));
// NOI18N
        jtfDiferencia.setName("jtfDiferencia"); // NOI18N
        jtfDiferencia.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                jtfDiferenciaKeyTyped(evt);
            }
        });
```

```
        jPanel10.add(jtfDiferencia);

        botonCobrar.setFont(resourceMap.getFont("botonCobrar.font"));
// NOI18N
        botonCobrar.setIcon(resourceMap.getIcon("botonCobrar.icon"));
// NOI18N

botonCobrar.setText(resourceMap.getString("botonCobrar.text")); //
NOI18N
        botonCobrar.setEnabled(false);
        botonCobrar.setName("botonCobrar"); // NOI18N
        botonCobrar.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                botonCobrarActionPerformed(evt);
            }
        });

        jlError.setText(resourceMap.getString("jlError.text")); //
NOI18N
        jlError.setName("jlError"); // NOI18N

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jPanel10,
javax.swing.GroupLayout.DEFAULT_SIZE, 286, Short.MAX_VALUE)
                    .addComponent(jSeparator1,
javax.swing.GroupLayout.DEFAULT_SIZE, 286, Short.MAX_VALUE)
                    .addComponent(jPanel2,
javax.swing.GroupLayout.DEFAULT_SIZE, 286, Short.MAX_VALUE)
                    .addComponent(jlError,
javax.swing.GroupLayout.DEFAULT_SIZE, 286, Short.MAX_VALUE)
                    .addComponent(botonCobrar,
javax.swing.GroupLayout.DEFAULT_SIZE, 286, Short.MAX_VALUE))
                .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jSeparator1,
javax.swing.GroupLayout.PREFERRED_SIZE, 10,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
52, Short.MAX_VALUE)
                .addComponent(botonCobrar,
javax.swing.GroupLayout.PREFERRED_SIZE, 34,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jlError,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE))
        );

        javax.swing.GroupLayout jDialog1Layout = new
javax.swing.GroupLayout(jDialog1.getContentPane());
        jDialog1.getContentPane().setLayout(jDialog1Layout);
        jDialog1Layout.setHorizontalGroup(

jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        jDialog1Layout.setVerticalGroup(

jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );

        setBorder(javax.swing.BorderFactory.createTitledBorder(null,
resourceMap.getString("Form.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("Form.border.titleFont"))); // NOI18N
        setName("Form"); // NOI18N
        setPreferredSize(new java.awt.Dimension(776, 909));

        jPanel3.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel3.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel3.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel3.border.titleFont"))); // NOI18N
        jPanel3.setMaximumSize(new java.awt.Dimension(500, 147));
        jPanel3.setMinimumSize(new java.awt.Dimension(500, 147));
        jPanel3.setName("jPanel3"); // NOI18N
        jPanel3.setOpaque(false);
```

```
        jPanel4.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel4.setName("jPanel4"); // NOI18N
        jPanel4.setOpaque(false);

        jLabel19.setFont(resourceMap.getFont("jLabel19.font")); //
NOI18N
        jLabel19.setText(resourceMap.getString("jLabel19.text")); //
NOI18N
        jLabel19.setName("jLabel19"); // NOI18N

        jLabel5.setFont(resourceMap.getFont("jLabel19.font")); //
NOI18N
        jLabel5.setText(resourceMap.getString("jLabel5.text")); //
NOI18N
        jLabel5.setName("jLabel5"); // NOI18N

        jLabel20.setFont(resourceMap.getFont("jLabel19.font")); //
NOI18N
        jLabel20.setText(resourceMap.getString("jLabel20.text")); //
NOI18N
        jLabel20.setName("jLabel20"); // NOI18N

        jLabel21.setFont(resourceMap.getFont("jLabel19.font")); //
NOI18N
        jLabel21.setText(resourceMap.getString("jLabel21.text")); //
NOI18N
        jLabel21.setName("jLabel21"); // NOI18N

        javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4);
        jPanel4.setLayout(jPanel4Layout);
        jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel4Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING)
                    .addComponent(jLabel21,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 122, Short.MAX_VALUE)
                    .addComponent(jLabel5,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 122, Short.MAX_VALUE)
                    .addComponent(jLabel20,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 122, Short.MAX_VALUE)
                    .addComponent(jLabel19,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addGap(16, 16, 16))
        );
        jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel4Layout.createSequentialGroup()
                .addContainerGap()
```

```
                .addComponent(jLabel19,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel5,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel20,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(11, 11, 11)
                .addComponent(jLabel21,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        jPanel5.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel5.setName("jPanel5"); // NOI18N
        jPanel5.setOpaque(false);

        jtfNombres.setEditable(false);

jtfNombres.setToolTipText(resourceMap.getString("jtfNombres.toolTipTex
t")); // NOI18N
        jtfNombres.setName("jtfNombres"); // NOI18N

        jtfApellidos.setEditable(false);

jtfApellidos.setToolTipText(resourceMap.getString("jtfApellidos.toolTi
pText")); // NOI18N
        jtfApellidos.setName("jtfApellidos"); // NOI18N

        jlCarrera.setFont(resourceMap.getFont("jlActivo.font")); //
NOI18N

jlCarrera.setToolTipText(resourceMap.getString("jlCarrera.toolTipText"
)); // NOI18N

jlCarrera.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER)
;
        jlCarrera.setName("jlCarrera"); // NOI18N

        jftfCedula.setEditable(false);
        jftfCedula.setText(resourceMap.getString("jftfCedula.text"));
// NOI18N

jftfCedula.setToolTipText(resourceMap.getString("jftfCedula.toolTipTex
t")); // NOI18N

jftfCedula.setFocusLostBehavior(javax.swing.JFormattedTextField.PERSIS
T);
        jftfCedula.setName("jftfCedula"); // NOI18N

        javax.swing.GroupLayout jPanel5Layout = new
javax.swing.GroupLayout(jPanel5);
        jPanel5.setLayout(jPanel5Layout);
```

```
        jPanel5Layout.setHorizontalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jtfApellidos,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 171, Short.MAX_VALUE)
                    .addComponent(jtfNombres,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 171, Short.MAX_VALUE)
                    .addGroup(jPanel5Layout.createSequentialGroup()
                        .addComponent(jftfCedula,
javax.swing.GroupLayout.DEFAULT_SIZE, 82, Short.MAX_VALUE)
                        .addGap(89, 89, 89))
                    .addComponent(jlCarrera,
javax.swing.GroupLayout.DEFAULT_SIZE, 171, Short.MAX_VALUE))
                .addContainerGap())
        );
        jPanel5Layout.setVerticalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel5Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jftfCedula,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jtfNombres,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jtfApellidos,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jlCarrera,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        jPanel6.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel6.setName("jPanel6"); // NOI18N
        jPanel6.setOpaque(false);

        jLabel28.setFont(resourceMap.getFont("jLabel19.font")); //
NOI18N
        jLabel28.setText(resourceMap.getString("jLabel28.text")); //
```

```
NOI18N
        jLabel28.setName("jLabel28"); // NOI18N

        jLabel29.setFont(resourceMap.getFont("jLabel19.font")); //
NOI18N
        jLabel29.setText(resourceMap.getString("jLabel29.text")); //
NOI18N
        jLabel29.setName("jLabel29"); // NOI18N

        jLabel30.setFont(resourceMap.getFont("jLabel19.font")); //
NOI18N
        jLabel30.setText(resourceMap.getString("jLabel30.text")); //
NOI18N
        jLabel30.setName("jLabel30"); // NOI18N

        javax.swing.GroupLayout jPanel6Layout = new
javax.swing.GroupLayout(jPanel6);
        jPanel6.setLayout(jPanel6Layout);
        jPanel6Layout.setHorizontalGroup(

jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel6Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addGroup(jPanel6Layout.createSequentialGroup()

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING)
                            .addComponent(jLabel28,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
jPanel6Layout.createSequentialGroup()
                                .addComponent(jLabel29)
                                .addGap(55, 55, 55)))
                        .addGap(36, 36, 36))
                    .addGroup(jPanel6Layout.createSequentialGroup()
                        .addComponent(jLabel30,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addGap(64, 64, 64))))
        );
        jPanel6Layout.setVerticalGroup(

jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel6Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel28,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel29,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel30,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        jPanel7.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel7.setName("jPanel7"); // NOI18N
        jPanel7.setOpaque(false);

        jlActivo.setFont(resourceMap.getFont("jlActivo.font")); //
NOI18N

jlActivo.setToolTipText(resourceMap.getString("jlActivo.toolTipText"))
; // NOI18N

jlActivo.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
        jlActivo.setName("jlActivo"); // NOI18N
        jlActivo.setPreferredSize(new java.awt.Dimension(0, 20));

        jtfTelefono.setEditable(false);

jtfTelefono.setToolTipText(resourceMap.getString("jtfTelefono.toolTipT
ext")); // NOI18N
        jtfTelefono.setName("jtfTelefono"); // NOI18N

        jtfCorreoElectronico.setEditable(false);

jtfCorreoElectronico.setToolTipText(resourceMap.getString("jtfCorreoEl
ectronico.toolTipText")); // NOI18N
        jtfCorreoElectronico.setName("jtfCorreoElectronico"); //
NOI18N

        javax.swing.GroupLayout jPanel7Layout = new
javax.swing.GroupLayout(jPanel7);
        jPanel7.setLayout(jPanel7Layout);
        jPanel7Layout.setHorizontalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jtfTelefono,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 171, Short.MAX_VALUE)
                    .addComponent(jtfCorreoElectronico,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 171, Short.MAX_VALUE)
                    .addComponent(jlActivo,
javax.swing.GroupLayout.DEFAULT_SIZE, 171, Short.MAX_VALUE))
                .addContainerGap())
        );
        jPanel7Layout.setVerticalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
```

```
ADING)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jtfCorreoElectronico,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jtfTelefono,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jlActivo,
javax.swing.GroupLayout.PREFERRED_SIZE, 20,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        javax.swing.GroupLayout jPanel3Layout = new
javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel5,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel6,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel7,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGap(18, 18, 18))
        );
        jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Al
```

```
ignment.TRAILING, false)
                        .addComponent(jPanel7,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jPanel6,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING, false)
                        .addComponent(jPanel5,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jPanel4,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        jPanel8.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel8.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel8.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel8.border.titleFont"))); // NOI18N
        jPanel8.setName("jPanel8"); // NOI18N

        jtfCota.setText(resourceMap.getString("jtfCota.text")); //
NOI18N

jtfCota.setToolTipText(resourceMap.getString("jtfCota.toolTipText"));
// NOI18N
        jtfCota.setName("jtfCota"); // NOI18N
        jtfCota.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusGained(java.awt.event.FocusEvent evt) {
                jtfCotaFocusGained(evt);
            }
        });
        jtfCota.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyPressed(java.awt.event.KeyEvent evt) {
                jtfCotaKeyPressed(evt);
            }
        });


botonBuscar.setBackground(resourceMap.getColor("botonBuscar.background
")); // NOI18N
        botonBuscar.setIcon(resourceMap.getIcon("botonBuscar.icon"));
// NOI18N

botonBuscar.setText(resourceMap.getString("botonBuscar.text")); //
NOI18N
        botonBuscar.setName("botonBuscar"); // NOI18N
```

```
        botonBuscar.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                botonBuscarActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel8Layout = new
javax.swing.GroupLayout(jPanel8);
        jPanel8.setLayout(jPanel8Layout);
        jPanel8Layout.setHorizontalGroup(

jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel8Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jtfCota,
javax.swing.GroupLayout.PREFERRED_SIZE, 125,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(botonBuscar)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel8Layout.setVerticalGroup(

jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel8Layout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

.addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(jtfCota,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(botonBuscar))
                .addContainerGap())
        );

        jPanel9.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel9.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel9.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel9.border.titleFont"))); // NOI18N
        jPanel9.setName("jPanel9"); // NOI18N
        jPanel9.setOpaque(false);


jScrollPane2.setBorder(javax.swing.BorderFactory.createEtchedBorder())
;
        jScrollPane2.setName("jScrollPane2"); // NOI18N

        jtListaEjemplares.setAutoCreateRowSorter(true);
```

```
        jtListaEjemplares.setModel(new
javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Nombre", "Cota", "Area", "Tipo", "Edición",
"Prestamo", "Deuda"
            }
        ) {
            Class[] types = new Class [] {
                java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class, java.lang.String.class
            };
            boolean[] canEdit = new boolean [] {
                false, false, false, false, false, false, false
            };

            public Class getColumnClass(int columnIndex) {
                return types [columnIndex];
            }

            public boolean isCellEditable(int rowIndex, int
columnIndex) {
                return canEdit [columnIndex];
            }
        });

jtListaEjemplares.setAutoResizeMode(javax.swing.JTable.AUTO_RESIZE_ALL
_COLUMNS);

jtListaEjemplares.setDebugGraphicsOptions(javax.swing.DebugGraphics.NO
NE_OPTION);
        jtListaEjemplares.setName("jtListaEjemplares"); // NOI18N

jtListaEjemplares.setSelectionMode(javax.swing.ListSelectionModel.SING
LE_SELECTION);

jtListaEjemplares.getTableHeader().setReorderingAllowed(false);
        jScrollPane2.setViewportView(jtListaEjemplares);

        jPanel11.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel11.setName("jPanel11"); // NOI18N
        jPanel11.setOpaque(false);

        jLabel10.setFont(resourceMap.getFont("jLabel13.font")); //
NOI18N

jLabel10.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel10.setText(resourceMap.getString("jLabel10.text")); //
NOI18N
        jLabel10.setName("jLabel10"); // NOI18N


jlFechaSalida.setFont(resourceMap.getFont("jlFechaSalida.font")); //
NOI18N

jlFechaSalida.setForeground(resourceMap.getColor("jlFechaSalida.foregr
ound")); // NOI18N
```

```
jlFechaSalida.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);

jlFechaSalida.setText(resourceMap.getString("jlFechaSalida.text")); //
NOI18N

jlFechaSalida.setToolTipText(resourceMap.getString("jlFechaSalida.tool
TipText")); // NOI18N
        jlFechaSalida.setName("jlFechaSalida"); // NOI18N

        jLabel12.setFont(resourceMap.getFont("jLabel13.font")); //
NOI18N

jLabel12.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel12.setText(resourceMap.getString("jLabel12.text")); //
NOI18N
        jLabel12.setName("jLabel12"); // NOI18N


jlFechaEntrega.setFont(resourceMap.getFont("jlFechaEntrega.font")); //
NOI18N

jlFechaEntrega.setForeground(resourceMap.getColor("jlFechaEntrega.fore
ground")); // NOI18N

jlFechaEntrega.setHorizontalAlignment(javax.swing.SwingConstants.LEFT)
;

jlFechaEntrega.setText(resourceMap.getString("jlFechaEntrega.text"));
// NOI18N

jlFechaEntrega.setToolTipText(resourceMap.getString("jlFechaEntrega.to
olTipText")); // NOI18N
        jlFechaEntrega.setName("jlFechaEntrega"); // NOI18N

        jLabel13.setFont(resourceMap.getFont("jLabel13.font")); //
NOI18N

jLabel13.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel13.setText(resourceMap.getString("jLabel13.text")); //
NOI18N
        jLabel13.setName("jLabel13"); // NOI18N


jlLibrosPrestados.setFont(resourceMap.getFont("jlLibrosPrestados.font"
)); // NOI18N

jlLibrosPrestados.setForeground(resourceMap.getColor("jlLibrosPrestado
s.foreground")); // NOI18N

jlLibrosPrestados.setText(resourceMap.getString("jlLibrosPrestados.tex
t")); // NOI18N

jlLibrosPrestados.setToolTipText(resourceMap.getString("jlLibrosPresta
dos.toolTipText")); // NOI18N
        jlLibrosPrestados.setName("jlLibrosPrestados"); // NOI18N

        javax.swing.GroupLayout jPanel11Layout = new
javax.swing.GroupLayout(jPanel11);
        jPanel11.setLayout(jPanel11Layout);
        jPanel11Layout.setHorizontalGroup(
```

```
jPanel11Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(jPanel11Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel11Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
                    .addComponent(jLabel13,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jLabel12,
javax.swing.GroupLayout.DEFAULT_SIZE, 113, Short.MAX_VALUE)
                    .addComponent(jLabel10,
javax.swing.GroupLayout.DEFAULT_SIZE, 113, Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel11Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)

.addGroup(jPanel11Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING, false)
                        .addComponent(jlFechaEntrega,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jlFechaSalida,
javax.swing.GroupLayout.PREFERRED_SIZE, 142,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jlLibrosPrestados,
javax.swing.GroupLayout.PREFERRED_SIZE, 87,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );
        jPanel11Layout.setVerticalGroup(

jPanel11Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(jPanel11Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel11Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING, false)
                    .addGroup(jPanel11Layout.createSequentialGroup()
                        .addComponent(jlFechaSalida,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jlFechaEntrega,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jlLibrosPrestados,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(jPanel11Layout.createSequentialGroup()
                        .addComponent(jLabel10,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel12,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jLabel13)))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        jPanel12.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel12.setName("jPanel12"); // NOI18N
        jPanel12.setOpaque(false);

        jLabel14.setFont(resourceMap.getFont("jLabel15.font")); //
NOI18N

jLabel14.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel14.setText(resourceMap.getString("jLabel14.text")); //
NOI18N
        jLabel14.setName("jLabel14"); // NOI18N

        jlFecha.setFont(resourceMap.getFont("jlFecha.font")); //
NOI18N

jlFecha.setForeground(resourceMap.getColor("jlFecha.foreground")); //
NOI18N

jlFecha.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
        jlFecha.setText(resourceMap.getString("jlFecha.text")); //
NOI18N

jlFecha.setToolTipText(resourceMap.getString("jlFecha.toolTipText"));
// NOI18N
        jlFecha.setName("jlFecha"); // NOI18N

        jLabel15.setFont(resourceMap.getFont("jLabel15.font")); //
NOI18N

jLabel15.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel15.setText(resourceMap.getString("jLabel15.text")); //
NOI18N
        jLabel15.setName("jLabel15"); // NOI18N

        jlRetraso.setFont(resourceMap.getFont("jlRetraso.font")); //
NOI18N

jlRetraso.setForeground(resourceMap.getColor("jlRetraso.foreground"));
// NOI18N

jlRetraso.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
        jlRetraso.setText(resourceMap.getString("jlRetraso.text")); //
NOI18N

jlRetraso.setToolTipText(resourceMap.getString("jlRetraso.toolTipText"
)); // NOI18N
        jlRetraso.setName("jlRetraso"); // NOI18N

        jLabel16.setFont(resourceMap.getFont("jLabel15.font")); //
```

```
NOI18N

jLabel16.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel16.setText(resourceMap.getString("jLabel16.text")); //
NOI18N
        jLabel16.setName("jLabel16"); // NOI18N

        jlDeuda.setFont(resourceMap.getFont("jlDeuda.font")); //
NOI18N

jlDeuda.setForeground(resourceMap.getColor("jlDeuda.foreground")); //
NOI18N

jlDeuda.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
        jlDeuda.setText(resourceMap.getString("jlDeuda.text")); //
NOI18N

jlDeuda.setToolTipText(resourceMap.getString("jlDeuda.toolTipText"));
// NOI18N
        jlDeuda.setName("jlDeuda"); // NOI18N

        jlMoneda2.setFont(resourceMap.getFont("jlMoneda2.font")); //
NOI18N

jlMoneda2.setForeground(resourceMap.getColor("jlMoneda2.foreground"));
// NOI18N
        jlMoneda2.setText(resourceMap.getString("jlMoneda2.text")); //
NOI18N
        jlMoneda2.setName("jlMoneda2"); // NOI18N

        javax.swing.GroupLayout jPanel12Layout = new
javax.swing.GroupLayout(jPanel12);
        jPanel12.setLayout(jPanel12Layout);
        jPanel12Layout.setHorizontalGroup(

jPanel12Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(jPanel12Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel12Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)

.addGroup(jPanel12Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING, false)
                        .addComponent(jLabel14,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jLabel15,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                    .addComponent(jLabel16,
javax.swing.GroupLayout.DEFAULT_SIZE, 89, Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel12Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING, false)
                    .addComponent(jlFecha,
javax.swing.GroupLayout.DEFAULT_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jlRetraso,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(jPanel12Layout.createSequentialGroup()
                        .addComponent(jlMoneda2,
javax.swing.GroupLayout.PREFERRED_SIZE, 22,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jlDeuda,
javax.swing.GroupLayout.PREFERRED_SIZE, 70,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(44, 44, 44))
        );
        jPanel12Layout.setVerticalGroup(

jPanel12Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(jPanel12Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel12Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
                    .addGroup(jPanel12Layout.createSequentialGroup()
                        .addComponent(jLabel14,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel15,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(jPanel12Layout.createSequentialGroup()
                        .addComponent(jlFecha,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jlRetraso,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel12Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.BASELINE)
                    .addComponent(jLabel16,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jlMoneda2,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jlDeuda,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(17, Short.MAX_VALUE))
        );

        jLabel1.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N
```

```
jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText(resourceMap.getString("jLabel1.text")); //
NOI18N
        jLabel1.setName("jLabel1"); // NOI18N


jlDeudaTotal.setFont(resourceMap.getFont("jlDeudaTotal.font")); //
NOI18N

jlDeudaTotal.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);

jlDeudaTotal.setText(resourceMap.getString("jlDeudaTotal.text")); //
NOI18N

jlDeudaTotal.setToolTipText(resourceMap.getString("jlDeudaTotal.toolTi
pText")); // NOI18N
        jlDeudaTotal.setName("jlDeudaTotal"); // NOI18N

        jlMoneda.setFont(resourceMap.getFont("jlDeudaTotal.font")); //
NOI18N

jlMoneda.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
        jlMoneda.setText(resourceMap.getString("jlMoneda.text")); //
NOI18N
        jlMoneda.setName("jlMoneda"); // NOI18N


jbBotonLimpiar.setIcon(resourceMap.getIcon("jbBotonLimpiar.icon")); //
NOI18N

jbBotonLimpiar.setText(resourceMap.getString("jbBotonLimpiar.text"));
// NOI18N

jbBotonLimpiar.setToolTipText(resourceMap.getString("jbBotonLimpiar.to
olTipText")); // NOI18N
        jbBotonLimpiar.setName("jbBotonLimpiar"); // NOI18N
        jbBotonLimpiar.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jbBotonLimpiarActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel9Layout = new
javax.swing.GroupLayout(jPanel9);
        jPanel9.setLayout(jPanel9Layout);
        jPanel9Layout.setHorizontalGroup(

jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel9Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jScrollPane2,
javax.swing.GroupLayout.DEFAULT_SIZE, 706, Short.MAX_VALUE)
                    .addGroup(jPanel9Layout.createSequentialGroup()
                        .addComponent(jPanel11,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(jPanel12,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel9Layout.createSequentialGroup()
                        .addComponent(jbBotonLimpiar,
javax.swing.GroupLayout.PREFERRED_SIZE, 130,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
376, Short.MAX_VALUE)
                        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 89,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jlMoneda,
javax.swing.GroupLayout.PREFERRED_SIZE, 32,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(jlDeudaTotal,
javax.swing.GroupLayout.PREFERRED_SIZE, 63,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap())
        );
        jPanel9Layout.setVerticalGroup(

jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel9Layout.createSequentialGroup()

.addGroup(jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addGroup(jPanel9Layout.createSequentialGroup()
                        .addComponent(jPanel11,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED))
                    .addComponent(jPanel12,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 118,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)

.addGroup(jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
```

```
                        .addComponent(jlDeudaTotal,
javax.swing.GroupLayout.PREFERRED_SIZE, 17,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jlMoneda,
javax.swing.GroupLayout.PREFERRED_SIZE, 17,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 17,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jbBotonLimpiar))
                .addContainerGap())
        );


botonRecibirEjemplares.setIcon(resourceMap.getIcon("botonRecibirEjempl
ares.icon")); // NOI18N

botonRecibirEjemplares.setText(resourceMap.getString("botonRecibirEjem
plares.text")); // NOI18N
        botonRecibirEjemplares.setEnabled(false);
        botonRecibirEjemplares.setName("botonRecibirEjemplares"); //
NOI18N
        botonRecibirEjemplares.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                botonRecibirEjemplaresActionPerformed(evt);
            }
        });


botonCancelar.setIcon(resourceMap.getIcon("botonCancelar.icon")); //
NOI18N

botonCancelar.setText(resourceMap.getString("botonCancelar.text")); //
NOI18N
        botonCancelar.setName("botonCancelar"); // NOI18N

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                            .addComponent(jPanel9,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jPanel3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
```

```
layout.createSequentialGroup()
                                .addComponent(botonRecibirEjemplares)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(botonCancelar,
javax.swing.GroupLayout.PREFERRED_SIZE, 109,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                        .addContainerGap())
                    .addComponent(jPanel8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel9,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
328, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                    .addComponent(botonRecibirEjemplares,
javax.swing.GroupLayout.PREFERRED_SIZE, 23,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(botonCancelar,
javax.swing.GroupLayout.PREFERRED_SIZE, 23,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );
    }// </editor-fold>

    private void botonBuscarActionPerformed(java.awt.event.ActionEvent
evt) {
         buscarCota();
    }

    private void jtfCotaKeyPressed(java.awt.event.KeyEvent evt) {
        if (evt.getKeyCode()==KeyEvent.VK_ENTER)
            buscarCota();
}

    private void jtfCotaFocusGained(java.awt.event.FocusEvent evt) {
        jtfCota.setBackground(Color.WHITE);
```

```
    }

    private void
jbBotonLimpiarActionPerformed(java.awt.event.ActionEvent evt) {
        limpiarFormularioEstudiante();
}

    private void
botonRecibirEjemplaresActionPerformed(java.awt.event.ActionEvent evt)
{
        recibirEjemplares();
    }

    private void botonCobrarActionPerformed(java.awt.event.ActionEvent
evt) {
        double monto ;
        jlError.setText(" ");
        try {
            jftfMontoCancelado.commitEdit();
            String montoString =
jftfMontoCancelado.getValue().toString().trim().replace(",",".");
            System.out.println("MONTO STRING "+montoString);
            monto =
Double.parseDouble(jftfMontoCancelado.getValue().toString().trim().rep
lace(",","."));

            jDialog1.dispose();
            setMontoCancelado (monto);
        }
        catch (ParseException ex) {

Logger.getLogger(GUIrecibirEjemplar.class.getName()).log(Level.SEVERE,
null, ex);
        }       catch(NumberFormatException  e){
            jlError.setText("Ingrese un monto valido.");
            pantallaPrincipal.errorBarraEstado("Ingrese un monto
válido para el pago de la deuda."+e);
        }
}

    private void jftfMontoCanceladoKeyTyped(java.awt.event.KeyEvent
evt) {

    }

    private void jtfDiferenciaKeyTyped(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
}

    private void jtfTotalKeyTyped(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
}

    private void jftfMontoCanceladoKeyPressed(java.awt.event.KeyEvent
evt) {
        jlError.setText(" ");
        double temp=0;
        if (!jftfMontoCancelado.getText().isEmpty() &&
evt.getKeyCode()==KeyEvent.VK_ENTER){
        try{
            try{
```

```
                jftfMontoCancelado.commitEdit();
                temp=
Double.valueOf(jftfMontoCancelado.getText().replaceAll(",", "."));
                double subTotal = (DEUDA_TOTAL - temp);
                if (temp>=0.0)
                    if (subTotal>0.0){
                        jtfDiferencia.setForeground(Rojo);
                        jlDiferencia.setText("Diferencia:");
                        botonCobrar.setEnabled(true);
                    }
                    else{
                        jtfDiferencia.setForeground(Verde);
                        if (subTotal<0.0){
                            subTotal*=-1.0;
                            jlDiferencia.setText("Vuelto:");
                            botonCobrar.setEnabled(true);
                        } else{
                            jlDiferencia.setText("Pago exacto:");
                            botonCobrar.setEnabled(true);
                        }
                    }
                jtfDiferencia.setText((""+
subTotal).replace(".",","));


jftfMontoCancelado.setText(String.valueOf(temp).replace(".",","));
                jftfMontoCancelado.commitEdit();
            } catch (ParseException ex) {

Logger.getLogger(GUIrecibirEjemplar.class.getName()).log(Level.SEVERE,
null, ex);
            }

        } catch (NumberFormatException ex){

Logger.getLogger(GUIrecibirEjemplar.class.getName()).log(Level.SEVERE,
null, ex);
            jlError.setText("Ingrese un monto valido.");
            jftfMontoCancelado.setText("");
        }

        }
        else {
            jlError.setText(" ");
            botonCobrar.setEnabled(false);
            jlDiferencia.setText("Diferencia:");
        }
    }


    // Variables declaration - do not modify
    private javax.swing.JButton botonBuscar;
    private javax.swing.JButton botonCancelar;
    private javax.swing.JButton botonCobrar;
    private javax.swing.JButton botonRecibirEjemplares;
    private javax.swing.JDialog jDialog1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
```

```
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel16;
    private javax.swing.JLabel jLabel19;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel20;
    private javax.swing.JLabel jLabel21;
    private javax.swing.JLabel jLabel28;
    private javax.swing.JLabel jLabel29;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel30;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel10;
    private javax.swing.JPanel jPanel11;
    private javax.swing.JPanel jPanel12;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JPanel jPanel5;
    private javax.swing.JPanel jPanel6;
    private javax.swing.JPanel jPanel7;
    private javax.swing.JPanel jPanel8;
    private javax.swing.JPanel jPanel9;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JButton jbBotonLimpiar;
    private javax.swing.JFormattedTextField jftfCedula;
    private javax.swing.JFormattedTextField jftfMontoCancelado;
    private javax.swing.JLabel jlActivo;
    private javax.swing.JLabel jlCarrera;
    private javax.swing.JLabel jlDeuda;
    private javax.swing.JLabel jlDeudaTotal;
    private javax.swing.JLabel jlDiferencia;
    private javax.swing.JLabel jlError;
    private javax.swing.JLabel jlFecha;
    private javax.swing.JLabel jlFechaEntrega;
    private javax.swing.JLabel jlFechaSalida;
    private javax.swing.JLabel jlLibrosPrestados;
    private javax.swing.JLabel jlMoneda;
    private javax.swing.JLabel jlMoneda2;
    private javax.swing.JLabel jlRetraso;
    private javax.swing.JTable jtListaEjemplares;
    private javax.swing.JTextField jtfApellidos;
    private javax.swing.JTextField jtfCorreoElectronico;
    private javax.swing.JTextField jtfCota;
    private javax.swing.JFormattedTextField jtfDiferencia;
    private javax.swing.JTextField jtfNombres;
    private javax.swing.JTextField jtfTelefono;
    private javax.swing.JFormattedTextField jtfTotal;
    // End of variables declaration

    // Gestor de acceso a la bd y procesamiento
    private GestorRecibirEjemplar gestorRecibirEjemplar;

    // Gestores de informacion
    private GestorInfoPrestamo gestorInfoPrestamo;
    private GestorInfoEjemplar gestorInfoEjemplar;
    private GestorInfoEstudiantes gestorInfoEstudiante;

    // Gestor Informacion del sistema
    private GestorVariablesSistema gestorVariablesSistema;
```

```
    private ArrayList listaGestorInfoPrestamo;

    // Variables
    private boolean ESTUDIANTE_CARGADO=false;
    private boolean DEUDA_CANCELADA =false;
    private double DEUDA_TOTAL;
    private double DEUDA_RESTANTE;

    private double DEUDA_UN_LIBRO;
    private int RETRASO;
    private double MONTO_CANCELADO;
    private DecimalFormat formatoDecimal;


    // Metodo llamado del evento del boto recibir ejemplar
    // @ Si la deuda es cero se puede recibir los ejemplares
    //   directamente
    // @ Si la deuda es mayor que cero se llama al metodo
    // mostrarJDialog() que muestra el jDialog1 para cobrar
    // la deuda
    public void recibirEjemplares(){
        // Verifica si hay alguna deuda o si no ha sido cancelada
        // entonces despliega el mensaje de cobro, sino
        // recibe el ejemplar
        if (DEUDA_TOTAL>0 &&  !DEUDA_CANCELADA){

jtfTotal.setText(String.valueOf(DEUDA_TOTAL).replace(".",","));
            try {
                jtfTotal.commitEdit();
            } catch (ParseException ex) {

Logger.getLogger(GUIrecibirEjemplar.class.getName()).log(Level.SEVERE,
null, ex);
            }
            mostrarJDialog();
        }
        else {
            botonRecibirEjemplares.setText("Recibir ejemplares");
            // Marca todos lo ejemplares como recibidos en la bd
            actualizarEjemplaresDisponibles(listaGestorInfoPrestamo);
            System.out.println("DEUDA TOTAL = "+DEUDA_TOTAL +" DEUDA
RESTANTE = "+DEUDA_RESTANTE);
            if (DEUDA_RESTANTE>0.0)
                gestorRecibirEjemplar.generarMultas
(listaGestorInfoPrestamo,false);

            else
                gestorRecibirEjemplar.generarMultas
(listaGestorInfoPrestamo,true);
            actualizarPrestamosRecibido(listaGestorInfoPrestamo);

actualizarSolvenciaEstudiantes(((GestorInfoPrestamo)listaGestorInfoPre
stamo.get(0)).getCedulaEstudiante());
            pantallaPrincipal.mensajeBarraEstado("Libros recibidos
satisfactoriamente.");
            limpiarFormularioEstudiante();
        }
    }

    //
    private void actualizarSolvenciaEstudiantes(String cedula){
```

```
            gestorRecibirEjemplar.actualizarEstadoSolvente(cedula);
        }

        // Metodo que toma la cota de todos los ejemplares recibidos
        // revisa la tabla ejemplare y marca a cada uno como recibido
        private void actualizarEjemplaresDisponibles(ArrayList
listaGestorInfoPrestamo) {

gestorRecibirEjemplar.marcarEjemplaresDisponibles(listaGestorInfoPrest
amo);
        }
        // Dado una lista de prestamos actualiza el estado de todos
        // los prestamos como recibido
        private void actualizarPrestamosRecibido(ArrayList
listaGestorInfoPrestamo) {

gestorRecibirEjemplar.marcarPrestamosRecibidos(listaGestorInfoPrestamo
);
        }



        // Metodo que retorna la deuda de los ejemplares recibidos
        // buscado los valores en la tabla jtListaEjemplares
        private Double[] getDeudaEjemplares(){
            Double deuda_ejemplares []=null;
            for (int i=0;i<jtListaEjemplares.getRowCount();i++){
                deuda_ejemplares [i] =
Double.valueOf(jtListaEjemplares.getModel().getValueAt(i,
6).toString());
            }
            return deuda_ejemplares;
        }

        // Metodo que retorna la cota de los ejemplares recibidos
        // buscado los valores en la tabla jtListaEjemplares
        private String[] getCotasEjemplares (){
            String cotas_deudas [] = new
String[jtListaEjemplares.getRowCount()];
            for (int i=0;i<jtListaEjemplares.getRowCount();i++){
                cotas_deudas[i]=jtListaEjemplares.getModel().getValueAt(i,
1).toString().trim();
            }
            return cotas_deudas;
        }

        // Metodo principal de la clase, que dada una cota llama a metodos
        // de la clase GestorRecibirEjemplares y buscar los ejemplares
        // por la cota dada.
        private void buscarCota() {
            gestorInfoPrestamo = new GestorInfoPrestamo();
            jtfCota.setBackground(Color.WHITE);
            // Establece fecha
            if (!ESTUDIANTE_CARGADO)
                gestorInfoEstudiante = new GestorInfoEstudiantes();

            // Buscar el prestamos del ejemplar cargado
            // si no hay ningun estudiante cargado el buscara cualquier
            // cota, en caso contrario solo buscara las cotas de
            // los prestamos cuya cedula coincida con la del estudiante
            if (!ESTUDIANTE_CARGADO){
```

```
            gestorInfoPrestamo=
buscarPrestamo(jtfCota.getText().trim(),"!!!");
            if (gestorInfoPrestamo==null){
                jtfCota.setBackground(Color.PINK);
                return;
            }
        }
        else{
            gestorInfoPrestamo=
buscarPrestamo(jtfCota.getText().trim(),String.valueOf(gestorInfoEstud
iante.getCedula()));
            if (gestorInfoPrestamo==null){
                jtfCota.setBackground(Color.PINK);
                return;
            }
        }

        // Dezpliega la informacion del prestamo cargado
        if (!mostrarInfoPrestamo(gestorInfoPrestamo)){
            jtfCota.setBackground(Color.PINK);
            return;
        }

        // Si el estediante no ha sido cargado lo carga de la BD
        // y los dezpliega en la pantalla
        if (!ESTUDIANTE_CARGADO){
            gestorInfoEstudiante =
buscarEstudiante(gestorInfoPrestamo.getCedulaEstudiante());
            mostrarEstudiante(gestorInfoEstudiante);
            ESTUDIANTE_CARGADO=true;
        }
        //Verifica que una persona solo retorne libros que le hayan
        // sido prestados
        if
(gestorInfoPrestamo.getCedulaEstudiante().compareTo(String.valueOf(ges
torInfoEstudiante.getCedula()))!=0){
            pantallaPrincipal.errorBarraEstado("El libro
"+gestorInfoPrestamo.getIdEjemplar()+" no fue prestado al estudiante
"+gestorInfoEstudiante.getCedula());
            jtfCota.setBackground(Color.PINK);
            return;
        }

        // Verifica si ya se cargo ese ejemplar en la lista,
        // de ser asi no se añade y detiene la ejecucion de
        // este metodo
        if
(verificarExisteEnLista(gestorInfoPrestamo.getIdEjemplar())){
            pantallaPrincipal.advertenciaBarraEstado("La cota
"+gestorInfoPrestamo.getIdEjemplar()+" ya se cargo en la lista de
entrega.");
            jtfCota.setBackground(Color.PINK);
            return;
        }
        gestorInfoEjemplar =
buscarEjemplar(gestorInfoPrestamo.getIdEjemplar());
        if (gestorInfoEjemplar==null){
            jtfCota.setBackground(Color.PINK);
            return;
        }
```

```
        // Se determina la deuda y luego dependiendo del
        // resultado se le da color a los campos deudas y
        // se le añaden los valores correspondiente
        DEUDA_UN_LIBRO = calcularDeuda(RETRASO);
        DEUDA_TOTAL += DEUDA_UN_LIBRO;
        if (DEUDA_TOTAL>0){
            jlDeudaTotal.setForeground(Rojo);
            jlDeuda.setForeground(Rojo);
        }
        else {
            jlDeudaTotal.setForeground(Verde);
            jlDeuda.setBackground(Verde);
        }
        formatoDecimal = new DecimalFormat("0.00");


jlDeudaTotal.setText(String.valueOf(formatoDecimal.format(DEUDA_TOTAL)
)+" ");

jlDeuda.setText(String.valueOf(formatoDecimal.format(DEUDA_UN_LIBRO))+
" ");

        // Se añade la nueva informacion del ejemplar a la table
        // de recibir ejemplares añadiendo tambien la deuda de ese
        // ejemplar, ademas se habilita el boton recibir ejemplares

añadirInfoTabla(gestorInfoEjemplar,String.valueOf(formatoDecimal.forma
t(DEUDA_UN_LIBRO)));
        gestorInfoPrestamo.setDeuda(DEUDA_UN_LIBRO);
        gestorInfoPrestamo.setDiasAtraso(RETRASO);
        listaGestorInfoPrestamo.add(gestorInfoPrestamo);
        botonRecibirEjemplares.setEnabled(true);

    }


    // Metodo que calcula la deuda de un ejemplar dado la cantidad de
dias
    // de atraso que tenga y las varibles del sistema
    // cargadas en el gestor variables del sistema que indica el cobro
    // por dia de atraso que se le hara al estudiante.
    public double calcularDeuda (double retraso){
        gestorVariablesSistema = new GestorVariablesSistema
(pantallaPrincipal);
        if (retraso ==0)
            return 0.0;
        else{
            botonRecibirEjemplares.setText("Cancelar deuda");
            if (retraso == 1){
                return gestorVariablesSistema.getCargoPorUnDia();
            } else
                if (retraso == 2)
                    return
gestorVariablesSistema.getCargoPorDosDias();
                else
                    return
gestorVariablesSistema.getCargoPorDosDias()+ ((retraso-
2)*gestorVariablesSistema.getCargoPorDiaExtra());
        }
    }
```

```
    // Verifica si la cota de un ejemplar dado por una instancia de
gestorEjemplar
    // existe o no en la lista de ejemplares retornando un boleano
para indicar
    // su existencia
    private boolean verificarExisteEnLista (String cota){
         for (int
i=0;i<jtListaEjemplares.getModel().getRowCount();i++){
            if
(jtListaEjemplares.getValueAt(i,1).toString().trim().equals(cota)){
                pantallaPrincipal.advertenciaBarraEstado("La cota
"+cota+" ya fue cargada.");
                jtfCota.setBackground(Color.PINK);
                return true;
            }
        }
     return false;
    }

    // Añade la informacion de un ejemplar dado por el objeto
getorEjemplar
    // a la tabla que muestra la lista de los ejemplares a prestar
    private void  añadirInfoTabla (GestorInfoEjemplar gestorEjemplar,
String deuda){
        DefaultTableModel tablaTemporal = (DefaultTableModel)
jtListaEjemplares.getModel();
        String interno;

        String libro;
        if (gestorEjemplar.getINTERNO())
                interno="INTERNO";
            else
                interno= "CIRCULANTE";

            if (gestorEjemplar.getLIBRO())
                libro = "LIBRO";
            else
                libro= "TESIS";

            Object[] nuevo = {gestorEjemplar.getNombreEjemplar(),
                              gestorEjemplar.getIdEjemplar(),
                              gestorEjemplar.getArea(),
                              libro,
                              gestorEjemplar.getEdicion(),
                              interno,
                              deuda};
             tablaTemporal.addRow(nuevo);
    }

    // Llama al metodo del gestor de Recibir ejemplares y llama
    // al metodo para buscar en la base de datos usando como
referencia
    // la cedula obtenida por la busqueda del prestamo
    private GestorInfoEjemplar buscarEjemplar (String cota){
        gestorRecibirEjemplar = new
GestorRecibirEjemplar(pantallaPrincipal);
        return gestorRecibirEjemplar.buscarEjemplar (cota);
    }
```

```
    // Llama al metodo del gestor de Recibir ejemplares y llama
    // al metodo para buscar en la base de datos usando como
referencia
    // la cedula obtenida por la busqueda del prestamo
    private GestorInfoEstudiantes buscarEstudiante (String cedula){
        gestorRecibirEjemplar = new
GestorRecibirEjemplar(pantallaPrincipal);
        return gestorRecibirEjemplar.buscarEstudiante(cedula);
    }

    // Metodo que llama al gestor Recibir Ejemplar
    // y busca la informacion de un prestamo en la base de datos
    // retornandola en una estructura de datos infoPrestamo
    private GestorInfoPrestamo buscarPrestamo(String cota, String
cedula){
        gestorRecibirEjemplar = new
GestorRecibirEjemplar(pantallaPrincipal);
        return
gestorRecibirEjemplar.buscarPrestamoPorCota(cota,cedula);
    }

    // Dezpliega la informacion del prestamo y determina las fechas
    // de de recepcion determinando asi si hay algun retraso o no
    private boolean mostrarInfoPrestamo (GestorInfoPrestamo
gestorInfoPrestamo){
        String fecha = convertirFecha (gestorInfoPrestamo.getFecha());
        jlFechaSalida.setText(fecha);

        fecha = convertirFecha (gestorInfoPrestamo.getFechaEntrega());
        jlFechaEntrega.setText(fecha);

        Calendar hoy ;
        Calendar diaPrestamo;
        Calendar diaRecepcion;

        hoy = Calendar.getInstance();
        diaPrestamo = Calendar.getInstance();
        diaRecepcion = Calendar.getInstance();


jlFecha.setText(String.valueOf(hoy.get(Calendar.DAY_OF_MONTH))+"/"+Str
ing.valueOf(hoy.get(Calendar.MONTH)+1)+"/"+hoy.get(Calendar.YEAR));

        diaPrestamo.set(getAño(gestorInfoPrestamo.getFecha()),
                        getMes(gestorInfoPrestamo.getFecha()),
                        getDia(gestorInfoPrestamo.getFecha()));

        diaRecepcion.set(getAño(gestorInfoPrestamo.getFechaEntrega()),
                         getMes(gestorInfoPrestamo.getFechaEntrega()),

getDia(gestorInfoPrestamo.getFechaEntrega()));
 //
jlFechaEntrega.setText(diaRecepcion.get(Calendar.DAY_OF_MONTH)+"/"+dia
Recepcion.get(Calendar.MONTH)+"/"+diaRecepcion.get(Calendar.YEAR));
        int diasRetraso = hoy.compareTo(diaPrestamo);

        if (diasRetraso<0){
            jlRetraso.setText(" ERROR");
            RETRASO = -1;
            pantallaPrincipal.errorBarraEstado("La fecha de entrega no
puede ser menor que la fecha de retiro, por favor revise el reloj del
```

```
sistema operativo. ");
            return false;
        }
        diasRetraso = hoy.compareTo(diaRecepcion);


        if (diasRetraso<=0){
            RETRASO=0;
            jlRetraso.setText(""+RETRASO+" días");

        }
        else {
            RETRASO = calcularRetraso(hoy,diaRecepcion);
            jlRetraso.setText(""+RETRASO+" días");
        }
        if (!ESTUDIANTE_CARGADO){
            gestorRecibirEjemplar = new
GestorRecibirEjemplar(pantallaPrincipal);
            jlLibrosPrestados.setText("
"+String.valueOf(gestorRecibirEjemplar.librosPrestadosAlEstudiante(ges
torInfoPrestamo.getCedulaEstudiante()))+" ejemplares");
        }


        return true;
    }

    // dado dos fechas 2da mayor o igual que la primera calcula la
    // diferencia de dias que hay entre ellas
    private int calcularRetraso(Calendar hoy, Calendar diaRecepcion) {
        int i=0;
        while (!hoy.equals(diaRecepcion)){
            diaRecepcion.add(Calendar.DAY_OF_MONTH,1);
            i++;
        }
        return i;

    }

    // Reciba una fecha de la BD y retorna un formato
    // compatible con el Calendar
    private String convertirFecha(String fecha){
        String separa[] = fecha.split("-");
        return (separa[2]+"/"+separa[1]+"/"+separa[0]);
    }

    // Recibe un string con la fecha del sistema
    // y retorna el dia
    private int getDia (String fecha){
         String separa[] = fecha.split("-");
         return Integer.parseInt(separa[2]);
    }

    // Recibe un string con la fecha del sistema
    // y retorna el mes
    private int getMes (String fecha){
         String separa[] = fecha.split("-");
         return Integer.parseInt(separa[1])-1;
    }

    // Recibe un string con la fecha del sistema
```

```
    // y retorna el año
    private int getAño (String fecha){
         String separa[] = fecha.split("-");
         return Integer.parseInt(separa[0]);
    }

    // Dezpliega la informacion del estudiante en la interfaz grafica
    // @ Recibe GestirInfoEstudiantes
    private void mostrarEstudiante (GestorInfoEstudiantes
gestorInfoEstudiantes){

jftfCedula.setText(String.valueOf(gestorInfoEstudiantes.getCedula()));
                jtfNombres.setText(gestorInfoEstudiantes.getNombre());

jtfApellidos.setText(gestorInfoEstudiantes.getApellido());

jtfCorreoElectronico.setText(gestorInfoEstudiantes.getCorreoElectronic
o());

jtfTelefono.setText(gestorInfoEstudiantes.getTelefono());
                if (gestorInfoEstudiantes.getCOMPUTACION()){
                   jlCarrera.setText("Computacion");
                }
                if (gestorInfoEstudiantes.getSISTEMAS()){
                   jlCarrera.setText("Sistemas");
                }
                if
(gestorInfoEstudiantes.getSISTEMAS()&&gestorInfoEstudiantes.getCOMPUTA
CION()){
                   jlCarrera.setText("Computacion y sistemas.");
                }

                if (gestorInfoEstudiantes.getACTIVO()){
                    jlActivo.setForeground(Azul);
                    jlActivo.setText("INSCRITO");
                }
                else{
                    jlActivo.setForeground(Color.DARK_GRAY);
                    jlActivo.setText("NO INSCRITO");
                    pantallaPrincipal.errorBarraEstado("El estudiante
no se encuentra inscrito y no puede retirar nigun ejemplar de la
sala");
                    return;
                }
    }

    // Metodo para ajustar el tamaño de la tabla que lista los
estudiantes
    private void ajustarAnchoTabla (){
        for (int i = 0; i < 6; i++) {
            column = jtListaEjemplares.getColumnModel().getColumn(i);
            if(i == 0)
                column.setPreferredWidth(338);
            else
                if (i==1)
                            column.setPreferredWidth(70);
                else
                    if(i == 2)
                        column.setPreferredWidth(126);
                    else
                        if(i == 3)
```

```
                                column.setPreferredWidth(49);

                                if (i==4)
                                    column.setPreferredWidth(51);
                                else
                                    if (i==5)
                                    column.setPreferredWidth(119);
                                    else
                                        if (i==6)
                                            column.setPreferredWidth(119);
            }
        }

            // Metodo que limpia los valores en los formularios y los deja
    en el
        // estado activo en el que se entoncontraban cuando se inicio la
        // la la interfaz grafica
        private void limpiarFormularioEstudiante(){

            ESTUDIANTE_CARGADO=false;
            DEUDA_CANCELADA=false;
            DEUDA_TOTAL=0;
            DEUDA_UN_LIBRO=0;
            RETRASO=0;
            jtfCota.setText("");
            jtfCota.setBackground(Color.WHITE);
            botonBuscar.setEnabled(true);
            jftfCedula.setText("");
            jtfNombres.setText("");
            jtfApellidos.setText("");
            jtfCorreoElectronico.setText("");
            jtfTelefono.setText("");
            jlCarrera.setText(" ");
            jlActivo.setText(" ");
            jlLibrosPrestados.setText("");
            listaGestorInfoPrestamo = new ArrayList();
            limpiarComponentesPrestamo ();
        }

        // Metodo para limpiar volver a iniciar los
        // valores de la zona de prestamo
        private void limpiarComponentesPrestamo (){
            // Componentes del prestamo
            botonRecibirEjemplares.setEnabled(false);
            jbBotonLimpiar.setEnabled(true);
            jlFechaSalida.setText("");
            jlFechaEntrega.setText("");
            jlLibrosPrestados.setText("");
            jlRetraso.setText("");
            jlDeuda.setText("");
            jlDeudaTotal.setText("");
            limpiarTabla ();
        }


        // Limpia la tabla eliminando todas la filas cargadas.
        private void limpiarTabla (){
            int columnas =
    ((DefaultTableModel)jtListaEjemplares.getModel()).getRowCount();
            if (columnas >0){
                for (int i=0;i<columnas;i++)
```

```
((DefaultTableModel)jtListaEjemplares.getModel()).removeRow(0);
        }
    }


    // Metodo llamado cuando el monto ha sido cancelado
    // establece los nuevo valores de las deuda y habilita
    // la opcion de recibir los ejemplares
    public void setMontoCancelado(double MONTO_CANCELADO){
        this.MONTO_CANCELADO=MONTO_CANCELADO;
        DEUDA_RESTANTE= DEUDA_TOTAL-MONTO_CANCELADO;
        jtfCota.setEnabled(false);
        jbBotonLimpiar.setEnabled(false);
        botonBuscar.setEnabled(false);
        if (DEUDA_RESTANTE>0.0){
            pantallaPrincipal.advertenciaBarraEstado("El estudiante
quedará con una deuda de "+gestorVariablesSistema.getMoneda()+"
"+DEUDA_RESTANTE+", ya puede recibir los ejemplares.");
            formatoDecimal = new DecimalFormat("0.00");

jlDeudaTotal.setText(String.valueOf(formatoDecimal.format(DEUDA_RESTAN
TE))+" ");

jlDeuda.setText(String.valueOf(formatoDecimal.format(DEUDA_RESTANTE))+
" ");
        }

        else{
            jlDeuda.setForeground(Verde);
            jlDeudaTotal.setForeground(Verde);
            pantallaPrincipal.mensajeBarraEstado("La deuda de este
préstamo ya ha sido cancelada, ya puede recibir los ejemplares.");
            formatoDecimal = new DecimalFormat("0.00");

jlDeudaTotal.setText(String.valueOf(formatoDecimal.format(0.0))+" ");

jlDeuda.setText(String.valueOf(formatoDecimal.format(0.0))+" ");
        }
        int cantEjemplares = jtListaEjemplares.getRowCount();

        if (MONTO_CANCELADO>DEUDA_TOTAL)
                for (int i=0;i<cantEjemplares;i++){
                    jtListaEjemplares.setValueAt("0.0",i, 6);

((GestorInfoPrestamo)listaGestorInfoPrestamo.get(i)).setDeuda(0.0);
                }
        else
            if (MONTO_CANCELADO>0.0)
                    for (int i=0;i<cantEjemplares;i++){
                        Double deudaTemp =
Double.parseDouble(jtListaEjemplares.getValueAt(i,
6).toString().replace(",","."));
                        deudaTemp =(deudaTemp * DEUDA_RESTANTE
)/DEUDA_TOTAL;

((GestorInfoPrestamo)listaGestorInfoPrestamo.get(i)).setDeuda(deudaTem
p);
                        System.out.println("Deuda TEMPORAL "+
deudaTemp);
                        jtListaEjemplares.setValueAt(
```

```
String.valueOf(formatoDecimal.format(deudaTemp)),i, 6);
                    }


        DEUDA_CANCELADA=true;
        botonRecibirEjemplares.setText("Recibir ejemplares");
     }

     // Dezpliega un JDialog para cobrar la multa de un ejemplar
    // en el centro de la pantalla
    private void mostrarJDialog(){

jDialog1.setLocation((this.getWidth()/2)+this.getLocationOnScreen().x-
160, (this.getHeight()/2)+this.getLocationOnScreen().y-120);
        jDialog1.setSize(320,240);
        jDialog1.setVisible(true);
    }
}
```

5.3.2.2.4 GestorRecibirEjemplar

```
package GestionPrestamos;


import BaseDeDatos.ConexionBD;
import GestionEjemplares.GestorInfoEjemplar;
import GestionEstudiantes.GestorInfoEstudiantes;
import Principal.GUIView;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.util.regex.Pattern;

public class GestorRecibirEjemplar {
    private ConexionBD conexionBD;
    private ResultSet rs = null;
    private GUIView pantallaPrincipal;
    private GestorInfoPrestamo gestorInfoPrestamo;
    private GestorInfoEstudiantes gestorEstudiante;

    private GestorInfoEjemplar gestorEjemplar;
    GestorRecibirEjemplar(GUIView pantallaPrincipal) {
        this.pantallaPrincipal = pantallaPrincipal;

    }

    // Dado un numero de cedula y de  cota, este metodo consultar en
la base
    // de datos y busca la infromacion de un prestamo no devuelto de
dicho
    // ejemplar a dicho estudiante
    public GestorInfoPrestamo buscarPrestamoPorCota(String cota,String
cedula) {
         // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Instancia un nuevo gestor infor prestamo
        gestorInfoPrestamo = new GestorInfoPrestamo();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        if (cedula.compareTo("!!!")==0)
            conexionBD.Consulta("SELECT * FROM prestamo WHERE
idejemplar= '"+cota+"' AND recibido='false'");
        else
            conexionBD.Consulta("SELECT * FROM prestamo WHERE
idejemplar= '"+cota+"' AND nrocedulaestudiante = '"+cedula+"' AND
recibido = 'false'");
        rs = conexionBD.MostrarDatosConsulta();
        try {
            if (rs.next()){

gestorInfoPrestamo.setIdPrestamo(rs.getString("idprestamo"));

gestorInfoPrestamo.setCeduladEstudiante(rs.getString("nrocedulaestudia
```

```
nte"));
                gestorInfoPrestamo.setFecha(rs.getString("fecha"));

gestorInfoPrestamo.setFechaEntrega(rs.getString("fechaentrega"));

gestorInfoPrestamo.setIdAyudante(rs.getString("idayudante"));

gestorInfoPrestamo.setIdEjemplar(rs.getString("idejemplar"));

gestorInfoPrestamo.setPrestamoINTERNO(rs.getBoolean("interno"));
                conexionBD.CerrarConexion();
                return gestorInfoPrestamo;
            }
            else {
                conexionBD.CerrarConexion();
                if (cedula.compareTo("!!!")==0)
                    pantallaPrincipal.advertenciaBarraEstado("La cota
"+cota+" no ha sido prestada o no esta en la base de datos.");
                else
                    pantallaPrincipal.advertenciaBarraEstado("La cota
"+cota+" no ha sido prestada al estudiante "+cedula+" o no esta en la
base de datos.");
                return null;
            }
        } catch (SQLException ex) {

Logger.getLogger(GestorRecibirEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            pantallaPrincipal.errorBarraEstado("Error al accesar a la
base de datos.");
            conexionBD.CerrarConexion();
            return null;
        }
    }

    // Dado un numero de cedula este metodo retonr la un objeto
getorInfo
    // estudiante con la informacion de un estudiante, en caso que no
se cosiga
    // de retornara el objeto con una cedula = -1
    public GestorInfoEstudiantes buscarEstudiante(String cedula) {
        // Limpia la barra de estado de la pantalla principal
        pantallaPrincipal.borrarBarraEstado();
        gestorEstudiante = new GestorInfoEstudiantes();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        conexionBD.Consulta("SELECT nombres, apellidos, email, activo,
computacion, sistemas, numerodetelefono FROM estudiantes WHERE
numerocedula = '"+cedula+"'");
            rs = conexionBD.MostrarDatosConsulta();
                try {
                    if (rs.next()) {

gestorEstudiante.setCedula(Long.parseLong(cedula));

gestorEstudiante.setNombre(rs.getString("nombres"));

gestorEstudiante.setApellido(rs.getString("apellidos"));

gestorEstudiante.setCorreoElectronico(rs.getString("email"));
```

```
gestorEstudiante.setACTIVO(rs.getBoolean("activo"));

gestorEstudiante.setCOMPUTACION(rs.getBoolean("computacion"));

gestorEstudiante.setSISTEMAS(rs.getBoolean("sistemas"));

gestorEstudiante.setTelefono(rs.getString("numerodetelefono"));
                        conexionBD.CerrarConexion();
                        return gestorEstudiante;
                    }
                    else {
                        conexionBD.CerrarConexion();
                        pantallaPrincipal.advertenciaBarraEstado("El
estudiante con la cedula "+cedula+" no se consiguió en la base de
datos.");
                        return null;
                    }
                    }   catch (SQLException e) {
                            conexionBD.CerrarConexion();
                            pantallaPrincipal.errorBarraEstado("Error
al acceder a la base de datos");
                            return null;
                    }
    }

    //  Metodo que revisa toda la tabla prestamo buscando las
coincidencias
    // con la cedula de un estudiante para determinar cuantos libros
tiene
    // en su poder un estudiante.
    public int librosPrestadosAlEstudiante (String cedula){
        try {
            // Limpia la barra de estado de la pantalla principal
            pantallaPrincipal.borrarBarraEstado();
            // Inicia y abre la conexion con la base de datos
            conexionBD = new ConexionBD(pantallaPrincipal);
            conexionBD.AbrirConexion();
            conexionBD.Consulta("SELECT COUNT(*) FROM prestamo WHERE
nrocedulaestudiante ='"+cedula+"' AND prestamo.recibido='false'");
            rs = conexionBD.MostrarDatosConsulta();
            rs.next();
            int cuenta = rs.getInt("COUNT");
            conexionBD.CerrarConexion();
            return cuenta;
        } catch (SQLException ex) {
            conexionBD.CerrarConexion();

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            return -1;
        }
    }

    // Dada la cota de un ejemplar, este metodo retorna la informacion
    // de dicho ejemplar, si no lo consigue retorna el Gestor con
valor
    // null
    public GestorInfoEjemplar buscarEjemplar(String cota) {
        // limpia los mensajes de la barra de estado de la pantalla
principal
```

```
        pantallaPrincipal.borrarBarraEstado();
        gestorEjemplar = new GestorInfoEjemplar();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();

        conexionBD.Consulta("SELECT * FROM datosejemplares, ejemplar
WHERE datosejemplares.iddatos = ejemplar.iddatos AND
ejemplar.idejemplar = '"+cota+"'");
        rs = conexionBD.MostrarDatosConsulta();

        try {
            if (rs.next()!=false){
                gestorEjemplar.setIdEjemplar(cota);

gestorEjemplar.setNombreEjemplar(rs.getString("nombre"));
                gestorEjemplar.setArea(rs.getString("area"));
                gestorEjemplar.setLibro(rs.getBoolean("libro"));
                gestorEjemplar.setEdicion(rs.getString("edicion"));
                gestorEjemplar.setInterno(rs.getBoolean("interno"));

gestorEjemplar.setDisonible(rs.getBoolean("disponible"));
                gestorEjemplar.setAño(rs.getInt("anio"));
                gestorEjemplar.setResumen(rs.getString("resumen"));

                String autores = rs.getString("autores");
                String REGEX = "!!!";
                Pattern p = Pattern.compile(REGEX);
                String a[] = p.split(autores) ;
                gestorEjemplar.setAutores(a[0].trim(), a[1].trim(),
a[2].trim());
                conexionBD.CerrarConexion();
                return gestorEjemplar;
            }
            else{
                conexionBD.CerrarConexion();
                pantallaPrincipal.advertenciaBarraEstado("La cota #
"+cota+" no se consigue en el sistema.");
                return null;
            }
        } catch ( SQLException ex) {
            conexionBD.CerrarConexion();
            pantallaPrincipal.errorBarraEstado("Error al buscar en la
base de datos.");

Logger.getLogger(GestorInfoEjemplar.class.getName()).log(Level.SEVERE,
null, ex);
            return null;
        }
    }

    // Con el reporte de prestamo de cada ejemplare y el total del
    // monto cancelado se genera la multa y se alamacenara en la base
    // de datos.
    public void generarMultas(ArrayList listaGestorInfoPrestamo,
boolean cancelada) {

        pantallaPrincipal.borrarBarraEstado();

        gestorEjemplar = new GestorInfoEjemplar();
        // Inicia y abre la conexion con la base de datos
```

```
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        for (int i=0;i<listaGestorInfoPrestamo.size();i++){
            if
(((GestorInfoPrestamo)listaGestorInfoPrestamo.get(i)).getDiaAtraso()>0
){
                conexionBD.Consulta("SELECT idprestamo FROM prestamo
WHERE " +

"idejemplar='"+((GestorInfoPrestamo)listaGestorInfoPrestamo.get(i)).ge
tIdEjemplar()+"' AND "+

"nrocedulaestudiante='"+((GestorInfoPrestamo)listaGestorInfoPrestamo.g
et(i)).getCedulaEstudiante()+"' AND "+
                "recibido='false'");
                System.out.println("LONGITUD LISTA
"+listaGestorInfoPrestamo.size());
                rs = conexionBD.MostrarDatosConsulta();
                try {
                    if (rs.next()) {
                        String idPrestamo;
                        double deuda;
                        if (cancelada)
                            deuda=0;
                        else
                            deuda=
((GestorInfoPrestamo)listaGestorInfoPrestamo.get(i)).getDeuda();
                        idPrestamo = rs.getString("idprestamo");

conexionBD.CargarDatos("multa(nrocedulaestudiante,idprestamo,monto,can
celada,diasatraso)",
                                                "
'"+((GestorInfoPrestamo)listaGestorInfoPrestamo.get(i)).getCedulaEstud
iante()+"',"+
                                                " '"+idPrestamo+"',"+
                                                " '"+deuda+"',"+
                                                " '"+cancelada+"',"+
                                                "
'"+((GestorInfoPrestamo)listaGestorInfoPrestamo.get(i)).getDiaAtraso()
+"'");
                    }// end if

                } catch (SQLException ex) {

Logger.getLogger(GestorRecibirEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
                    pantallaPrincipal.errorBarraEstado("Error al
accesar a la base de datos.");
                    conexionBD.CerrarConexion();
                }
        }
    }

        conexionBD.CerrarConexion();
        return;
    }

    // Metodo que dado un lista de cota de ejempalres recibidos
    // busca dichos ejemplares en le tabla ejemplar y activa la opcion
    // disponibles para indicar que estan disponibles para ser
    // prestandos
```

```
    public void marcarEjemplaresDisponibles(ArrayList
listaGestorInfoPrestamo) {
        // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        gestorEjemplar = new GestorInfoEjemplar();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        for (int i=0;i<listaGestorInfoPrestamo.size();i++)
            if (!conexionBD.Actualizar("UPDATE ejemplar SET disponible
= 'true' WHERE idejemplar= '"

+((GestorInfoPrestamo)listaGestorInfoPrestamo.get(i)).getIdEjemplar()+
"'")){
                pantallaPrincipal.errorBarraEstado("Hubo uno o mas
errores al recibir los ejemplares, por favor verifique que no hayan
sido eliminados de la base de datos.");
            }
        conexionBD.CerrarConexion();
        return ;
    }

    // Metodo que recibe un lista de los prestamos que se estan
recibiendo
    // y dada la informacion de estos revisa la tabla prestamo
buscandolos
    // y marcando la opcion recibido como true, indicando que el
prestamo
    // de dicho ejemplar ha sido recibido.
    void marcarPrestamosRecibidos(ArrayList listaGestorInfoPrestamo) {
        // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        gestorEjemplar = new GestorInfoEjemplar();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        for (int i=0;i<listaGestorInfoPrestamo.size();i++)
            if (!conexionBD.Actualizar("UPDATE prestamo SET recibido =
'true' WHERE idejemplar= '"

+((GestorInfoPrestamo)listaGestorInfoPrestamo.get(i)).getIdEjemplar()
                +"' AND recibido = 'false'")){
                pantallaPrincipal.errorBarraEstado("Hubo uno o mas
errores al recibir los ejemplares, por favor verifique que no hayan
sido eliminados de la base de datos.");
            }
        conexionBD.CerrarConexion();
        return ;
    }

    void actualizarEstadoSolvente (String cedula){
         // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        gestorEjemplar = new GestorInfoEjemplar();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
```

```
        if (conexionBD.Actualizar("UPDATE estudiantes set solvente =
false where numerocedula = '"+cedula+"' AND '"+cedula+"' IN("+
                                  "SELECT DISTINCT e.numerocedula
FROM estudiantes AS e, multa AS m, prestamo AS p  WHERE"+
                                  "(
(e.numerocedula=p.nrocedulaestudiante OR
e.numerocedula=m.nrocedulaestudiante) AND "+
                                  "(p.recibido=false OR
p.fechaentrega <(SELECT current_date)  OR m.cancelada=false)))"))
            //pantallaPrincipal.advertenciaBarraEstado("El estudiante
"+cedula+" NO esta solvente.");

        if (conexionBD.Actualizar("UPDATE estudiantes set solvente =
true WHERE numerocedula='"+cedula+"' AND '"+cedula+"' IN "+
                                    "( SELECT DISTINCT
e.numerocedula FROM estudiantes AS e, multa AS m, prestamo AS p  "+
                                    "WHERE
(e.numerocedula=p.nrocedulaestudiante AND (p.recibido=true OR
p.fechaentrega "+
                                    ">=(SELECT current_date)) AND
e.numerocedula NOT IN (SELECT m.nrocedulaestudiante "+
                                    "FROM multa AS m WHERE
m.cancelada=false)))"))
            //pantallaPrincipal.advertenciaBarraEstado("El estudiante
"+cedula+" esta solvente.");

        conexionBD.CerrarConexion();
        return ;
    }



}
```

5.3.2.2.5 GUIReportesPrestamos

```
package GestionPrestamos;

import Principal.GUIView;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.event.KeyEvent;
import java.util.ArrayList;
import java.util.Calendar;
import javax.swing.ImageIcon;
import javax.swing.table.DefaultTableModel;
import procesamiento.CellRenderPersonalizado;


public class GUIreportesPrestamos extends javax.swing.JPanel {
    private GUIView pantallaPrincipal;
    Calendar calendario;
    private String path;
```

```
    private ImageIcon imagenFondo;
    public GUIreportesPrestamos(GUIView pantallaPrincipal) {
        initComponents();
        this.pantallaPrincipal = pantallaPrincipal;

jtTablaPrestamos.setDefaultRenderer(jtTablaPrestamos.getColumnClass(0)
, new CellRenderPersonalizado(5));
        calendario = Calendar.getInstance();

jcbDias.setSelectedIndex(calendario.get(Calendar.DAY_OF_MONTH)-1);
        jcbMes.setSelectedIndex(calendario.get(Calendar.MONTH));
        jcbAño.setSelectedIndex(calendario.get(Calendar.YEAR)-2009);
        cargarIconos();
    }

    private void cargarIconos() {
        imagenFondo = new ImageIcon
(GUIView.class.getResource("resources/fondo.png"));
    }

    // Metodo para dibujar una imagen de fondo en el jPanel del
formulario
    // que se esta desplegando
    @Override
    public void paintComponent(Graphics g)
     {
        Graphics2D g2 = (Graphics2D)g;

g2.drawImage(imagenFondo.getImage(),0,0,this.getWidth(),this.getHeight
(),this);

    }


    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        buttonGroup1 = new javax.swing.ButtonGroup();
        buttonGroup2 = new javax.swing.ButtonGroup();
        jPanel1 = new javax.swing.JPanel();
        jcbDias = new javax.swing.JComboBox();
        jcbMes = new javax.swing.JComboBox();
        jcbAño = new javax.swing.JComboBox();
        jButton1 = new javax.swing.JButton();
        jPanel2 = new javax.swing.JPanel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jtTablaPrestamos = new javax.swing.JTable();
        jButton2 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        exportarPDF = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();
        jButton5 = new javax.swing.JButton();
        jPanel3 = new javax.swing.JPanel();
        jtfCotaCedula = new javax.swing.JTextField();
        jButton6 = new javax.swing.JButton();
        jPanel4 = new javax.swing.JPanel();
        jcbCota = new javax.swing.JCheckBox();
        jcbCédula = new javax.swing.JCheckBox();
        jcbTodos = new javax.swing.JCheckBox();
        jcbRecibidos = new javax.swing.JCheckBox();
```

```
        jcbNoRecibidos = new javax.swing.JCheckBox();

        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(Principal.GUIApp.clas
s).getContext().getResourceMap(GUIreportesPrestamos.class);
        setBorder(javax.swing.BorderFactory.createTitledBorder(null,
resourceMap.getString("Form.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("Form.border.titleFont"))); // NOI18N
        setName("Form"); // NOI18N
        setOpaque(false);

        jPanel1.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel1.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel1.border.titleFont"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N
        jPanel1.setOpaque(false);

        jcbDias.setModel(new javax.swing.DefaultComboBoxModel(new
String[] { "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11",
"12", "13", "14", "15", "16", "17", "18", "19", "20", "21", "22",
"23", "24", "25", "26", "27", "28", "29", "30", "31" }));
        jcbDias.setName("jcbDias"); // NOI18N

        jcbMes.setModel(new javax.swing.DefaultComboBoxModel(new
String[] { "Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio",
"Julio", "Agosto", "Septiembre", "Octubre", "Noviembre", "Diciembre"
}));
        jcbMes.setName("jcbMes"); // NOI18N

        jcbAño.setModel(new javax.swing.DefaultComboBoxModel(new
String[] { "2009", "2010", "2011", "2012", "2013", "2014", "2015",
"2016", "2017", "2018", "2019", "2020", "2021", "2022", "2023",
"2024", "2025", "2026", "2027", "2028", "2029", "2030" }));
        jcbAño.setName("jcbAño"); // NOI18N

        jButton1.setIcon(resourceMap.getIcon("jButton1.icon")); //
NOI18N
        jButton1.setText(resourceMap.getString("jButton1.text")); //
NOI18N

jButton1.setToolTipText(resourceMap.getString("jButton1.toolTipText"))
; // NOI18N
        jButton1.setName("jButton1"); // NOI18N
        jButton1.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton1ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
```

```
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jcbDias,
javax.swing.GroupLayout.PREFERRED_SIZE, 37,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jcbMes,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jcbAño,
javax.swing.GroupLayout.PREFERRED_SIZE, 74,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jButton1)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(jcbDias,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jcbMes,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jcbAño,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton1))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        jPanel2.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel2.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel2.border.titleFont"))); // NOI18N
        jPanel2.setName("jPanel2"); // NOI18N
        jPanel2.setOpaque(false);
```

```
jScrollPane1.setHorizontalScrollBarPolicy(javax.swing.ScrollPaneConsta
nts.HORIZONTAL_SCROLLBAR_NEVER);
        jScrollPane1.setName("jScrollPane1"); // NOI18N

        jtTablaPrestamos.setModel(new
javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Ced. estudiante", "Ejemplar", "Fecha prestamo",
"Fecha recepción", "Prestamo", "Estado"
            }
        ) {
            Class[] types = new Class [] {
                java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class
            };
            boolean[] canEdit = new boolean [] {
                false, false, false, false, false, false
            };

            public Class getColumnClass(int columnIndex) {
                return types [columnIndex];
            }

            public boolean isCellEditable(int rowIndex, int
columnIndex) {
                return canEdit [columnIndex];
            }
        });
        jtTablaPrestamos.setName("jtTablaPrestamos"); // NOI18N

jtTablaPrestamos.setSelectionMode(javax.swing.ListSelectionModel.SINGL
E_SELECTION);
        jScrollPane1.setViewportView(jtTablaPrestamos);

        jButton2.setIcon(resourceMap.getIcon("jButton2.icon")); //
NOI18N
        jButton2.setText(resourceMap.getString("jButton2.text")); //
NOI18N

jButton2.setToolTipText(resourceMap.getString("jButton2.toolTipText"))
; // NOI18N
        jButton2.setName("jButton2"); // NOI18N
        jButton2.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton2ActionPerformed(evt);
            }
        });

        jButton3.setIcon(resourceMap.getIcon("jButton3.icon")); //
NOI18N
        jButton3.setText(resourceMap.getString("jButton3.text")); //
NOI18N

jButton3.setToolTipText(resourceMap.getString("jButton3.toolTipText"))
```

```
; // NOI18N
        jButton3.setName("jButton3"); // NOI18N
        jButton3.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton3ActionPerformed(evt);
            }
        });

        exportarPDF.setIcon(resourceMap.getIcon("exportarPDF.icon"));
// NOI18N

exportarPDF.setText(resourceMap.getString("exportarPDF.text")); //
NOI18N
        exportarPDF.setName("exportarPDF"); // NOI18N
        exportarPDF.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                exportarPDFActionPerformed(evt);
            }
        });

        jLabel1.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N

jLabel1.setForeground(resourceMap.getColor("jLabel1.foreground")); //
NOI18N

jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText(resourceMap.getString("jLabel1.text")); //
NOI18N
        jLabel1.setName("jLabel1"); // NOI18N

        javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
jPanel2Layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jLabel1,
javax.swing.GroupLayout.DEFAULT_SIZE, 778, Short.MAX_VALUE))
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addGap(339, 339, 339)
                        .addComponent(jButton2)
                        .addGap(18, 18, 18)
                        .addComponent(jButton3)
                        .addGap(18, 18, 18)
                        .addComponent(exportarPDF,
javax.swing.GroupLayout.DEFAULT_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
jPanel2Layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 778, Short.MAX_VALUE)))
                .addContainerGap())
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 27,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 230, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(exportarPDF)
                    .addComponent(jButton2)
                    .addComponent(jButton3))
                .addContainerGap())
        );

        jButton5.setIcon(resourceMap.getIcon("jButton5.icon")); //
NOI18N
        jButton5.setText(resourceMap.getString("jButton5.text")); //
NOI18N
        jButton5.setName("jButton5"); // NOI18N
        jButton5.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton5ActionPerformed(evt);
            }
        });


jPanel3.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel3.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel3.border.titleFont"))); // NOI18N
        jPanel3.setName("jPanel3"); // NOI18N
        jPanel3.setOpaque(false);


jtfCotaCedula.setText(resourceMap.getString("jtfCotaCedula.text")); //
NOI18N
        jtfCotaCedula.setName("jtfCotaCedula"); // NOI18N
        jtfCotaCedula.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyPressed(java.awt.event.KeyEvent evt) {
```

```
                jtfCotaCedulaKeyPressed(evt);
            }
        });

        jButton6.setIcon(resourceMap.getIcon("jButton6.icon")); //
NOI18N
        jButton6.setText(resourceMap.getString("jButton6.text")); //
NOI18N

jButton6.setToolTipText(resourceMap.getString("jButton6.toolTipText"))
; // NOI18N
        jButton6.setName("jButton6"); // NOI18N
        jButton6.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton6ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel3Layout = new
javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jtfCotaCedula,
javax.swing.GroupLayout.PREFERRED_SIZE, 109,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jButton6)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(jtfCotaCedula,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton6))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        jPanel4.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel4.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
```

```
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel4.border.titleFont"))); // NOI18N
        jPanel4.setName("jPanel4"); // NOI18N
        jPanel4.setOpaque(false);

        jcbCota.setBackground(new java.awt.Color(0x00ffffff, true ));
        buttonGroup2.add(jcbCota);
        jcbCota.setFont(resourceMap.getFont("jcbCota.font")); //
NOI18N
        jcbCota.setSelected(true);
        jcbCota.setText(resourceMap.getString("jcbCota.text")); //
NOI18N
        jcbCota.setName("jcbCota"); // NOI18N
        jcbCota.setOpaque(false);
        jcbCota.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jcbCotaActionPerformed(evt);
            }
        });

        jcbCédula.setBackground(new java.awt.Color(0x00ffffff, true
));
        buttonGroup2.add(jcbCédula);
        jcbCédula.setFont(resourceMap.getFont("jcbCédula.font")); //
NOI18N
        jcbCédula.setText(resourceMap.getString("jcbCédula.text")); //
NOI18N
        jcbCédula.setName("jcbCédula"); // NOI18N
        jcbCédula.setOpaque(false);

        jcbTodos.setBackground(new java.awt.Color(0x00ffffff, true ));
        jcbTodos.setFont(resourceMap.getFont("jcbTodos.font")); //
NOI18N
        jcbTodos.setSelected(true);
        jcbTodos.setText(resourceMap.getString("jcbTodos.text")); //
NOI18N
        jcbTodos.setName("jcbTodos"); // NOI18N
        jcbTodos.setOpaque(false);
        jcbTodos.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jcbTodosActionPerformed(evt);
            }
        });

        jcbRecibidos.setBackground(new java.awt.Color(0x00ffffff, true
));
        buttonGroup1.add(jcbRecibidos);
        jcbRecibidos.setFont(resourceMap.getFont("jCheckBox3.font"));
// NOI18N

jcbRecibidos.setText(resourceMap.getString("jcbRecibidos.text")); //
NOI18N
        jcbRecibidos.setEnabled(false);
        jcbRecibidos.setName("jcbRecibidos"); // NOI18N
        jcbRecibidos.setOpaque(false);

        jcbNoRecibidos.setBackground(new java.awt.Color(0x00ffffff,
```

```
true ));
        buttonGroup1.add(jcbNoRecibidos);

jcbNoRecibidos.setFont(resourceMap.getFont("jcbNoRecibidos.font")); //
NOI18N

jcbNoRecibidos.setText(resourceMap.getString("jcbNoRecibidos.text"));
// NOI18N
        jcbNoRecibidos.setEnabled(false);
        jcbNoRecibidos.setName("jcbNoRecibidos"); // NOI18N
        jcbNoRecibidos.setOpaque(false);

        javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4);
        jPanel4.setLayout(jPanel4Layout);
        jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel4Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jcbCota,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jcbCédula,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(58, 58, 58)
                .addComponent(jcbTodos,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jcbRecibidos,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jcbNoRecibidos,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(53, 53, 53))
        );
        jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel4Layout.createSequentialGroup()

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(jcbCota)
                    .addComponent(jcbCédula)
                    .addComponent(jcbTodos)
                    .addComponent(jcbRecibidos)
                    .addComponent(jcbNoRecibidos))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
```

```
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(jPanel2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addContainerGap())
                        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.TRAILING, false)
                                .addComponent(jPanel4,
javax.swing.GroupLayout.Alignment.LEADING, 0, 627, Short.MAX_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup()
                                    .addComponent(jPanel3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                    .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                            .addGap(160, 160, 160)))
                    .addComponent(jButton5,
javax.swing.GroupLayout.Alignment.TRAILING)))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING, false)
                    .addComponent(jPanel3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel4,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGap(11, 11, 11)
                .addComponent(jButton5))
        );
    }// </editor-fold>

    private void jcbTodosActionPerformed(java.awt.event.ActionEvent
evt) {
        if (jcbTodos.isSelected()){
            jcbRecibidos.setEnabled(false);
            jcbNoRecibidos.setEnabled(false);
        }
        else {
            jcbRecibidos.setEnabled(true);
            jcbNoRecibidos.setEnabled(true);
        }

    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {
        consultarPrestamo ();
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent
evt) {
        if (jtTablaPrestamos.getSelectedRow()!=-1){

pantallaPrincipal.buscarEstudiante(Long.parseLong(jtTablaPrestamos.get
Model().getValueAt(jtTablaPrestamos.getSelectedRow(),0).toString()));
        }

    }

    private void jButton5ActionPerformed(java.awt.event.ActionEvent
evt) {
        pantallaPrincipal.home();
    }

    private void jButton6ActionPerformed(java.awt.event.ActionEvent
evt) {
        consultarCotaCedula ();
    }

    private void jcbCotaActionPerformed(java.awt.event.ActionEvent
evt) {
        // TODO add your handling code here:
}

    private void jtfCotaCedulaKeyPressed(java.awt.event.KeyEvent evt)
{
        if (evt.getKeyCode()== KeyEvent.VK_ENTER)
           consultarCotaCedula ();
    }
```

```
    private void jButton3ActionPerformed(java.awt.event.ActionEvent
evt) {
        consultarEjemplar();
    }

    private void exportarPDFActionPerformed(java.awt.event.ActionEvent
evt) {
        exportarPDF();
    }


    // Variables declaration - do not modify
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.ButtonGroup buttonGroup2;
    private javax.swing.JButton exportarPDF;
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton5;
    private javax.swing.JButton jButton6;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JComboBox jcbAño;
    private javax.swing.JCheckBox jcbCota;
    private javax.swing.JCheckBox jcbCédula;
    private javax.swing.JComboBox jcbDias;
    private javax.swing.JComboBox jcbMes;
    private javax.swing.JCheckBox jcbNoRecibidos;
    private javax.swing.JCheckBox jcbRecibidos;
    private javax.swing.JCheckBox jcbTodos;
    private javax.swing.JTable jtTablaPrestamos;
    private javax.swing.JTextField jtfCotaCedula;
    // End of variables declaration

    private GestorReportesPrestamos gestorReportesPestamos;
    private PrestamosExportarPDF prestamosPDF;
    private boolean POR_FECHA=false;
    private String tituloPDF;
     private void exportarPDF() {
         prestamosPDF = new PrestamosExportarPDF (pantallaPrincipal);
         if (POR_FECHA){
             if (jcbTodos.isSelected()){
                 tituloPDF = "Lista de préstamos para la fecha
"+jcbDias.getSelectedItem()+
                             " de "+jcbMes.getSelectedItem()+" de
"+jcbAño.getSelectedItem();

prestamosPDF.exportarReportesPorFecha(jtTablaPrestamos, tituloPDF);
             }
             else
                 if (jcbRecibidos.isSelected()){
                     tituloPDF = "Lista de préstamos recibidos para la
fecha "+jcbDias.getSelectedItem()+
                             " de "+jcbMes.getSelectedItem()+" de
"+jcbAño.getSelectedItem();
```

```
prestamosPDF.exportarReportesPorFecha(jtTablaPrestamos, tituloPDF);
                }
                else{
                    tituloPDF = "Lista de préstamos no recibidos para
la fecha "+jcbDias.getSelectedItem()+
                                " de "+jcbMes.getSelectedItem()+" de
"+jcbAño.getSelectedItem();

prestamosPDF.exportarReportesPorFecha(jtTablaPrestamos, tituloPDF);
                }
         }//end if
         else {
             if (jcbCota.isSelected()){
                    tituloPDF = "Lista de préstamos para el ejemplar
"+jtfCotaCedula.getText();

prestamosPDF.exportarReportesPorFecha(jtTablaPrestamos, tituloPDF);
             }
             else {
                    tituloPDF = "Lista de préstamos para el estudiante
"+jtfCotaCedula.getText();

prestamosPDF.exportarReportesPorFecha(jtTablaPrestamos, tituloPDF);
             }
         }
         System.out.println("CADENA= "+tituloPDF );


    }

    public void consultarPrestamoCota (String cota){
        jtfCotaCedula.setText(cota.trim());
        jcbCota.setSelected(true);
        consultarCotaCedula ();
    }

    private void consultarEjemplar(){
      pantallaPrincipal.borrarBarraEstado();
        if (jtTablaPrestamos.getSelectedRow()==-1)
            pantallaPrincipal.advertenciaBarraEstado("Debe seleccionar
un ejemplar para consultar.");
        else

pantallaPrincipal.buscarEjemplar(jtTablaPrestamos.getModel().getValueA
t(jtTablaPrestamos.getSelectedRow(), 1).toString());
    }

    private void consultarPrestamo() {
        POR_FECHA=true;
        gestorReportesPestamos = new GestorReportesPrestamos
(pantallaPrincipal);
        limpiarTabla ();
        String fecha = jcbAño.getSelectedItem().toString()+"-"+
                       (jcbMes.getSelectedIndex()+1)+"-"+
                       (jcbDias.getSelectedIndex()+1);
        if (jcbTodos.isSelected())

gestorReportesPestamos.consultarPrestamos(jtTablaPrestamos, fecha);
        else

gestorReportesPestamos.consultarPrestamos(jtTablaPrestamos,
```

```
fecha,jcbRecibidos.isSelected());
    }

    private void consultarCotaCedula (){
        POR_FECHA = false;
        gestorReportesPestamos = new GestorReportesPrestamos
(pantallaPrincipal);
        limpiarTabla ();
        String cadena = jtfCotaCedula.getText().trim();
        if (jcbCota.isSelected())
            if (jcbTodos.isSelected())

gestorReportesPestamos.consultarPrestamosCota(jtTablaPrestamos,
cadena);
            else

gestorReportesPestamos.consultarPrestamosCota(jtTablaPrestamos,
cadena,jcbRecibidos.isSelected());
        else
            if (jcbTodos.isSelected())

gestorReportesPestamos.consultarPrestamosCedula(jtTablaPrestamos,
cadena);
            else

gestorReportesPestamos.consultarPrestamosCedula(jtTablaPrestamos,
cadena,jcbRecibidos.isSelected());



    }


    private void limpiarTabla (){
        int columnas =
((DefaultTableModel)jtTablaPrestamos.getModel()).getRowCount();
        if (columnas >0){
            for (int i=0;i<columnas;i++)

((DefaultTableModel)jtTablaPrestamos.getModel()).removeRow(0);
        }
    }

}
```

5.3.2.2.6 GestorReportesPrestamos

```
package GestionPrestamos;

import BaseDeDatos.ConexionBD;
import Principal.GUIView;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.logging.Level;
```

```
import java.util.logging.Logger;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;

public class GestorReportesPrestamos {
    private GUIView pantallaPrincipal;
    private ConexionBD conexionBD;
    private ResultSet rs = null;
    private GestorInfoPrestamo gestorInfoPrestamo;
    private boolean recibido;
    private String recibidoTexto;
    private String internoTexto;

    GestorReportesPrestamos(GUIView pantallaPrincipal) {
        this.pantallaPrincipal = pantallaPrincipal;
    }


    public boolean consultarPrestamos (JTable listaPrestamos, String
fecha){
         // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        gestorInfoPrestamo = new GestorInfoPrestamo();
        conexionBD.Consulta("SELECT * FROM prestamo WHERE
fecha='"+fecha+"'");

        DefaultTableModel tablaTemporal = (DefaultTableModel)
listaPrestamos.getModel();
        rs = conexionBD.MostrarDatosConsulta();

        try {
            while (rs.next()!=false){

gestorInfoPrestamo.setCeduladEstudiante(rs.getString("nrocedulaestudia
nte"));

gestorInfoPrestamo.setIdEjemplar(rs.getString("idejemplar"));

gestorInfoPrestamo.setIdAyudante(rs.getString("idayudante"));
                gestorInfoPrestamo.setFecha(rs.getString("fecha"));

gestorInfoPrestamo.setFechaEntrega(rs.getString("fechaentrega"));

gestorInfoPrestamo.setPrestamoINTERNO(rs.getBoolean("interno"));
                recibido = rs.getBoolean("recibido");

                if (recibido)
                    recibidoTexto = "RECIBIDO";
                else
                    recibidoTexto= "NO RECIBIDO";

                if (gestorInfoPrestamo.getPrestamoINTERNO())
                    internoTexto = "INTERNO";
                else
                    internoTexto= "CIRCULANTE";
                Object[] nuevo =
{gestorInfoPrestamo.getCedulaEstudiante(),
```

```
                                    gestorInfoPrestamo.getIdEjemplar(),
                                    convertirFecha
(gestorInfoPrestamo.getFecha()),
                                    convertirFecha
(gestorInfoPrestamo.getFechaEntrega()),
                                    recibidoTexto,
                                    internoTexto,
                                    };
                   tablaTemporal.addRow(nuevo);
               }
               conexionBD.CerrarConexion();
               if (listaPrestamos.getRowCount()==0){
                    pantallaPrincipal.advertenciaBarraEstado("No se
consiguió ningun prestamo para la fecha señalada.");
                    return false;
               }
               else
                    return true;
           } catch ( SQLException ex) {
               conexionBD.CerrarConexion();
               pantallaPrincipal.errorBarraEstado("Error al buscar en la
base de datos.");

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
               return false;
           }
       }



       public boolean consultarPrestamos(JTable listaPrestamos, String
fecha, boolean selected) {
           // limpia los mensajes de la barra de estado de la pantalla
principal
           pantallaPrincipal.borrarBarraEstado();
           // Inicia y abre la conexion con la base de datos
           conexionBD = new ConexionBD(pantallaPrincipal);
           conexionBD.AbrirConexion();
           gestorInfoPrestamo = new GestorInfoPrestamo();
           conexionBD.Consulta("SELECT * FROM prestamo WHERE
fecha='"+fecha+"' AND recibido='"+selected+"'");
           DefaultTableModel tablaTemporal = (DefaultTableModel)
listaPrestamos.getModel();
           rs = conexionBD.MostrarDatosConsulta();

           try {
               while (rs.next()!=false){

gestorInfoPrestamo.setCeduladEstudiante(rs.getString("nrocedulaestudia
nte"));

gestorInfoPrestamo.setIdEjemplar(rs.getString("idejemplar"));

gestorInfoPrestamo.setIdAyudante(rs.getString("idayudante"));
                    gestorInfoPrestamo.setFecha(rs.getString("fecha"));

gestorInfoPrestamo.setFechaEntrega(rs.getString("fechaentrega"));

gestorInfoPrestamo.setPrestamoINTERNO(rs.getBoolean("interno"));
                    recibido = rs.getBoolean("recibido");
```

```
                if (recibido)
                    recibidoTexto = "RECIBIDO";
                else
                    recibidoTexto= "NO RECIBIDO";

                if (gestorInfoPrestamo.getPrestamoINTERNO())
                    internoTexto = "INTERNO";
                else
                    internoTexto= "CIRCULANTE";
                Object[] nuevo =
{gestorInfoPrestamo.getCedulaEstudiante(),
                                    gestorInfoPrestamo.getIdEjemplar(),
                                    convertirFecha
(gestorInfoPrestamo.getFecha()),
                                    convertirFecha
(gestorInfoPrestamo.getFechaEntrega()),
                                    recibidoTexto,
                                    internoTexto,
                                    };
               tablaTemporal.addRow(nuevo);
            }
            conexionBD.CerrarConexion();
            if (listaPrestamos.getRowCount()==0){
                pantallaPrincipal.advertenciaBarraEstado("No se
consiguió ningun prestamo para la fecha señalada.");
                return false;
            }
            else
                return true;
        } catch ( SQLException ex) {
            conexionBD.CerrarConexion();
            pantallaPrincipal.errorBarraEstado("Error al buscar en la
base de datos.");

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            return false;
        }
    }

    boolean consultarPrestamosCedula(JTable jtTablaPrestamos, String
cadena) {
         // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        gestorInfoPrestamo = new GestorInfoPrestamo();
        conexionBD.Consulta("SELECT * FROM prestamo WHERE
nrocedulaestudiante='"+cadena+"'");

        DefaultTableModel tablaTemporal = (DefaultTableModel)
jtTablaPrestamos.getModel();
        rs = conexionBD.MostrarDatosConsulta();

        try {
            while (rs.next()!=false){

gestorInfoPrestamo.setCeduladEstudiante(rs.getString("nrocedulaestudia
```

```
nte"));

gestorInfoPrestamo.setIdEjemplar(rs.getString("idejemplar"));

gestorInfoPrestamo.setIdAyudante(rs.getString("idayudante"));
                gestorInfoPrestamo.setFecha(rs.getString("fecha"));

gestorInfoPrestamo.setFechaEntrega(rs.getString("fechaentrega"));

gestorInfoPrestamo.setPrestamoINTERNO(rs.getBoolean("interno"));
                recibido = rs.getBoolean("recibido");

                if (recibido)
                    recibidoTexto = "RECIBIDO";
                else
                    recibidoTexto= "NO RECIBIDO";

                if (gestorInfoPrestamo.getPrestamoINTERNO())
                    internoTexto = "INTERNO";
                else
                    internoTexto= "CIRCULANTE";
                Object[] nuevo =
{gestorInfoPrestamo.getCedulaEstudiante(),
                                    gestorInfoPrestamo.getIdEjemplar(),
                                    convertirFecha
(gestorInfoPrestamo.getFecha()),
                                    convertirFecha
(gestorInfoPrestamo.getFechaEntrega()),
                                    recibidoTexto,
                                    internoTexto,
                                    };
               tablaTemporal.addRow(nuevo);
            }
            conexionBD.CerrarConexion();
            if (jtTablaPrestamos.getRowCount()==0){
                pantallaPrincipal.advertenciaBarraEstado("No se
consiguió ningun prestamo para la cédula señalada.");
                return false;
            }
            else
                return true;
        } catch ( SQLException ex) {
            conexionBD.CerrarConexion();
            pantallaPrincipal.errorBarraEstado("Error al buscar en la
base de datos.");

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            return false;
        }
    }

    boolean consultarPrestamosCota(JTable jtTablaPrestamos, String
cadena) {
        // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        gestorInfoPrestamo = new GestorInfoPrestamo();
```

```
        conexionBD.Consulta("SELECT * FROM prestamo WHERE
idejemplar='"+cadena+"'");

        DefaultTableModel tablaTemporal = (DefaultTableModel)
jtTablaPrestamos.getModel();
        rs = conexionBD.MostrarDatosConsulta();

        try {
            while (rs.next()!=false){

gestorInfoPrestamo.setCeduladEstudiante(rs.getString("nrocedulaestudia
nte"));

gestorInfoPrestamo.setIdEjemplar(rs.getString("idejemplar"));

gestorInfoPrestamo.setIdAyudante(rs.getString("idayudante"));
                gestorInfoPrestamo.setFecha(rs.getString("fecha"));

gestorInfoPrestamo.setFechaEntrega(rs.getString("fechaentrega"));

gestorInfoPrestamo.setPrestamoINTERNO(rs.getBoolean("interno"));
                recibido = rs.getBoolean("recibido");

                if (recibido)
                    recibidoTexto = "RECIBIDO";
                else
                    recibidoTexto= "NO RECIBIDO";

                if (gestorInfoPrestamo.getPrestamoINTERNO())
                    internoTexto = "INTERNO";
                else
                    internoTexto= "CIRCULANTE";
                Object[] nuevo =
{gestorInfoPrestamo.getCedulaEstudiante(),
                                    gestorInfoPrestamo.getIdEjemplar(),
                                    convertirFecha
(gestorInfoPrestamo.getFecha()),
                                    convertirFecha
(gestorInfoPrestamo.getFechaEntrega()),
                                    recibidoTexto,
                                    internoTexto,
                                    };
               tablaTemporal.addRow(nuevo);
            }
            conexionBD.CerrarConexion();
            if (jtTablaPrestamos.getRowCount()==0){
                pantallaPrincipal.advertenciaBarraEstado("No se
consiguió ningun prestamo para la cota señalada.");
                return false;
            }
            else
                return true;
        } catch ( SQLException ex) {
            conexionBD.CerrarConexion();
            pantallaPrincipal.errorBarraEstado("Error al buscar en la
base de datos.");

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            return false;
        }
```

```
    }

      boolean consultarPrestamosCedula(JTable jtTablaPrestamos, String
cadena, boolean recibido) {
         // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        gestorInfoPrestamo = new GestorInfoPrestamo();
        conexionBD.Consulta("SELECT * FROM prestamo WHERE
nrocedulaestudiante='"+cadena+"' AND recibido="+recibido);

        DefaultTableModel tablaTemporal = (DefaultTableModel)
jtTablaPrestamos.getModel();
        rs = conexionBD.MostrarDatosConsulta();

        try {
            while (rs.next()!=false){

gestorInfoPrestamo.setCeduladEstudiante(rs.getString("nrocedulaestudia
nte"));

gestorInfoPrestamo.setIdEjemplar(rs.getString("idejemplar"));

gestorInfoPrestamo.setIdAyudante(rs.getString("idayudante"));
                gestorInfoPrestamo.setFecha(rs.getString("fecha"));

gestorInfoPrestamo.setFechaEntrega(rs.getString("fechaentrega"));

gestorInfoPrestamo.setPrestamoINTERNO(rs.getBoolean("interno"));
                recibido = rs.getBoolean("recibido");

                if (recibido)
                    recibidoTexto = "RECIBIDO";
                else
                    recibidoTexto= "NO RECIBIDO";

                if (gestorInfoPrestamo.getPrestamoINTERNO())
                    internoTexto = "INTERNO";
                else
                    internoTexto= "CIRCULANTE";
                Object[] nuevo =
{gestorInfoPrestamo.getCedulaEstudiante(),
                                    gestorInfoPrestamo.getIdEjemplar(),
                                    convertirFecha
(gestorInfoPrestamo.getFecha()),
                                    convertirFecha
(gestorInfoPrestamo.getFechaEntrega()),
                                    recibidoTexto,
                                    internoTexto,
                                    };
               tablaTemporal.addRow(nuevo);
            }
            conexionBD.CerrarConexion();
            if (jtTablaPrestamos.getRowCount()==0){
                pantallaPrincipal.advertenciaBarraEstado("No se
consiguió ningun prestamo para la cédula señalada.");
                return false;
            }
```

```
            else
                return true;
        } catch ( SQLException ex) {
            conexionBD.CerrarConexion();
            pantallaPrincipal.errorBarraEstado("Error al buscar en la
base de datos.");

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
            return false;
        }
    }

    boolean consultarPrestamosCota(JTable jtTablaPrestamos, String
cadena, boolean recibido) {
        // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();
        gestorInfoPrestamo = new GestorInfoPrestamo();
        conexionBD.Consulta("SELECT * FROM prestamo WHERE
idejemplar='"+cadena+"' AND recibido="+recibido);

        DefaultTableModel tablaTemporal = (DefaultTableModel)
jtTablaPrestamos.getModel();
        rs = conexionBD.MostrarDatosConsulta();

        try {
            while (rs.next()!=false){

gestorInfoPrestamo.setCeduladEstudiante(rs.getString("nrocedulaestudia
nte"));

gestorInfoPrestamo.setIdEjemplar(rs.getString("idejemplar"));

gestorInfoPrestamo.setIdAyudante(rs.getString("idayudante"));
                gestorInfoPrestamo.setFecha(rs.getString("fecha"));

gestorInfoPrestamo.setFechaEntrega(rs.getString("fechaentrega"));

gestorInfoPrestamo.setPrestamoINTERNO(rs.getBoolean("interno"));
                recibido = rs.getBoolean("recibido");

                if (recibido)
                    recibidoTexto = "RECIBIDO";
                else
                    recibidoTexto= "NO RECIBIDO";

                if (gestorInfoPrestamo.getPrestamoINTERNO())
                    internoTexto = "INTERNO";
                else
                    internoTexto= "CIRCULANTE";
                Object[] nuevo =
{gestorInfoPrestamo.getCedulaEstudiante(),
                                    gestorInfoPrestamo.getIdEjemplar(),
                                    convertirFecha
(gestorInfoPrestamo.getFecha()),
                                    convertirFecha
(gestorInfoPrestamo.getFechaEntrega()),
```

```
                                    recibidoTexto,
                                    internoTexto,
                                    };
                   tablaTemporal.addRow(nuevo);
               }
               conexionBD.CerrarConexion();
               if (jtTablaPrestamos.getRowCount()==0){
                    pantallaPrincipal.advertenciaBarraEstado("No se
consiguió ningun prestamo para la cota señalada.");
                   return false;
               }
               else
                   return true;
           } catch ( SQLException ex) {
               conexionBD.CerrarConexion();
               pantallaPrincipal.errorBarraEstado("Error al buscar en la
base de datos.");

Logger.getLogger(GestorPrestarEjemplar.class.getName()).log(Level.SEVE
RE, null, ex);
               return false;
           }
       }
       private String convertirFecha(String fecha){
           String separa[] = fecha.split("-");
           return (separa[2]+"/"+separa[1]+"/"+separa[0]);
       }

}
```

5.3.2.2.7 GUIVariablesSistema

```
package GestionPrestamos;

import Principal.GUIView;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.io.File;
import java.util.regex.Matcher;
import javax.swing.ImageIcon;


public class GUIvariablesSistema extends javax.swing.JPanel {
    private ImageIcon imagenFondo;
    private GestorVariablesSistema gestorVariablesSistema;
    private GUIView pantallaPrincipal;
    /** Creates new form GUIvariablesSistema */
    public GUIvariablesSistema(GUIView pantallaPrincipal) {
        initComponents();
        cargarIconos();
        this.pantallaPrincipal = pantallaPrincipal;
        gestorVariablesSistema = new
GestorVariablesSistema(pantallaPrincipal);
        cargarValores();
        cargarIconos();
    }
```

```
    private void cargarIconos() {
        imagenFondo = new ImageIcon
(GUIView.class.getResource("resources/fondo.png"));
    }

    // Metodo para dibujar una imagen de fondo en el jPanel del
formulario
    // que se esta desplegando
    @Override
    public void paintComponent(Graphics g)
     {
        Graphics2D g2 = (Graphics2D)g;

g2.drawImage(imagenFondo.getImage(),0,0,this.getWidth(),this.getHeight
(),this);

    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jFileChooser = new javax.swing.JFileChooser();
        jFileChooser1 = new javax.swing.JFileChooser();
        jPanel1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        jPanel3 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jPanel4 = new javax.swing.JPanel();
        jsUnDiaRetraso = new javax.swing.JSpinner();
        jsDosDiasRetraso = new javax.swing.JSpinner();
        jsDiaExtraRetraso = new javax.swing.JSpinner();
        jPanel5 = new javax.swing.JPanel();
        jPanel6 = new javax.swing.JPanel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jPanel7 = new javax.swing.JPanel();
        jsDuracionPrestamo = new javax.swing.JSpinner();
        jsEjemplaresPorPersona = new javax.swing.JSpinner();
        jcbPrestamoTesis = new javax.swing.JComboBox();
        jtfMoneda = new javax.swing.JTextField();
        jPanel8 = new javax.swing.JPanel();
        jLabel7 = new javax.swing.JLabel();
        jtfDirHTML = new javax.swing.JTextField();
        jButton1 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        jButton4 = new javax.swing.JButton();
        jPanel9 = new javax.swing.JPanel();
        jButton5 = new javax.swing.JButton();
        jButton6 = new javax.swing.JButton();
        jtfDirHTML2 = new javax.swing.JTextField();
        jButton7 = new javax.swing.JButton();

        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(Principal.GUIApp.clas
s).getContext().getResourceMap(GUIvariablesSistema.class);
```

```
jFileChooser.setDialogTitle(resourceMap.getString("jFileChooser.dialog
Title")); // NOI18N

jFileChooser.setDialogType(javax.swing.JFileChooser.SAVE_DIALOG);

jFileChooser.setFileSelectionMode(javax.swing.JFileChooser.DIRECTORIES
_ONLY);
        jFileChooser.setName("jFileChooser"); // NOI18N
        jFileChooser.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jFileChooserActionPerformed(evt);
            }
        });

        jFileChooser1.setName("jFileChooser1"); // NOI18N

        setName("Form"); // NOI18N


jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(null,
resourceMap.getString("jPanel1.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel1.border.titleFont"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N
        jPanel1.setOpaque(false);

        jPanel2.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel2.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel2.border.titleFont"))); // NOI18N
        jPanel2.setName("jPanel2"); // NOI18N
        jPanel2.setOpaque(false);

        jPanel3.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel3.setName("jPanel3"); // NOI18N
        jPanel3.setOpaque(false);

        jLabel1.setFont(resourceMap.getFont("jLabel2.font")); //
NOI18N
        jLabel1.setText(resourceMap.getString("jLabel1.text")); //
NOI18N
        jLabel1.setName("jLabel1"); // NOI18N

        jLabel2.setFont(resourceMap.getFont("jLabel2.font")); //
NOI18N
        jLabel2.setText(resourceMap.getString("jLabel2.text")); //
NOI18N
        jLabel2.setName("jLabel2"); // NOI18N

        jLabel3.setFont(resourceMap.getFont("jLabel2.font")); //
NOI18N
        jLabel3.setText(resourceMap.getString("jLabel3.text")); //
NOI18N
```

```
        jLabel3.setName("jLabel3"); // NOI18N
        jLabel3.setPreferredSize(new java.awt.Dimension(135, 20));

        javax.swing.GroupLayout jPanel3Layout = new
javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addGroup(jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                            .addComponent(jLabel3,
javax.swing.GroupLayout.DEFAULT_SIZE, 216, Short.MAX_VALUE)
                            .addComponent(jLabel1))
                        .addContainerGap())
                    .addGroup(jPanel3Layout.createSequentialGroup()
                        .addComponent(jLabel2,
javax.swing.GroupLayout.DEFAULT_SIZE, 211, Short.MAX_VALUE)
                        .addGap(15, 15, 15))))
        );
        jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)
                .addGap(18, 18, 18)
                .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
16, Short.MAX_VALUE)
                .addComponent(jLabel3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
        );

        jPanel4.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel4.setName("jPanel4"); // NOI18N
        jPanel4.setOpaque(false);

        jsUnDiaRetraso.setModel(new
javax.swing.SpinnerNumberModel(0.0d, 0.0d, 50.0d, 0.25d));

jsUnDiaRetraso.setToolTipText(resourceMap.getString("jsUnDiaRetraso.to
olTipText")); // NOI18N
        jsUnDiaRetraso.setName("jsUnDiaRetraso"); // NOI18N

        jsDosDiasRetraso.setModel(new
javax.swing.SpinnerNumberModel(0.0d, 0.0d, 50.0d, 0.25d));

jsDosDiasRetraso.setToolTipText(resourceMap.getString("jsDosDiasRetras
```

```
o.toolTipText")); // NOI18N
        jsDosDiasRetraso.setName("jsDosDiasRetraso"); // NOI18N

        jsDiaExtraRetraso.setModel(new
javax.swing.SpinnerNumberModel(0.0d, 0.0d, 50.0d, 0.25d));

jsDiaExtraRetraso.setToolTipText(resourceMap.getString("jsDiaExtraRetr
aso.toolTipText")); // NOI18N
        jsDiaExtraRetraso.setName("jsDiaExtraRetraso"); // NOI18N

        javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4);
        jPanel4.setLayout(jPanel4Layout);
        jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel4Layout.createSequentialGroup()
                .addContainerGap(40, Short.MAX_VALUE)

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jsDiaExtraRetraso,
javax.swing.GroupLayout.PREFERRED_SIZE, 63,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jsDosDiasRetraso,
javax.swing.GroupLayout.PREFERRED_SIZE, 63,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jsUnDiaRetraso,
javax.swing.GroupLayout.PREFERRED_SIZE, 63,
javax.swing.GroupLayout.PREFERRED_SIZE)))
        );
        jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel4Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jsUnDiaRetraso,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jsDosDiasRetraso,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jsDiaExtraRetraso,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
        );

        javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(
```

```
jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jPanel4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );

        jPanel5.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel5.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel5.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel5.border.titleFont"))); // NOI18N
        jPanel5.setName("jPanel5"); // NOI18N
        jPanel5.setOpaque(false);

        jPanel6.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel6.setName("jPanel6"); // NOI18N
        jPanel6.setOpaque(false);

        jLabel4.setFont(resourceMap.getFont("jLabel4.font")); //
NOI18N
        jLabel4.setText(resourceMap.getString("jLabel4.text")); //
NOI18N
        jLabel4.setName("jLabel4"); // NOI18N

        jLabel5.setFont(resourceMap.getFont("jLabel5.font")); //
NOI18N
        jLabel5.setText(resourceMap.getString("jLabel5.text")); //
NOI18N
        jLabel5.setName("jLabel5"); // NOI18N
```

```
        jLabel6.setFont(resourceMap.getFont("jLabel6.font")); //
NOI18N
        jLabel6.setText(resourceMap.getString("jLabel6.text")); //
NOI18N
        jLabel6.setName("jLabel6"); // NOI18N

        jLabel8.setFont(resourceMap.getFont("jLabel8.font")); //
NOI18N
        jLabel8.setText(resourceMap.getString("jLabel8.text")); //
NOI18N
        jLabel8.setName("jLabel8"); // NOI18N

        javax.swing.GroupLayout jPanel6Layout = new
javax.swing.GroupLayout(jPanel6);
        jPanel6.setLayout(jPanel6Layout);
        jPanel6Layout.setHorizontalGroup(

jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel6Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jLabel4,
javax.swing.GroupLayout.DEFAULT_SIZE, 214, Short.MAX_VALUE)
                    .addComponent(jLabel5,
javax.swing.GroupLayout.DEFAULT_SIZE, 214, Short.MAX_VALUE)
                    .addComponent(jLabel6,
javax.swing.GroupLayout.DEFAULT_SIZE, 214, Short.MAX_VALUE)
                    .addComponent(jLabel8,
javax.swing.GroupLayout.DEFAULT_SIZE, 214, Short.MAX_VALUE))
                .addContainerGap())
        );
        jPanel6Layout.setVerticalGroup(

jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel6Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel4)
                .addGap(18, 18, 18)
                .addComponent(jLabel5)
                .addGap(18, 18, 18)
                .addComponent(jLabel6)
                .addGap(18, 18, 18)
                .addComponent(jLabel8)
                .addContainerGap(14, Short.MAX_VALUE))
        );

        jPanel7.setBackground(new java.awt.Color(0x00ffffff, true ));
        jPanel7.setName("jPanel7"); // NOI18N
        jPanel7.setOpaque(false);

        jsDuracionPrestamo.setModel(new
javax.swing.SpinnerNumberModel(1, 1, 50, 1));

jsDuracionPrestamo.setToolTipText(resourceMap.getString("jsDuracionPre
stamo.toolTipText")); // NOI18N
        jsDuracionPrestamo.setName("jsDuracionPrestamo"); // NOI18N
```

```
        jsEjemplaresPorPersona.setModel(new
javax.swing.SpinnerNumberModel(1, 1, 50, 1));

jsEjemplaresPorPersona.setToolTipText(resourceMap.getString("jsEjempla
resPorPersona.toolTipText")); // NOI18N
        jsEjemplaresPorPersona.setName("jsEjemplaresPorPersona"); //
NOI18N

        jcbPrestamoTesis.setModel(new
javax.swing.DefaultComboBoxModel(new String[] { "Permitido", "No
permitido" }));

jcbPrestamoTesis.setToolTipText(resourceMap.getString("jcbPrestamoTesi
s.toolTipText")); // NOI18N
        jcbPrestamoTesis.setName("jcbPrestamoTesis"); // NOI18N


jtfMoneda.setHorizontalAlignment(javax.swing.JTextField.RIGHT);
        jtfMoneda.setText(resourceMap.getString("jtfMoneda.text")); //
NOI18N

jtfMoneda.setToolTipText(resourceMap.getString("jtfMoneda.toolTipText"
)); // NOI18N
        jtfMoneda.setName("jtfMoneda"); // NOI18N

        javax.swing.GroupLayout jPanel7Layout = new
javax.swing.GroupLayout(jPanel7);
        jPanel7.setLayout(jPanel7Layout);
        jPanel7Layout.setHorizontalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addComponent(jsDuracionPrestamo,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 63,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jsEjemplaresPorPersona,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 63,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jcbPrestamoTesis, 0,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jtfMoneda,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 85, Short.MAX_VALUE))
                .addContainerGap())
        );
        jPanel7Layout.setVerticalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel7Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jsDuracionPrestamo,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jsEjemplaresPorPersona,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jcbPrestamoTesis,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jtfMoneda,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        javax.swing.GroupLayout jPanel5Layout = new
javax.swing.GroupLayout(jPanel5);
        jPanel5.setLayout(jPanel5Layout);
        jPanel5Layout.setHorizontalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel5Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel6,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        jPanel5Layout.setVerticalGroup(

jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel5Layout.createSequentialGroup()

.addGroup(jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING, false)
                    .addComponent(jPanel6,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel7,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
```

```
                .addContainerGap(12, Short.MAX_VALUE))
        );

        jPanel8.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel8.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel8.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel8.border.titleFont"))); // NOI18N
        jPanel8.setName("jPanel8"); // NOI18N
        jPanel8.setOpaque(false);

        jLabel7.setFont(resourceMap.getFont("jLabel7.font")); //
NOI18N
        jLabel7.setText(resourceMap.getString("jLabel7.text")); //
NOI18N
        jLabel7.setName("jLabel7"); // NOI18N

        jtfDirHTML.setEditable(false);
        jtfDirHTML.setText(resourceMap.getString("jtfDirHTML.text"));
// NOI18N

jtfDirHTML.setToolTipText(resourceMap.getString("jtfDirHTML.toolTipTex
t")); // NOI18N
        jtfDirHTML.setName("jtfDirHTML"); // NOI18N

        jButton1.setIcon(resourceMap.getIcon("jButton1.icon")); //
NOI18N
        jButton1.setText(resourceMap.getString("jButton1.text")); //
NOI18N
        jButton1.setName("jButton1"); // NOI18N
        jButton1.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton1ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel8Layout = new
javax.swing.GroupLayout(jPanel8);
        jPanel8.setLayout(jPanel8Layout);
        jPanel8Layout.setHorizontalGroup(

jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel8Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                    .addGroup(jPanel8Layout.createSequentialGroup()
                        .addComponent(jLabel7,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addGap(129, 129, 129))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel8Layout.createSequentialGroup()
```

```
                        .addComponent(jButton1)
                        .addContainerGap())
                    .addGroup(jPanel8Layout.createSequentialGroup()
                        .addComponent(jtfDirHTML,
javax.swing.GroupLayout.DEFAULT_SIZE, 318, Short.MAX_VALUE)
                        .addContainerGap())))
        );
        jPanel8Layout.setVerticalGroup(

jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel8Layout.createSequentialGroup()
                .addComponent(jLabel7)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
18, Short.MAX_VALUE)
                .addComponent(jtfDirHTML,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jButton1)
                .addContainerGap())
        );

        jButton3.setIcon(resourceMap.getIcon("jButton3.icon")); //
NOI18N
        jButton3.setText(resourceMap.getString("jButton3.text")); //
NOI18N
        jButton3.setName("jButton3"); // NOI18N
        jButton3.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton3ActionPerformed(evt);
            }
        });

        jButton4.setIcon(resourceMap.getIcon("jButton4.icon")); //
NOI18N
        jButton4.setText(resourceMap.getString("jButton4.text")); //
NOI18N
        jButton4.setName("jButton4"); // NOI18N
        jButton4.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton4ActionPerformed(evt);
            }
        });

        jPanel9.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel9.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel9.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel9.border.titleFont"))); // NOI18N
        jPanel9.setName("jPanel9"); // NOI18N
        jPanel9.setOpaque(false);
```

```
        jButton5.setIcon(resourceMap.getIcon("jButton5.icon")); //
NOI18N
        jButton5.setText(resourceMap.getString("jButton5.text")); //
NOI18N
        jButton5.setName("jButton5"); // NOI18N
        jButton5.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton5ActionPerformed(evt);
            }
        });

        jButton6.setIcon(resourceMap.getIcon("jButton6.icon")); //
NOI18N
        jButton6.setText(resourceMap.getString("jButton6.text")); //
NOI18N
        jButton6.setName("jButton6"); // NOI18N

        jtfDirHTML2.setEditable(false);

jtfDirHTML2.setText(resourceMap.getString("jtfDirHTML2.text")); //
NOI18N

jtfDirHTML2.setToolTipText(resourceMap.getString("jtfDirHTML2.toolTipT
ext")); // NOI18N
        jtfDirHTML2.setName("jtfDirHTML2"); // NOI18N

        jButton7.setText(resourceMap.getString("jButton7.text")); //
NOI18N
        jButton7.setName("jButton7"); // NOI18N

        javax.swing.GroupLayout jPanel9Layout = new
javax.swing.GroupLayout(jPanel9);
        jPanel9.setLayout(jPanel9Layout);
        jPanel9Layout.setHorizontalGroup(

jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel9Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel9Layout.createSequentialGroup()
                        .addComponent(jtfDirHTML2,
javax.swing.GroupLayout.DEFAULT_SIZE, 288, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jButton7,
javax.swing.GroupLayout.PREFERRED_SIZE, 24,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jButton6,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 112,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton5,
javax.swing.GroupLayout.DEFAULT_SIZE, 318, Short.MAX_VALUE))
```

```
                .addContainerGap())
        );
        jPanel9Layout.setVerticalGroup(

jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel9Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jButton5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
45, Short.MAX_VALUE)

.addGroup(jPanel9Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(jtfDirHTML2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton7))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jButton6)
                .addContainerGap())
        );

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                        .addComponent(jButton4,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(jButton3,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                            .addComponent(jPanel5,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
                            .addComponent(jPanel9,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jPanel8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING, false)
                    .addComponent(jPanel8,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jPanel2,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING, false)
                    .addComponent(jPanel9,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel5,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
44, Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.BASELINE)
                    .addComponent(jButton3)
                    .addComponent(jButton4)))
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
```

```
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE, 410, Short.MAX_VALUE)
        );
    }// </editor-fold>

    private void jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {
        cambiarDireccionHTML();
    }

    private void
jFileChooserActionPerformed(java.awt.event.ActionEvent evt) {
        cargarCadena();

    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent
evt) {
        pantallaPrincipal.home();
    }

    private void jButton4ActionPerformed(java.awt.event.ActionEvent
evt) {
        guardarTodo ();
        pantallaPrincipal.home();
    }

    private void jButton5ActionPerformed(java.awt.event.ActionEvent
evt) {
        respaldarBD();
    }


    // Variables declaration - do not modify
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JButton jButton5;
    private javax.swing.JButton jButton6;
    private javax.swing.JButton jButton7;
    private javax.swing.JFileChooser jFileChooser;
    private javax.swing.JFileChooser jFileChooser1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
```

```
    private javax.swing.JPanel jPanel4;
    private javax.swing.JPanel jPanel5;
    private javax.swing.JPanel jPanel6;
    private javax.swing.JPanel jPanel7;
    private javax.swing.JPanel jPanel8;
    private javax.swing.JPanel jPanel9;
    private javax.swing.JComboBox jcbPrestamoTesis;
    private javax.swing.JSpinner jsDiaExtraRetraso;
    private javax.swing.JSpinner jsDosDiasRetraso;
    private javax.swing.JSpinner jsDuracionPrestamo;
    private javax.swing.JSpinner jsEjemplaresPorPersona;
    private javax.swing.JSpinner jsUnDiaRetraso;
    private javax.swing.JTextField jtfDirHTML;
    private javax.swing.JTextField jtfDirHTML2;
    private javax.swing.JTextField jtfMoneda;
    // End of variables declaration


     private void respaldarBD() {
         String directorio =
gestorVariablesSistema.getDirectorioHTML();
         directorio += "/BACKUPS/";
         new File(directorio).mkdirs();
         gestorVariablesSistema.respaldarBD(directorio);

     }

    private void cargarCadena(){
        String cadena;
        if (jFileChooser.getSelectedFile().getPath().toString()==null)
            return;
        cadena = jFileChooser.getSelectedFile().getPath().toString();
        cadena = Matcher.quoteReplacement(cadena);
        cadena = cadena.replaceAll("\\\\", "/");
        cadena = cadena.replaceAll("//", "/");

        jtfDirHTML.setText(cadena);

    }

    // Carga los valores de las variables del sistema en el esquema
    // de la interfaz grafica
    private void cargarValores() {

jsUnDiaRetraso.setValue(gestorVariablesSistema.getCargoPorUnDia());

jsDosDiasRetraso.setValue(gestorVariablesSistema.getCargoPorDosDias())
;

jsDiaExtraRetraso.setValue(gestorVariablesSistema.getCargoPorDiaExtra(
));

jsDuracionPrestamo.setValue(gestorVariablesSistema.getDuracionDeUnPres
tamo());

jsEjemplaresPorPersona.setValue(gestorVariablesSistema.getMaxEjemplare
sPorPersona());
        jtfMoneda.setText(gestorVariablesSistema.getMoneda());
        if (gestorVariablesSistema.getTesisPrestamoExterno())
            jcbPrestamoTesis.setSelectedIndex(0);
        else
```

```
            jcbPrestamoTesis.setSelectedIndex(1);

jtfDirHTML.setText(gestorVariablesSistema.getDirectorioHTML());
    }

    // Abre el file chooser para q la persona pueda cambiar
    // el directorio del exportar a html
    private void cambiarDireccionHTML() {
        jFileChooser.setEnabled(true);
        jFileChooser.setVisible(true);
        jFileChooser.showSaveDialog(this);
    }

    private void guardarTodo() {

gestorVariablesSistema.setCargoPorUnDia(Double.valueOf(jsUnDiaRetraso.
getValue().toString()));

gestorVariablesSistema.setCargoPorDosDias(Double.valueOf(jsDosDiasRetr
aso.getValue().toString()));

gestorVariablesSistema.setCargoPorDiaExtra(Double.valueOf(jsDiaExtraRe
traso.getValue().toString()));

gestorVariablesSistema.setDuracionDeUnPrestamo(Integer.valueOf(jsDurac
ionPrestamo.getValue().toString()));

gestorVariablesSistema.setMaxEjemplaresPorPersona(Integer.valueOf(jsEj
emplaresPorPersona.getValue().toString()));
        gestorVariablesSistema.setMoneda(jtfMoneda.getText().trim());

gestorVariablesSistema.setDirectorioHTML(jtfDirHTML.getText().trim());
        if (jcbPrestamoTesis.getSelectedIndex()==0){
            System.out.println("PRESTAMO = TRUE ");
            gestorVariablesSistema.setTesisPrestamoExterno(true);
        }
        else {
            gestorVariablesSistema.setTesisPrestamoExterno(false);
            System.out.println("PRESTAMO = FALSE ");
        }

        gestorVariablesSistema.guardarValoresBD();
    }

}
```

5.3.2.2.8 GestorVariablesSistema

```
package GestionPrestamos;

import BaseDeDatos.ConexionBD;
import Principal.GUIView;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
```

```
import java.util.regex.Pattern;


public class GestorVariablesSistema {

private GUIView pantallaPrincipal;
private ConexionBD conexionBD;
private ResultSet rs;

// Variables de pago por retraso de prestamo
private double cargoUnDia;              // Deuda por un dia de atraso
private double cargoDosDias;            // Deuda por dos dias de atraso
private double cargoDiaExtra;           // Deuda por dia extra a partir
de dos dias
                                        // de atraso
private int maxDuracionDePrestamo;  // Cantidad de dias que se puede
prestar
                                        // un ejemplar
private int maxEjemplaresPorPersona;// Cantidad maxima de ejemplares
que se
                                        // pueden prestar a una persona
private String directoriosreporteshtml; //directorio en el cual se
almacenaran
                                         // los archivos de reporte
html
private boolean prestarTesisExterno; // Determina si es posible
                                     // procesar un prestamo externo
                                     // de una tesis
private String moneda; // Almacena la moneda actual
// Contructor
public GestorVariablesSistema (GUIView pantallaPrincipal){
    this.pantallaPrincipal=pantallaPrincipal;
    cargarVariablesSistema ();
}

// Metodo que accede a la base de datos buscando en la tabla
// opcionessitema las variables del sistema y las carga
public boolean cargarVariablesSistema (){
        // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();

        conexionBD.Consulta("SELECT * FROM opcionessistema");
        rs = conexionBD.MostrarDatosConsulta();
        try {
            if (rs.next()!=false){
                cargoUnDia= rs.getDouble("cargo_por_un_dia");
                cargoDosDias= rs.getDouble("cargo_por_dos_dias");
                cargoDiaExtra =
rs.getDouble("cargo_por_dia_adicional");
                maxDuracionDePrestamo = rs.getInt("duracionPrestamo");
                maxEjemplaresPorPersona =
rs.getInt("nroejemplaresporpersona");
                directoriosreporteshtml =
rs.getString("directorioreportehtml");
                prestarTesisExterno =
rs.getBoolean("prestamo_tesis_externo");
                moneda = rs.getString("moneda");
```

```
                conexionBD.CerrarConexion();
                return true;
            }
            else{
                conexionBD.CerrarConexion();
                pantallaPrincipal.errorBarraEstado("Error al cargar
las variables del SIGESLEC.");
                return false;
            }
        } catch ( SQLException ex) {
            conexionBD.CerrarConexion();
            pantallaPrincipal.errorBarraEstado("Error al acceder a la
base de datos.");

Logger.getLogger(GestorVariablesSistema.class.getName()).log(Level.SEV
ERE, null, ex);
            return false;
        }
}

void guardarValoresBD() {
     // limpia los mensajes de la barra de estado de la pantalla
principal
        pantallaPrincipal.borrarBarraEstado();
        // Inicia y abre la conexion con la base de datos
        conexionBD = new ConexionBD(pantallaPrincipal);
        conexionBD.AbrirConexion();

        if (conexionBD.Actualizar("UPDATE opcionessistema SET
cargo_por_un_dia = '"+ this.getCargoPorUnDia()+

"',cargo_por_dos_dias = '"+ this.getCargoPorDosDias()+

"',cargo_por_dia_adicional ='"+ this.getCargoPorDiaExtra()+

"',duracionprestamo = '"+this.getDuracionDeUnPrestamo()+

"',nroejemplaresporpersona ='"+this.getMaxEjemplaresPorPersona()+

"',directorioreportehtml ='"+this.getDirectorioHTML()+

"',prestamo_tesis_externo ='"+this.getTesisPrestamoExterno()+
                                                    "',moneda
='"+this.getMoneda()+"'"))
                                                    //"' WHERE
numerocedula = '"+gestorEstudiante.getCedula()+"'" );
            pantallaPrincipal.mensajeBarraEstado("Datos actualizados
correctamente.");
        else
            pantallaPrincipal.errorBarraEstado("Error al acceder a la
base de datos.");
        conexionBD.CerrarConexion();

}

public void respaldarBD (String directorio){
  //  conexionBD.AbrirConexion();

    if (conexionBD.backupPGSQL(directorio))
        pantallaPrincipal.mensajeBarraEstado("Base de datos respaldada
en "+directorio);
```

```
    else
        pantallaPrincipal.errorBarraEstado("Error al respaldar la base
de datos.");

 //  conexionBD.CerrarConexion();
}

public String getMoneda(){
    return moneda;
}
public boolean getTesisPrestamoExterno (){
    return prestarTesisExterno;
}
public String getDirectorioHTML (){
    return directoriosreporteshtml;
}
public double getCargoPorUnDia (){
    return cargoUnDia;
}
public double getCargoPorDosDias (){
    return cargoDosDias;
}
public double getCargoPorDiaExtra (){
    return cargoDiaExtra;
}
public int getDuracionDeUnPrestamo(){
    return maxDuracionDePrestamo;
}
public int getMaxEjemplaresPorPersona(){
    return maxEjemplaresPorPersona;
}


public void setMoneda (String moneda){
    this.moneda=moneda;
}
public void setTesisPrestamoExterno (boolean prestarTesisExterno){
    this.prestarTesisExterno=prestarTesisExterno;
}
public void setDirectorioHTML (String directoriosreporteshtml){
    this.directoriosreporteshtml=directoriosreporteshtml;
}
public void setCargoPorUnDia (double cargoUnDia){
    this.cargoUnDia=cargoUnDia;
}
public void setCargoPorDosDias (double cargoDosDias){
    this.cargoDosDias=cargoDosDias;
}
public void setCargoPorDiaExtra (double cargoDiaExtra){
    this.cargoDiaExtra=cargoDiaExtra;
}
public void setDuracionDeUnPrestamo(int maxDuracionDePrestamo){
    this.maxDuracionDePrestamo=maxDuracionDePrestamo;
}
public void setMaxEjemplaresPorPersona(int maxEjemplaresPorPersona){
    this.maxEjemplaresPorPersona=maxEjemplaresPorPersona;
}
```

```
}
```

5.3.2.2.9 GestorInfoPrestamo

```
package GestionPrestamos;


// Autores: Oscar Saavedra y Sair Amer
//      Universidad de Oriente
//    Barcelona - Venezuela - 2009


// Clase para gestionar temporalmente la informacion
// de un prestamo
public class GestorInfoPrestamo {

    private String idPrestamo;
    private String cedulaEstudiante;
    private String fecha;
    private String fechaEntrega;
    private String idEjemplar;
    private String idAyudante;
    private boolean prestamoInterno;
    private int diasAtraso;
    private double deuda;

    public GestorInfoPrestamo(){
        nuevoPrestamo();
    }

    // Limpia todos los valores del prestamo
    public void nuevoPrestamo(){
       idPrestamo="";
       cedulaEstudiante="";
       fecha="";
       idEjemplar="";
       idAyudante="";
       fechaEntrega="";
    }

    // Metodos para estables los valores del prestamo

    public double getDeuda (){
        return deuda;
    }

    public int getDiaAtraso (){
        return diasAtraso;
    }



    public boolean getPrestamoINTERNO (){
        return prestamoInterno;
    }
```

```
    public String getIdPrestamo(){
        return idPrestamo;
    }
    public String getCedulaEstudiante () {
        return cedulaEstudiante;
    }
    public String getFecha (){
        return fecha;
    }
    public String getFechaEntrega (){
        return fechaEntrega;
    }
    public String getIdEjemplar (){
        return idEjemplar;
    }
    public String getIdAyudante (){
        return idAyudante;
    }

    // Metodo para retornar los valores del prestamo
    public void setDeuda (double deuda){
        this.deuda=deuda;
    }

    public void setDiasAtraso (int diasAtraso){
        this.diasAtraso=diasAtraso;
    }
    public void setPrestamoINTERNO (boolean prestamoInterno){
        this.prestamoInterno=prestamoInterno;
    }

    public void setIdPrestamo(String idPrestamo) {
        this.idPrestamo=idPrestamo;
    }

    public void setCeduladEstudiante (String cedulaEstudiante){
        this.cedulaEstudiante=cedulaEstudiante;
    }
    public void setFechaEntrega (String fechaEntrega){
        this.fechaEntrega=fechaEntrega;
    }
    public void setFecha (String fecha){
        this.fecha=fecha;
    }
    public void setIdEjemplar (String idEjemplar){
        this.idEjemplar=idEjemplar;
    }
    public void setIdAyudante (String idAyudante){
        this.idAyudante=idAyudante;
    }


}
```

5.3.2.2.10 GUICobroPago

```
package GestionPrestamos;

import java.awt.Graphics;
import java.awt.Graphics2D;
import javax.swing.ImageIcon;


public class GUICobroPago extends javax.swing.JFrame {
    private String path;

    public GUICobroPago() {
        initComponents();

this.setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_C
LOSE);
        path = determinarPath();
    }

    private String determinarPath() {
        String pathLocal;
        pathLocal = this.getClass().getResource("").getPath();
        pathLocal = pathLocal.substring(1,
pathLocal.lastIndexOf("GestionPrestamos"));
        pathLocal+="Principal/resources/";
        pathLocal = pathLocal.replaceAll("%20"," ");
        return pathLocal;
    }

  //  @Override
    public void paintComponent(Graphics g)
     {
        Graphics2D g2 = (Graphics2D)g;
        g2.drawImage(new
ImageIcon(path+"fondo.png").getImage(),0,0,this.getWidth(),this.getHei
ght(),this);

    }


    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jTextField2 = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        jTextField1 = new javax.swing.JTextField();
        jSeparator1 = new javax.swing.JSeparator();
        jPanel3 = new javax.swing.JPanel();
        jLabel3 = new javax.swing.JLabel();
        jTextField3 = new javax.swing.JTextField();
        jButton1 = new javax.swing.JButton();


setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE)
```

```
;
        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(Principal.GUIApp.clas
s).getContext().getResourceMap(GUICobroPago.class);
        setTitle(resourceMap.getString("Form.title")); // NOI18N
        setAlwaysOnTop(true);
        setName("Form"); // NOI18N

        jPanel1.setBackground(new java.awt.Color(0x00ffffff, true ));

jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAIS
ED), resourceMap.getString("jPanel1.border.title"),
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION,
resourceMap.getFont("jPanel1.border.titleFont"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        jPanel2.setName("jPanel2"); // NOI18N
        jPanel2.setLayout(new java.awt.GridLayout(2, 2, 0, 5));

        jLabel1.setFont(resourceMap.getFont("jLabel1.font")); //
NOI18N
        jLabel1.setText(resourceMap.getString("jLabel1.text")); //
NOI18N
        jLabel1.setName("jLabel1"); // NOI18N
        jPanel2.add(jLabel1);

        jTextField2.setEditable(false);
        jTextField2.setFont(resourceMap.getFont("jTextField2.font"));
// NOI18N
        jTextField2.setName("jTextField2"); // NOI18N
        jPanel2.add(jTextField2);

        jLabel2.setFont(resourceMap.getFont("jLabel2.font")); //
NOI18N
        jLabel2.setText(resourceMap.getString("jLabel2.text")); //
NOI18N
        jLabel2.setName("jLabel2"); // NOI18N
        jPanel2.add(jLabel2);

        jTextField1.setFont(resourceMap.getFont("jTextField1.font"));
// NOI18N

jTextField1.setText(resourceMap.getString("jTextField1.text")); //
NOI18N
        jTextField1.setName("jTextField1"); // NOI18N
        jPanel2.add(jTextField1);

        jSeparator1.setName("jSeparator1"); // NOI18N

        jPanel3.setName("jPanel3"); // NOI18N
        jPanel3.setLayout(new java.awt.GridLayout(1, 2, 5, 0));

        jLabel3.setFont(resourceMap.getFont("jLabel3.font")); //
NOI18N
        jLabel3.setText(resourceMap.getString("jLabel3.text")); //
NOI18N
        jLabel3.setName("jLabel3"); // NOI18N
        jPanel3.add(jLabel3);
```

```
        jTextField3.setFont(resourceMap.getFont("jTextField3.font"));
// NOI18N

jTextField3.setText(resourceMap.getString("jTextField3.text")); //
NOI18N
        jTextField3.setName("jTextField3"); // NOI18N
        jPanel3.add(jTextField3);

        jButton1.setFont(resourceMap.getFont("jButton1.font")); //
NOI18N
        jButton1.setIcon(resourceMap.getIcon("jButton1.icon")); //
NOI18N
        jButton1.setText(resourceMap.getString("jButton1.text")); //
NOI18N
        jButton1.setName("jButton1"); // NOI18N

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.TRAILING)
                    .addComponent(jButton1,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 322, Short.MAX_VALUE)
                    .addComponent(jPanel3,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 322, Short.MAX_VALUE)
                    .addComponent(jSeparator1,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 322, Short.MAX_VALUE)
                    .addComponent(jPanel2,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 322, Short.MAX_VALUE))
                .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jSeparator1,
javax.swing.GroupLayout.PREFERRED_SIZE, 10,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel3,
```

```
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
75, Short.MAX_VALUE)
                .addComponent(jButton1,
javax.swing.GroupLayout.PREFERRED_SIZE, 34,
javax.swing.GroupLayout.PREFERRED_SIZE))
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );

        pack();
    }// </editor-fold>

    /**
    * @param args the command line arguments
    */


    // Variables declaration - do not modify
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
    // End of variables declaration

}
```

5.3.2.2.11 PrestamosExportarPDF

```
package GestionPrestamos;


import Principal.GUIView;
import com.lowagie.text.Image;
import com.lowagie.text.pdf.PdfPCell;
import com.lowagie.text.pdf.PdfPTable;
import java.io.FileOutputStream;
import java.io.IOException;
import com.lowagie.text.PageSize;
import com.lowagie.text.Document;
import com.lowagie.text.Element;
import com.lowagie.text.DocumentException;
import com.lowagie.text.Font;
import com.lowagie.text.FontFactory;
import com.lowagie.text.Paragraph;
import com.lowagie.text.pdf.PdfWriter;
import java.awt.Color;
import java.io.File;
import java.util.Calendar;
import javax.swing.JTable;

// Autores: Oscar Saavedra y Sair Amer
//      Universidad de Oriente
//    Barcelona - Venezuela - 2009

public class PrestamosExportarPDF {
    private GUIView pantallaPrincipal;
    private GestorVariablesSistema gestorVariables;
    private String Fecha;
    private String Hora;
    private String nombreArchivo;
    private Color Azul;
    private Color Verde;
    private Color Rojo;

    PrestamosExportarPDF(GUIView pantallaPrincipal) {
        this.pantallaPrincipal = pantallaPrincipal;
        gestorVariables = new
GestorVariablesSistema(pantallaPrincipal);
        Azul = new Color(51,94,168);
        Verde = new Color (51,153,0);
        Rojo = new Color (255, 51, 51);
    }

    public boolean exportarReportesPorFecha (JTable lista, String
titulo){
        calcularFecha ();
        nombreArchivo = gestorVariables.getDirectorioHTML();


        try {
                // Creamos el documento con sus margenes
                Document document = new Document(PageSize.A4, 30, 30,
20, 50);
                document.addTitle("Lista de prestamos");
```

```
                document.addAuthor("SIGESLEC");
                document.addSubject("Lista de prestamos.");
                document.addKeywords("Lista,Prestamos,Sala,Lectura");
                document.addCreator("SIGESLEC usando el paquete
iText");
                // Se crea el writer
                nombreArchivo +=
Fecha+"_"+Hora+"_"+"LISTA_PRESTAMOS.pdf" ;

                PdfWriter.getInstance(document, new
FileOutputStream(nombreArchivo));

                // Se abre el documento a editar
                document.open();

                //Se carga la imagen del encabezado
                Image encabezado =
Image.getInstance(GUIView.class.getResource

("resources/bannerPDF.png").getPath());

                encabezado.setAlignment(Image.ALIGN_CENTER);
                encabezado.scaleAbsolute(495, 45);

                document.add(encabezado);

                Font letraTemp =
FontFactory.getFont(FontFactory.COURIER, 12, Font.BOLD,
                                    Color.BLACK);
                Paragraph p = new Paragraph(titulo);
                p.setAlignment(Element.ALIGN_CENTER);
                document.add(p);
                p = new Paragraph(Fecha.replaceAll("_",
"/"),letraTemp);
                p.setSpacingAfter(20);
                p.setAlignment(Element.ALIGN_CENTER);
                document.add(p);

                letraTemp = FontFactory.getFont(FontFactory.COURIER,
10, Font.BOLD,
                                    Color.BLACK);

                PdfPTable table = new PdfPTable(6);
                table.setTotalWidth(535f);
                table.setLockedWidth(true);
                float[] anchoColumnas = { 1f, 0.8f, 1f,1f,0.8f,0.9f };
                table.setWidths(anchoColumnas);
                PdfPCell cell = new PdfPCell(new
Paragraph("Cédula",letraTemp));
                cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
                table.addCell(cell);
                cell = new PdfPCell(new Paragraph("Cota",letraTemp));
                cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
                table.addCell (cell);
                cell = new PdfPCell(new Paragraph("Fecha
Prestamo",letraTemp));
                cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
                table.addCell (cell);
                cell = new PdfPCell(new Paragraph("Fecha
Recepción",letraTemp));
                cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
```

```
                table.addCell (cell);
                cell = new PdfPCell(new
Paragraph("Estado",letraTemp));
                cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
                table.addCell (cell);
                cell = new PdfPCell(new Paragraph("Tipo",letraTemp));
                cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
                table.addCell (cell);
                Font Cedula =   FontFactory.getFont(FontFactory.TIMES,
10, Font.BOLD,Color.BLACK);
                Font Normal =   FontFactory.getFont(FontFactory.TIMES,
10, Font.NORMAL,Color.BLACK);
                Font solvente = FontFactory.getFont(FontFactory.TIMES,
10, Font.BOLD,Verde);
                Font no_solvente =
FontFactory.getFont(FontFactory.TIMES, 10, Font.BOLD,Rojo);
                Font inscrito = FontFactory.getFont(FontFactory.TIMES,
10, Font.BOLD,
                                        Azul);
                Font no_inscrito =
FontFactory.getFont(FontFactory.TIMES, 10, Font.BOLD,
Color.DARK_GRAY);

for (int i=0;i<lista.getRowCount();i++){
                    cell = new PdfPCell(new
Paragraph(lista.getValueAt(i,0).toString(),Cedula));
                    cell.setHorizontalAlignment(PdfPCell.ALIGN_RIGHT);
                    table.addCell (cell);
                    cell = new PdfPCell(new
Paragraph(lista.getValueAt(i,1).toString(),Cedula));
                    cell.setHorizontalAlignment(PdfPCell.ALIGN_RIGHT);
                    table.addCell (cell);
                    cell = new PdfPCell(new
Paragraph(lista.getValueAt(i,2).toString(),Normal));

cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
                    table.addCell (cell);
                    cell = new PdfPCell(new
Paragraph(lista.getValueAt(i,3).toString(),Normal));

cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
                    table.addCell (cell);
                    if
(lista.getValueAt(i,4).toString().compareTo("RECIBIDO")==0)
                        cell = new PdfPCell(new
Paragraph(lista.getValueAt(i,4).toString(),solvente));
                   else
cell = new PdfPCell(new
Paragraph(lista.getValueAt(i,4).toString(),no_solvente));

cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
                    table.addCell (cell);
                    if
(lista.getValueAt(i,5).toString().compareTo("INTERNO")==0)
                        cell = new PdfPCell(new
Paragraph(lista.getValueAt(i,5).toString(),inscrito));
                    else
                        cell = new PdfPCell(new
Paragraph(lista.getValueAt(i,5).toString(),no_inscrito));

cell.setHorizontalAlignment(PdfPCell.ALIGN_CENTER);
```

```
                    table.addCell (cell);
                }
              document.add(table);
              document.close();
              return true;
            } catch (DocumentException de) {
                pantallaPrincipal.errorBarraEstado("Error al generar
el archivos PDF.");
                System.err.println(de.getMessage());
                return false;
        } catch (IOException ioe) {
                pantallaPrincipal.errorBarraEstado("Error al exportar
el archivos PDF.");
                System.err.println(ioe.getMessage());
                return false;
         }
    }

    private void calcularFecha (){
        Calendar cal = Calendar.getInstance();
        Fecha = "";
        Hora  = "";
        if (cal.get(Calendar.DAY_OF_MONTH)<10)
            Fecha += "0"+cal.get(Calendar.DAY_OF_MONTH);
        else
            Fecha += cal.get(Calendar.DAY_OF_MONTH);

        Fecha += "_";
        if ((cal.get(Calendar.MONTH)+1)<10)
            Fecha += "0"+(cal.get(Calendar.MONTH)+1);
        else
            Fecha += (cal.get(Calendar.MONTH)+1);
        Fecha += "_";
        Fecha += cal.get(Calendar.YEAR);

        if (cal.get(Calendar.HOUR)<10)
            Hora+= "0"+cal.get(Calendar.HOUR);
        else
            Hora+= cal.get(Calendar.HOUR);
        Hora += "_";
        if (cal.get(Calendar.MINUTE)<10)
            Hora+= "0"+cal.get(Calendar.MINUTE);
        else
            Hora+= cal.get(Calendar.MINUTE);
    }}
```

5.4 Conclusión de la Fase de Implementación

Durante la fase de implementación se completó el diseño de la arquitectura base de SIGESLEC y se codificaron e integraron todos los componentes del sistema.

Las pruebas de unidad se aplicaron mediante el método de caja negra, adicionalmente se aplicaron pruebas de integración lo cual permitió detectar y corregir errores en el funcionamiento de los componentes y en la forma de ajustarse unos a otros.

Durante la construcción, se amortizaron los riesgos correspondientes a esta fase, logrando el propósito de tener lista la primera versión funcional del sistema.

Capítulo VI: Fase de Transición

6.1 Introducción

La fase de transición tiene como objetivo colocar el producto en manos de los usuarios finales, durante la cual se pueden desarrollar nuevas versiones mejoradas del producto, completar la documentación mediante la elaboración del manual de usuario, adiestrar al usuario en el manejo del producto, y en general se realizan tareas relacionadas con el ajuste, configuración, instalación y facilidad de uso del producto.

6.2 Mejoras Realizadas al Sistema

Durante esta fase se realizaron mejoras en módulos específicos del sistema, los cuales se mencionan a continuación:

- Se realizó la validación de la longitud máxima del campo de nombre del archivo de respaldo de la base de datos, para evitar errores con el sistema operativo al realizar la creación y almacenamiento del archivo.

- Se creo una rutina que evita que se sobreescriban archivos del sistema operativo al momento de guardar los archivos de reportes PDF, HTML y Backup de base de datos.

6.3 Manual de Usuario

Como último producto se desarrolló el documento que servirá como base principal para los usuarios del sistema, para su correcto uso, este es el Manual de usuario. El manual de usuario se muestra en la sección de anexos, marcado como Anexo A.

El manual de usuario está dirigido a los usuarios finales del sistema, que serán los ayudantes y colaboradores de la sala de lectura del departamento de Computación y Sistemas de la Universidad de Oriente, Núcleo de Anzoátegui.

El manual fue elaborado la siguiente manera: Se comenzó describiendo los diversos componentes que conforman la interfaz gráfica de inicio del sistema: Pantalla de inicio, Pestaña de Estudiantes, Pestaña de Ejemplares, las distintas barras (de accesos, de estado, de menúes), de manera tal que los usuarios se familiaricen con los distintos componentes que la conforman. Seguidamente se procedió a describir detalladamente cada uno de los componentes de la interfaz que permiten la realización de todas las actividades requeridas por el sistema. Finalmente se incluyen los requisitos especiales de instalación del sistema, que son necesarios para el correcto funcionamiento del software y que, de igual manera, garantizen el correcto enlace con la base de datos de SIGESLEC.

Conclusiones

Con este trabajo de investigación se ha creado una herramienta que facilitara la gestión de estudiantes y ejemplares adscritos a la sala de lectura del departamento de computación y sistemas de la Universidad de Oriente, núcleo Anzoátegui. Dicho sistema lleva un control detallado de préstamos, devoluciones, multas en el manejo de ejemplares y de inscripciones en el manejo de estudiantes, así como también un módulo de reportes de préstamos y eventos; para garantizar el crecimiento y orden de la sala de lectura de nuestro departamento.

Este proyecto fue realizado con el objetivo fundamental de optimizar el funcionamiento de la sala de lectura, para facilitar así su uso por parte de los estudiantes y ayudantes que hacen vida en el departamento.

El RUP, (Rational Unified Process), como metodología elegida, aunado al uso de UML como instrumento principal para la documentación y guía, permitieron la organización y perfeccionamiento del sistema, efectuando todas las etapas de este proceso y ayudó a la prevención de fallas que se pudieron presentar, evitando así un replanteamiento del proyecto, canalizando los diferentes flujos de trabajo necesarios por cada una de las fases de desarrollo.

Durante la fase de inicio se determinaron los requisitos específicos necesarios para la construcción del sistema y posteriormente fueron actualizados en la fase de elaboración, derivando esto en el punto de partida para la definición de una completa funcionalidad.

La implementación de la base de datos, diseñada utilizando el modelo relacional, permite el almacenamiento y recuperación de los datos de manera sencilla y eficiente.

El uso de Java y PostgreSQL, como lenguaje de programación y Sistema Manejador de Base de Datos respectivamente, permitieron el desarrollo del sistema de manera clara y efectiva, debido a que estas tecnologías son fácilmente utilizables en cualquier entorno computacional, puesto a que sus requerimientos para el funcionamiento son mínimos y al ser tecnologías de Software Libre se pueden obtener en la Web.

Se realizó de manera exitosa la integración, pruebas y documentación del funcionamiento del sistema, se evaluó y depuró el sistema en varias iteraciones, obteniendo un software altamente funcional, al que puede realizarse mantenimiento y actualizaciones.

Recomendaciones

- Integrar al sistema un módulo de control de asistencia de ayudantes que proporcione características de registro de sesiones y privilegios de usuarios.

- Realizar mantenimiento a la base de datos periódicamente para garantizar su óptimo funcionamiento y la integridad de los datos almacenados.

- Mantener el sistema actualizado con el stock de ejemplares de la sala, para garantizar así la máxima productividad del sistema.

- Apegarse a las rutinas de mantenimiento establecidas en los manuales de usuario del lector de código de barras y de la cámara Web, para garantizar así el correcto funcionamiento de los diferentes módulos que conforman el software.

- Agregar un menú de mantenimiento para la eliminación de reportes de vieja data.

- Implementar una opción para la realización de inventario físico.

- Modificar la búsqueda de ejmplares por resumen para permitir la utilización de descriptores (Tags), para agilizar el procedimiento.

- Agregar el motivo al momento de realizar la eliminación de un ejemplar.

- Apegar la codificación de las cotas de ejemplares a los estándares ISBN.

- Apegarse a la utilización del sistema como se muestra en el manual de usuario proporcionado.

Bibliografía

[3] **Arcila, A., Zacarias, E. (2006).** "Desarrollo de un software como soporte para la automatización de las actividades administrativas que se llevan acabo en una institución educativa".
Trabajo de grado, Universidad de Oriente, Núcleo de Anzoátegui.

[2] **Caldarello, D. (1999).** "Desarrollo de un sistema homologado de control de estudio de la universidad de oriente (sishoc)".
Trabajo de grado, Universidad de Oriente, Núcleo de Sucre.

[11] Fermín, J., Ovando, N., Trabajo presentado como proyecto de fin de curso de asignatura desarrollo de software avanzado en el semestre I-2009, en el departamento de Computación y Sistemas, escuela de Ingeniería y Ciencias Aplicadas de la Universidad de Oriente, núcleo de Anzoátegui.

[8] **Grupo Soluciones Innova.** (2007). http://www.rational.com.ar
Disponible en: http://www.rational.com.ar/herramientas/rup.html

[1] **Guevara E., J. (1998).** "Desarrollo e implementación de los servicios académicos del departamento de computación y sistemas usando tecnología www".
Trabajo de grado, Universidad de Oriente, Núcleo de Anzoátegui.

[6] JOYANES, Luis, FERNÁNDEZ, Matilde, "**Java 2 Manual de Programación**", 1ra Edición, Editorial Mc Graw Hill, (2001).

[4] **Millan, L., Garelli, L. (2007).** "Desarrollo de un software que permita la automatización de las actividades asociadas al departamento de admisión y control de estudios de la extensión centro/sur del núcleo de Anzoátegui de la universidad de oriente".
Trabajo de grado, Universidad de Oriente, Núcleo de Anzoátegui.

[9] **Sun Microsystems.** (2009). http://java.com/

Disponible en:
http://java.sun.com/javase/technologies/desktop/media/jmf/2.1.1/guide/index.html

[5] TEMPLEMAN, J., OLSEN, A. "**Microsoft Visual C++. Net Aprenda Ya**". Primera Edición. Editorial McGraw – Hill, (2002).

[10] **TOMASI, W.,** "Sistemas de comunicación electrónicas".
Editorial Prentice Hall. (2003).

[7] ULLMAN, J., WIDOM, P., "**Introducción a los sistemas de Base de Datos**". Editorial Prentice Hall. (1999).

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

TÍTULO	DESARROLLO DE UN SOFTWARE QUE PERMITA LA AUTOMATIZACIÓN DE LAS ACTIVIDADES ADMINISTRATIVAS ASOCIADAS A LA SALA DE LECTURA DEL DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS DEL NÚCLEO DE ANZOÁTEGUI DE LA UNIVERSIDAD DE ORIENTE
SUBTÍTULO	

AUTOR (ES):

APELLIDOS Y NOMBRES	CÓDIGO CULAC / E MAIL
Amer M., Sair D.	**CVLAC: V-15.875.692** **E MAIL: sairdaniel@yahoo.com**
Saavedra S., Oscar R.	**CVLAC: V-17.870.119** **E MAIL: oscarrss@hotmail.com**
	CVLAC: **E MAIL:**
	CVLAC: **E MAIL:**

PALÁBRAS O FRASES CLAVES:

SOFTWARE
AUTOMATIZACIÓN
SALA DE LECTURA
ACTIVIDADES ADMINISTRATIVAS
COMPUTACION
JAVA
POSTGRESQL

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ÀREA	SUBÀREA
Ingeniería y Ciencias Aplicadas	Ingeniería en Computación

RESUMEN (ABSTRACT):

La Sala de Lectura del Departamento de Computación y Sistemas de la Universidad de Oriente sirve como fuente de información más especializada que la ofrecida por la biblioteca central a todos los estudiantes que hacen vida en el Departamento de Computación y Sistemas. En ella se realizan una cantidad de procesos administrativos para el manejo de la información de los estudiantes, así como de los libros y tesis que en ella se encuentran. Dichos procesos se realizan en la actualidad de manera manual, lo cual conlleva una dificultad mayor, así como también una mayor inversión de tiempo y esfuerzo por parte de los ayudantes y colaboradores de la sala. Debido al incremento en la cantidad de estudiantes que ingresan cada semestre al Departamento se hace imperiosa la necesidad de automatizar el manejo de los procesos administrativos que se realizan en la Sala de Lectura mediante el desarrollo e implementación de un software, para así optimizar su funcionamiento y maximizar su calidad de servicio para los estudiantes. Este software fue desarrollado utilizando Java como lenguaje de programación y PostgreSQL como sistema gestor de base de datos.

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

CONTRIBUIDORES:

APELLIDOS Y NOMBRES	ROL / CÓDIGO CVLAC / E_MAIL				
Anato, Julima	ROL	CA	AS X	TU	JU
	CVLAC:	V-8.259.495			
	E_MAIL	Anajul71@hotmail.com			
	E_MAIL				
Cortinez, Claudio	ROL	CA	AS	TU X	JU
	CVLAC:	V-12.155.334			
	E_MAIL	clcortinez@gmail.com			
	E_MAIL				
Torrealba, Aquiles	ROL	CA	AS	TU	JU X
	CVLAC:	V-7.385.840			
	E_MAIL	tmar1966@hotmail.com			
	E_MAIL				
	ROL	CA	AS	TU	JU X
	CVLAC:				
	E_MAIL				
	E_MAIL				

FECHA DE DISCUSIÓN Y APROBACIÓN:

2009	12	04
AÑO	MES	DÍA

LENGUAJE. SPA

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ARCHIVO (S):

NOMBRE DE ARCHIVO	TIPO MIME
Tesis SIGESLEC.doc	Aplication/msword

CARACTERES EN LOS NOMBRES DE LOS ARCHIVOS: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z. a b c d e f g h i j k l m n o p q r s t u v w x y z. 0 1 2 3 4 5 6 7 8 9.

ALCANCE

ESPACIAL: _____ (OPCIONAL)

TEMPORAL: _____ (OPCIONAL)

TÍTULO O GRADO ASOCIADO CON EL TRABAJO:

Ingeniero en Computación

NIVEL ASOCIADO CON EL TRABAJO:

Pre-Grado

ÁREA DE ESTUDIO:

Departamento de Computación y Sistemas

INSTITUCIÓN:

Universidad de Oriente – Núcleo de Anzoátegui

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

DERECHOS

De acuerdo con el artículo 41 del reglamento de trabajo de grado:

"Los trabajos de grado son de exclusiva propiedad de la Universidad de Oriente y sólo podrán ser utilizados a otros fines con el consentimiento del consejo de núcleo respectivo, quien lo participará al Consejo Universitario".

AUTOR	AUTOR	AUTOR
Amer M., Sair D.	Saavedra S., Oscar R.	

TUTOR	JURADO	JURADO
Anato, Julima	Torrealba, Aquiles	Cortinez, Claudio

POR LA SUBCOMISIÓN DE TESIS