**“Año de la Recuperación y Consolidación de la Economía Peruana”**

Icono

Descripción generada automáticamente con confianza media

**FACULTAD DE INGENIERÍA**

**Algoritmos y Estructura de Datos**

**Título del Proyecto:**

**“Sistema de Gestión de Biblioteca Digital para Librería Crisol”**

**DOCENTE:**

Palomino Vidal Carlos Efraín

**GRUPO:** N.° 2

**INTEGRANTES:**

Sanchez Rios Jheremy Ayrton -U21205138

Hurtado Lorenzo Ronal Dany -U20238833

Sanchez Morales Jairo -U23219328

Yanac Gutiérrez Julio Gabriel -U23247082

**Lima, 2025**

**Tabla de contenido**

[1 Introducción 6](#_Toc203607713)

[2 Capítulo 1 7](#_Toc203607714)

[2.1 Aspectos Generales 7](#_Toc203607715)

[2.2 Organigrama del Sistema 8](#_Toc203607716)

[2.3 Misión y Visión 10](#_Toc203607717)

[2.3.1 Misión 10](#_Toc203607718)

[2.3.2 Visión 10](#_Toc203607719)

[2.4 Objetivos Estratégicos 10](#_Toc203607720)

[2.5 Problemática 11](#_Toc203607721)

[2.6 Alternativa de Solución 11](#_Toc203607722)

[2.7 Propuesta Elegida 12](#_Toc203607723)

[3 Capítulo 2 13](#_Toc203607724)

[3.1 Estado del Arte 13](#_Toc203607725)

[3.2 Marco Teórico 14](#_Toc203607726)

[4 Capítulo 3 15](#_Toc203607727)

[4.1 Alcances 15](#_Toc203607728)

[4.2 Requerimientos 16](#_Toc203607729)

[4.3 Restricciones 16](#_Toc203607730)

[4.4 Diagrama de clases 18](#_Toc203607731)

[4.5 Prototipo de Interfaces 24](#_Toc203607732)

[5 Capítulo 4 25](#_Toc203607733)

[5.1 Diseño de Aplicación 25](#_Toc203607734)

[5.1.1 Modelo 25](#_Toc203607735)

[5.1.2 Vista 26](#_Toc203607736)

[5.1.3 Controlador 27](#_Toc203607737)

[5.1.4 Ejecutable 27](#_Toc203607738)

[5.2 Funcionalidades Implementadas 27](#_Toc203607739)

[5.2.1 Módulo de Gestión de Libros 27](#_Toc203607740)

[5.2.2 Módulo de Carrito de Compras 28](#_Toc203607741)

[5.2.3 Módulo de Historial 28](#_Toc203607742)

[5.2.4 Módulo de Préstamos 28](#_Toc203607743)

[5.2.5 Módulo de Ciclo de Lectura 28](#_Toc203607744)

[5.2.6 Módulo de Búsqueda 29](#_Toc203607745)

[5.2.7 Módulo de Categorías 29](#_Toc203607746)

[5.2.8 Módulo de Usuarios (GestorUsuarios) 29](#_Toc203607747)

[5.3 Estructura de Paquetes 30](#_Toc203607748)

[5.4 Clases Principales 30](#_Toc203607749)

[6 Capítulo 5 32](#_Toc203607750)

[6.1 Conclusiones 32](#_Toc203607751)

[6.2 Recomendaciones 32](#_Toc203607752)

[6.3 Glosario de términos 33](#_Toc203607753)

[7 Referencias 34](#_Toc203607754)

**Tabla de figuras**

[**Figura 1** Diagrama UML de lista enlazada simple 18](#_Toc203607755)

[**Figura 2** Diagrama UML de lista doble 19](#_Toc203607756)

[**Figura 3** Diagrama UML de lista circular 20](#_Toc203607757)

[**Figura 4** Diagrama UML de pila 21](#_Toc203607758)

[**Figura 5** Diagrama UML de cola 22](#_Toc203607759)

[**Figura 6** Diagrama UML de árbol 23](#_Toc203607760)

[**Figura 7** Prototipo de Interfaz de Usuario: Pantalla de Inicio de Sesión 24](#_Toc203607761)

[**Figura 8** Prototipo de Interfaz de Usuario: Gestor de Biblioteca Digital 24](#_Toc203607762)

[**Figura 9** Estructura del Sistema de Gestión de Biblioteca 30](#_Toc203607763)

# Introducción

En el Perú, muchas pequeñas y medianas empresas, como la cadena de librerías Crisol, aún dependen de registros físicos para gestionar su inventario de libros. Este método manual, aunque sencillo, presenta grandes limitaciones: es propenso a errores humanos, dificulta la búsqueda eficiente de ejemplares y consume tiempo valioso en tareas repetitivas como el ordenamiento y la consulta de inventario. En un contexto donde la competitividad exige agilidad y precisión, la falta de digitalización se convierte en una desventaja, especialmente para negocios que manejan grandes volúmenes de registros.

Ante esta problemática, surge la necesidad de soluciones tecnológicas accesibles que optimicen la gestión sin requerir grandes inversiones. Este proyecto propone un Sistema de Biblioteca Digital, desarrollado en Java, diseñado para reemplazar los registros en papel en Librería Crisol. La aplicación permite no solo almacenar datos de libros, sino también ordenarlos automáticamente, realizar búsquedas instantáneas e implementar un carrito de compras para realizar ventas, mejorando la productividad del negocio. Al combinar estructuras de datos, como listas enlazadas, pilas, colas y árboles, con algoritmos de ordenamiento, el sistema ofrece una alternativa viable para modernizar operaciones sin complejidades técnicas innecesarias.

# Capítulo 1

## Aspectos Generales

El Sistema de Biblioteca Digital para Librería Crisol ahora incluye un módulo de ventas con carrito de compras implementado con lista doblemente enlazada, permitiendo:

* Digitalización de registros: Sustituye los archivos físicos por una base de datos digital.
* Operaciones automatizadas: Inserción rápida de nuevos títulos, eliminación inmediata de ejemplares no existentes, búsqueda instantánea por título y ordenamiento automático por múltiples criterios.
* Nuevas tecnologías empleadas: Listas enlazadas, algoritmos de ordenamiento y una interfaz gráfica intuitiva desarrollada en Java Swing.
* Gestión eficiente de ítems en un carrito de compras para agregar/eliminar y cálculo automático de totales y subtotales.

## Organigrama del Sistema

SISTEMA DE BIBLIOTECA DIGITAL

├── Modulo de Login

│ └── Autenticación de usuario

│

├── Módulo de Registro

│ ├── Agregar libro (inicio/final)

│ └── Eliminar libro

│ └── Ordenamiento Por título (Bubble Sort)

│ └── Ordenamiento Por autor (Merge Sort)

│ └── Ordenamiento Por año (Bubble Sort)

│

├── Módulo de Prestamos

│ ├── Buscar y añadir al historial

│ ├── Solicitar libro

│ ├── Ver cola de préstamo

│ └── Procesar solicitud

│

├── Módulo de Historial

│ └── Historial

│

├── Módulo de Categorías

│ └── Archivos de categorías

│

├── Módulo de CicloLectura

│ └── Agregar al ciclo

│ └── Siguiente Libro

│ └── Libro Anterior

│ └── Ordenar por titulo

│

├── Módulo de ArbolBinario

│ └── Agregar libro

│ └── Buscar en árbol

│ └── Mostar libros

│

└── Módulo de Carrito

├── Agregar al carrito

├── Eliminar del carrito

├── Ver carrito

└── Pagar carrito

Cada módulo se integra mediante una arquitectura MVC (Modelo-Vista-Controlador)

## Misión y Visión

### Misión

Optimizar la gestión de inventario de Librería Crisol mediante una solución tecnológica accesible que elimine procesos manuales, reduzca errores y mejore la eficiencia operativa, contribuyendo a una mejor experiencia de clientes y colaboradores.

### Visión

Convertirnos en el sistema de gestión bibliográfica estándar para pequeñas y medianas librerías peruanas, siendo reconocidos por nuestra simplicidad, eficiencia y adaptabilidad a diversos modelos de negocio del rubro.

## Objetivos Estratégicos

* Reducir el tiempo dedicado al registro de nuevos títulos, búsqueda de ejemplares y ordenamiento de inventario.
* Reducir los errores por registro manual, manteniendo un registro actualizado en tiempo real de los ejemplares disponibles.
* Adaptar la aplicación para incorporar nuevos criterios de búsqueda/ordenamiento y permitir el manejo de más registros sin perder el rendimiento.
* Reducir el uso de papel en la gestión de inventario y disminuir costos operativos.
* Gestionar eficientemente un carrito de compras y calcular en automático las ventas totales realizadas.

## Problemática

La Librería Crisol enfrenta desafíos significativos en la gestión de su inventario de libros debido a su dependencia de registros manuales en papel. Entre los principales problemas identificados se encuentran:

* Ineficiencia en búsquedas: Los empleados pierden tiempo valioso revisando archivos físicos para localizar un libro específico.
* Errores humanos en registros: La digitación manual de datos genera inconsistencias (errores de escritura, duplicación de registros, omisiones).
* Ordenamiento manual lento: Organizar libros por criterios como título, autor o año de publicación requiere tiempo y esfuerzo.
* Falta de escalabilidad: A medida que crece el catálogo de libros, los métodos tradicionales se vuelven insostenibles.

## Alternativa de Solución

Alternativa 1: Sistema de Hojas de Cálculo (Excel) Requiere un bajo costo de implementación y es un programa muy conocido entre los usuarios.

Alternativa 2: Software de Gestión Comercial (ERP Genérico) Cuenta con una automatización avanzada de inventario y además integración con otros módulos (ventas, proveedores, etc.).

Alternativa 3: Sistema Personalizado en Java, Programa adaptado a las necesidades específicas del cliente, además cuenta con algoritmos de ordenamiento eficientes, tiene una interfaz intuitiva y un bajo costo.

## Propuesta Elegida

Se optó por desarrollar el Sistema de Biblioteca Digital en Java por las siguientes razones:

* **Eficiencia probada:** Las listas enlazadas permiten manejar dinámicamente altos volúmenes de libros sin pérdida de rendimiento y la búsqueda instantánea localiza un libro en segundos.
* **Automatización de procesos:** El ordenamiento automático y la implementación escalable facilita la adaptación de nuevos campos.
* **Reducción de costos operativos:** Disminuye la necesidad de personal dedicado solo a inventario y elimina el papel usado en registros.

# Capítulo 2

## Estado del Arte

En la actualidad, la gestión eficiente de librerías requiere herramientas tecnológicas que permitan controlar inventarios, ventas y clientes de manera automatizada. Según un estudio de Grand View Research (2023), el mercado global de software para librerías y bibliotecas está en crecimiento, impulsado por la necesidad de digitalizar procesos que antes se manejaban de forma manual. Soluciones como LibraSoft y BookTique son ejemplos de sistemas comerciales que ofrecen módulos de inventario, ventas y reportes, pero suelen estar orientados a medianas y grandes empresas, dejando a las pequeñas librerías con opciones limitadas o adaptaciones de software genérico.

A pesar de los avances, persisten limitaciones en las soluciones existentes. Muchos sistemas no están especializados en librerías, lo que dificulta funciones clave como la búsqueda por ISBN, la gestión de autores o la integración con distribuidores de libros. Además, las opciones gratuitas o de código abierto, como Koha, suelen estar diseñadas para bibliotecas y carecen de módulos de venta y facturación. Esto crea una oportunidad para desarrollar un software accesible, enfocado en las necesidades reales de las tiendas de libros pequeñas y medianas, con características como sincronización en la nube, informes de ventas automatizados y compatibilidad con dispositivos móviles.

En este contexto, el proyecto propuesto busca ofrecer una solución integral, aprovechando tecnologías actuales y centrándose en la usabilidad.

## Marco Teórico

**Listas enlazadas:** Estructura lineal donde cada elemento (nodo) contiene los datos y una referencia al siguiente nodo. Particularmente útiles para inserción/eliminación rápida, manejo dinámico de memoria e implementación sencilla.

**Listas doblemente enlazadas:** Estructura donde cada nodo contiene referencias tanto al nodo anterior como al siguiente, permitiendo navegación bidireccional y eliminación eficiente.

**Listas circulares:** Variante donde el último nodo enlaza de vuelta al primero, formando un ciclo, permitiendo navegación continua sin fin a través de los objetos seleccionados. (Pozo, S. 2001)

**Pilas (LIFO):** Estructura donde el último elemento en entrar es el primero en salir. Implementadas en la clase PilaLibros para gestionar el historial de vistas, permitiendo operaciones push/pop en tiempo constante.

**Colas (FIFO):** Estructura donde el primer elemento en entrar es el primero en salir. Implementadas en la clase ColaLibros para gestionar préstamos.

**Árboles binarios de búsqueda:** Estructura jerárquica donde cada nodo tiene como máximo dos hijos, con ordenamiento izquierda < padre < derecha.

**Algoritmos de ordenamiento:** conjunto de instrucciones o pasos que se utilizan para organizar elementos de una lista o arreglo en un orden específico.

**Bubble Sort:** Algoritmo de ordenamiento simple que funciona comparando e intercambiando elementos adyacentes en una lista hasta que estén en el orden correcto.

**Merge Sort:** Algoritmo que funciona dividiendo una lista en sublistas hasta que cada sublista tenga un solo elemento, luego fusiona estas sublistas ordenadamente para obtener la lista final ordenada.

**Programación orientada a objetos (POO):** Paradigma de programación que organiza el código alrededor de objetos, que combinan datos y métodos (funciones) que operan sobre esos datos.

**Patrón Modelo-Vista-Controlador (MVC):** es un enfoque arquitectónico que divide una aplicación en tres componentes interconectados: el Modelo, la Vista y el Controlador. (Cyberstream, 2024)

**Java Swing:** conjunto de clases en Java que te permite crear interfaces gráficas de usuario (GUI) para aplicaciones de escritorio.

# Capítulo 3

## Alcances

El sistema gestionará el registro básico de libros (título, autor, género y año).

Permitirá ordenar el inventario por título, autor o año usando algoritmos eficientes.

Implementación de un carrito de compra que permite visualizar el detalle de venta antes de la compra.

Incluirá una interfaz gráfica intuitiva para búsquedas y operaciones diarias, también cuenta con validación de datos básicos.

## Requerimientos

Requerimientos funcionales:

* Registro de libros: Gestiona el registro y la clasificación de libros. Para la manipulación de los títulos se ha implementado una Lista Enlazada. Esta estructura, fue seleccionada estratégicamente debido a su eficiencia en operaciones de inserción y eliminación de elementos tanto al inicio como al final de la colección.
* Ordenamiento: Para la clasificación, se han utilizado algoritmos de ordenamiento, tales como Merge Sort y Bubble Sort para organizar los libros según título, autor y año de publicación. La aplicación de estos algoritmos facilita la recuperación y presentación ordenada de la información.
* Búsqueda inmediata y prestamos: Se ha implementado una cola para administrar eficientemente las solicitudes. Además, incorpora un campo de texto de búsqueda general que permite a los usuarios buscar los libros por cualquier método y, al mismo tiempo, añade las consultas realizadas al historial de vistas para futuras referencias.
* Historial: Se implemento una Pila. Ideal para mantener un registro de las búsquedas y elementos visualizados, permitiendo una recuperación en orden inverso al que fue insertado, lo que simula el comportamiento de un historial "último en entrar, primero en salir" orden LIFO.
* Categorias: Se construye utilizando un componente JTree, el cual implementa una estructura de árbol N-ario. Esta nos permite organizar y visualizar de manera jerárquica las distintas categorías, facilitando la navegación y la gestión de la información al tener múltiples ramificaciones desde un nodo.
* Ciclo de lectura: Se implementó utilizando una Lista Circular. Ya que esta estructura permite, agregar libros, avanzar o retroceder libros, ordenarlos asegurando que la navegación sea fluida y que no tenga un fin.
* Carrito de compras: Para esta funcionalidad se implemento una lista doblemente enlazada, para ejecutar operaciones como añadir/eliminar en cualquier punto de la lista, lo que es ideal para gestionar los elementos en el carrito. Además, incluye un historial de ventas.
* Busqueda rápida: Se ha implemento un árbol binario para gestionar y organizar elementos. Permitiendo operaciones básicas como agregar/buscar y mostrar elementos en un recorrido In-Orden, asegurando una visualización ordenada de los datos.
* Visualización: Se implementaron mediante la creación de distintos paneles. Cada panel posee una estructura dedicada, diseñada para presentar la información de manera organizada y clara dentro del proyecto. Para el desarrollo de esta interfaz gráfica de usuario (GUI), se utilizó la biblioteca Java Swing.

Requerimientos no funcionales:

* El sistema debe responder en menos de 1 segundo para operaciones y menos de 3 segundos para ordenamientos con hasta 1,000 libros registrados.
* La aplicación debe ejecutarse sin problemas en Java 11+ y ser compatible con Windows, Linux y macOS.
* El código debe estar documentado con comentarios Javadoc y seguir una estructura modular (MVC) para facilitar actualizaciones.
* Se deben filtrar caracteres especiales en los campos de texto para evitar inyección de código o errores de procesamiento.

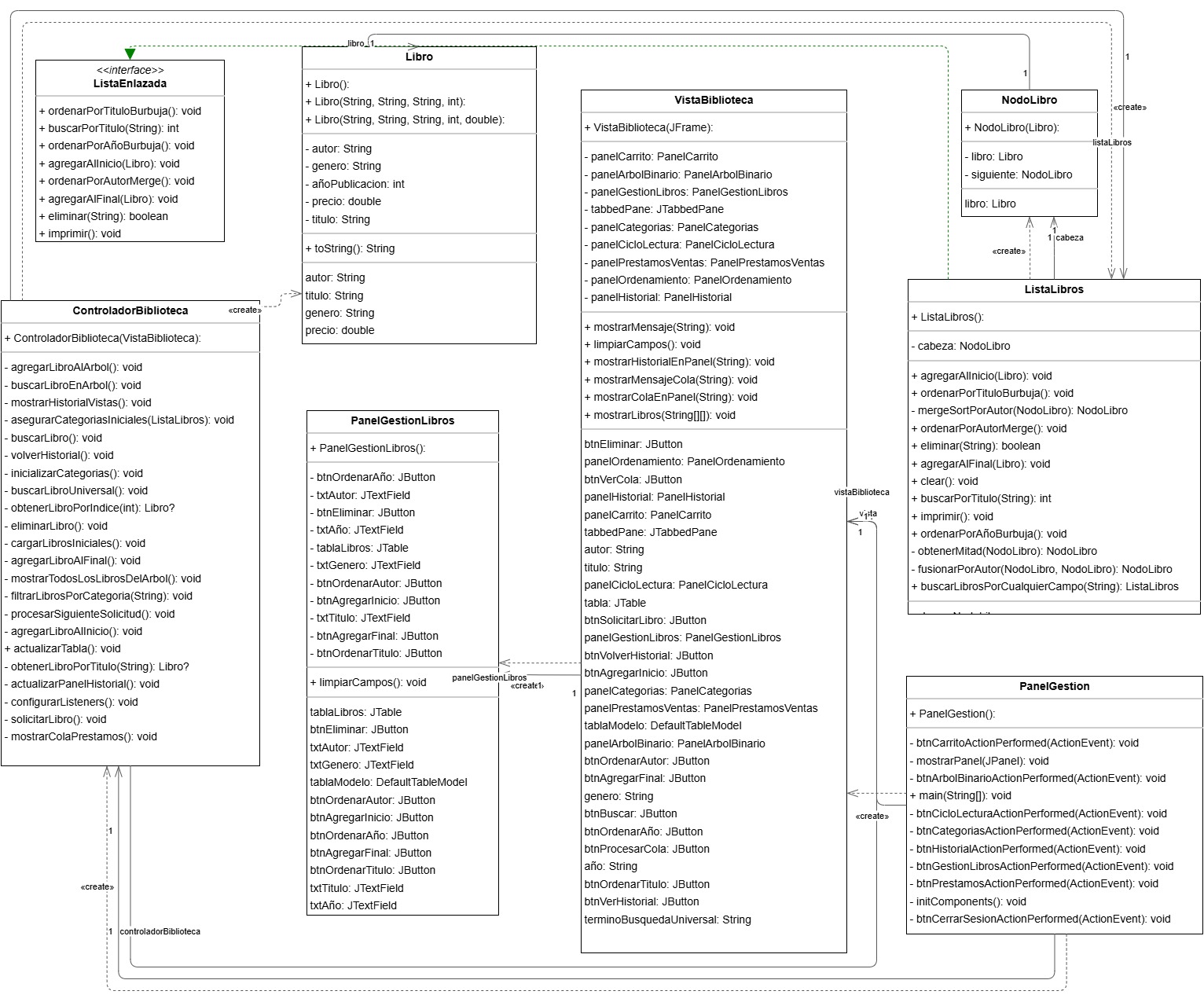
## Restricciones

* El desarrollo se limitará a Java SE 11+ sin bases de datos externas.
* Solo soportará 1 usuario operando el programa.
* El hardware mínimo requerido será 4GB RAM y procesador Intel i3.

## Diagrama de clases

A continuación, se presentan los Diagrama de Clases UML, que ilustran la arquitectura y el diseño del sistema. Diagramas que detallan las clases empleadas, sus atributos, métodos y las relaciones entre ellas, ofreciendo una visión clara de la estructura y la interacción de los componentes.

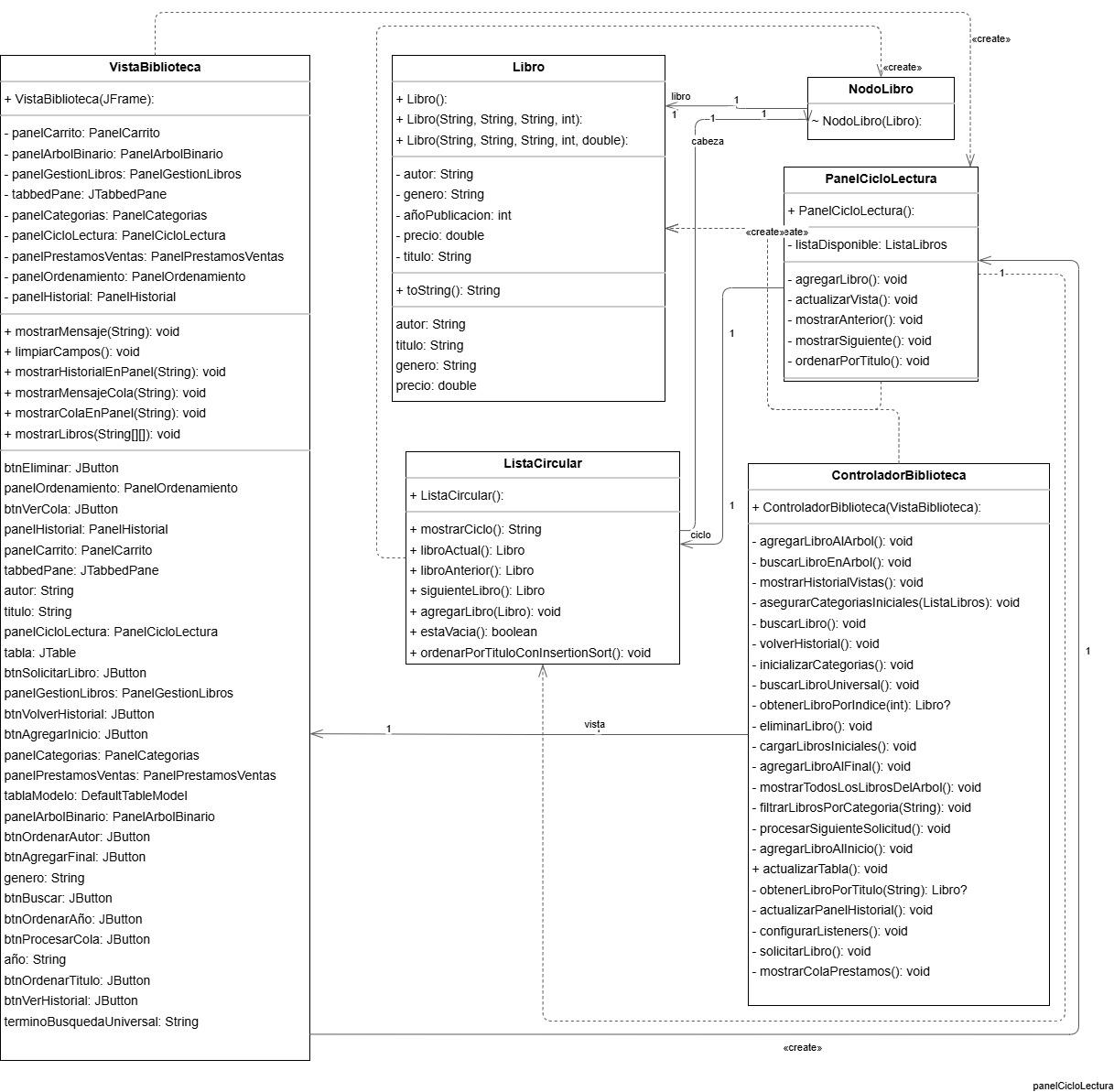
**Figura 1**  
Diagrama UML de lista enlazada simple



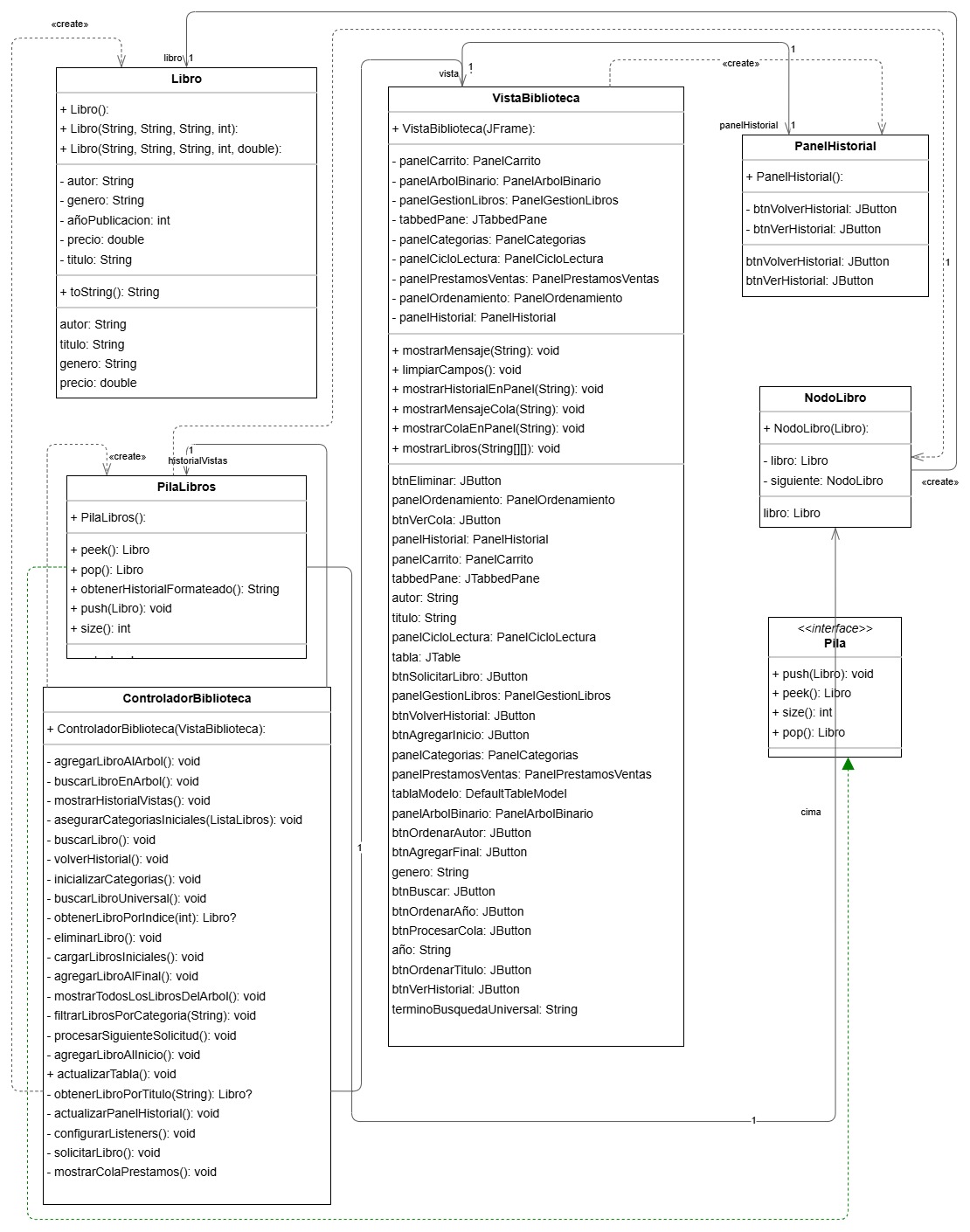
**Figura 2**  
Diagrama UML de lista doble



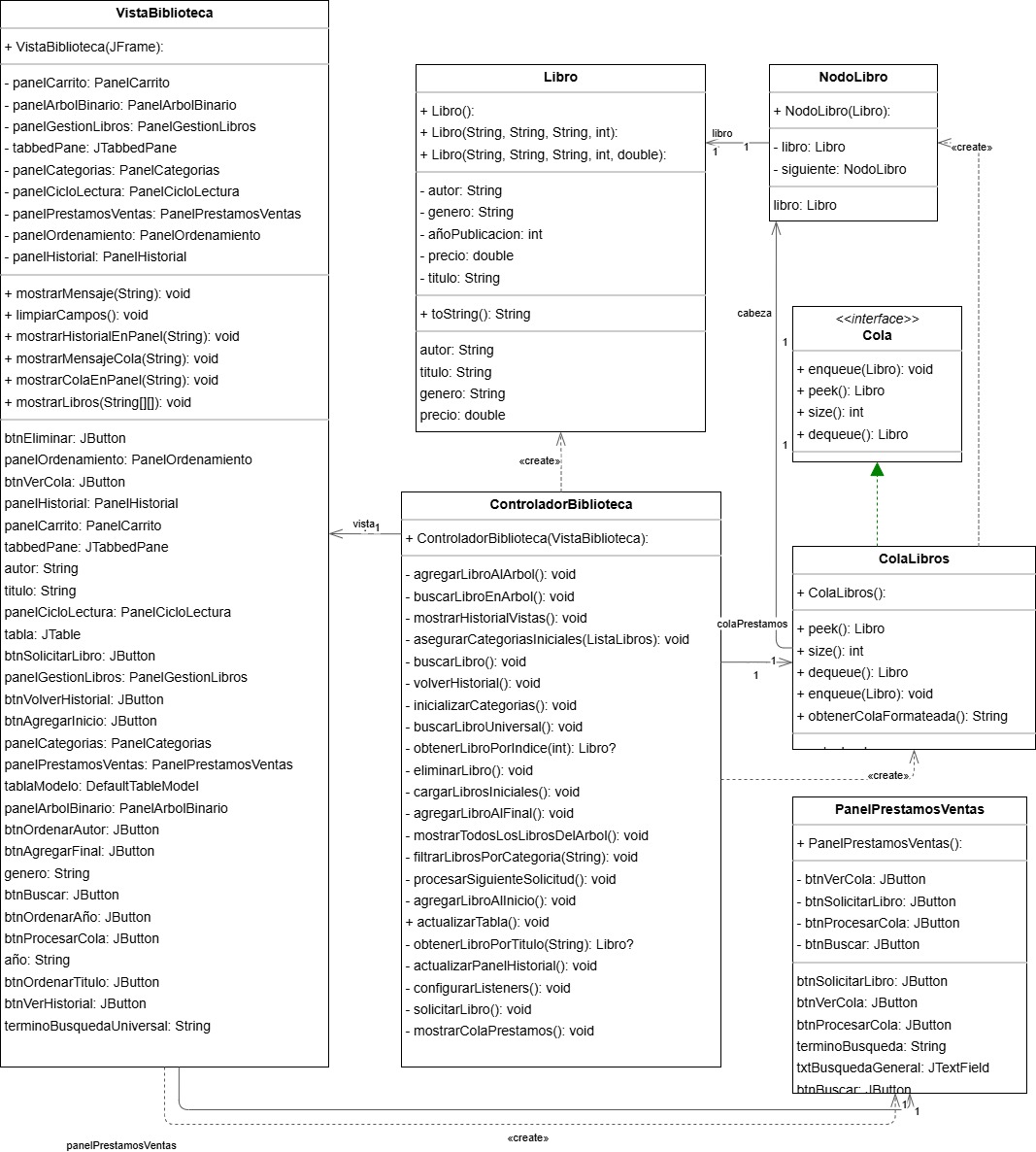
**Figura 3**  
Diagrama UML de lista circular



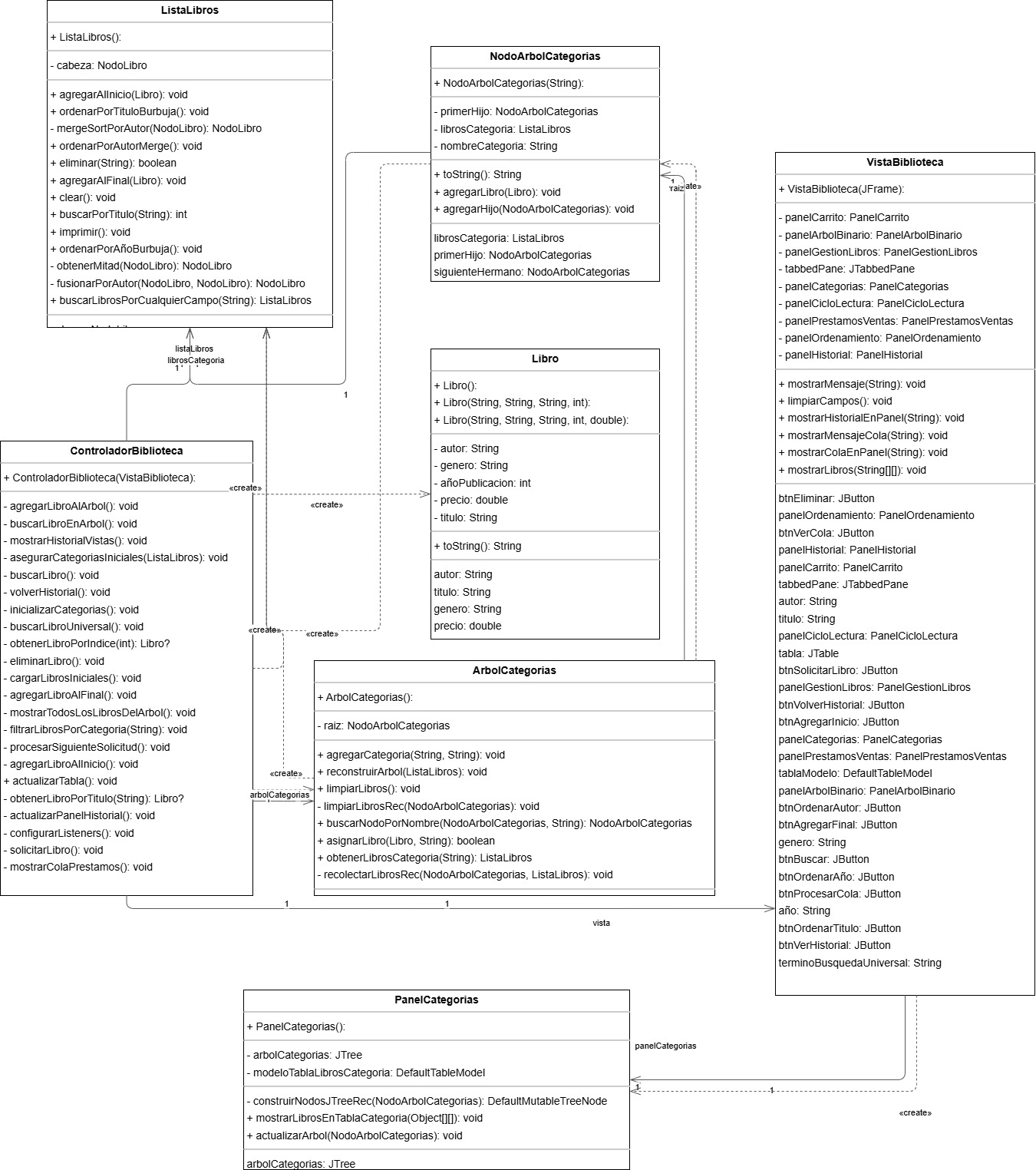
**Figura 4**  
Diagrama UML de pila



**Figura 5**  
Diagrama UML de cola



**Figura 6**  
Diagrama UML de árbol



## Prototipo de Interfaces

Como se puede observar en ***Figura 7*** y ***Figura 8*** son prototipos de interfaces iniciales del Login y de la ventana principal.

**Figura 7**  
Prototipo de Interfaz de Usuario: Pantalla de Inicio de Sesión

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

**Figura 8**  
Prototipo de Interfaz de Usuario: Gestor de Biblioteca Digital

Interfaz de usuario gráfica, Tabla

El contenido generado por IA puede ser incorrecto.

*Nota*. Presentación inicial con funcionalidades básicas.

# Capítulo 4

## Diseño de Aplicación

El sistema sigue una arquitectura MVC (Modelo-Vista-Controlador) para separar claramente la lógica de negocio, la interfaz y la gestión de datos. Se implementó con:

### Modelo

**ListaLibros:** Maneja la estructura de datos (lista enlazada) y algoritmos de ordenamiento.

**ListaEnlazada:** Establece el contrato para las operaciones básicas de la lista.

**Libro:** Clase que almacena los atributos básicos de un libro.

**NodoLibro:** Representa un nodo en una lista enlazada simple, conteniendo un Libro y una referencia al siguiente nodo.

**ListaLibros:** Implementa una lista enlazada simple para gestionar el catálogo de libros, con operaciones básicas.

**ListaDobleCarrito:** Implementa una lista doblemente enlazada para gestionar el carrito de compras.

**NodoItem:** Nodo utilizado en ListaDobleCarrito, que almacena un Libro, cantidad y referencias al nodo anterior y siguiente.

**PilaLibros:** Implementa una pila (LIFO) para el historial de libros vistos.

**ColaLibros:** Implementa una cola (FIFO) para gestionar préstamos de libros.

**ListaCircular:** Implementa una lista circular para el ciclo de lectura.

**ArbolBinarioLibros:** Implementa un árbol binario de búsqueda para búsquedas eficientes por título.

**ArbolCategorias:** Implementa un árbol n-ario para organizar libros por categorías y subcategorías.

**NodoArbolBinario:** Nodo utilizado en ArbolBinarioLibros, con referencias a los hijos izquierdo y derecho.

**NodoArbolCategorias:** Nodo utilizado en ArbolCategorias, con capacidad de almacenar múltiples hijos.

**Usuario:** Almacena información de usuarios para el sistema de autenticación.

**GestorUsuarios:** Gestiona el registro y autenticación de usuarios.

### Vista

**VistaBiblioteca:** Ventana principal que integra todos los paneles y muestra la tabla de libros.

**PanelGestionLibros:** Panel para agregar, eliminar y visualizar libros en el catálogo. Además de ordenamiento de libros según título, autor y fecha de publicación.

**PanelGestion:** Frame padre que alberga todos los paneles dentro de un JPanel para una vista mas estética.

**PanelPrestamosVentas:** Panel para gestionar préstamos y búsquedas de libros.

**PanelHistorial:** Panel para visualizar y navegar en el historial de libros vistos.

**PanelCategorias:** Panel que muestra el árbol de categorías de libros en un JTree.

**PanelCicloLectura:** Panel para gestionar el ciclo de lectura con ListaCircular.

**PanelArbolBinario:** Panel para interactuar con el árbol binario de búsqueda.

**PanelCarrito:** Panel para gestionar el carrito de compras.

**Login:** Ventana de inicio de sesión que autentica usuarios mediante GestorUsuarios.

### Controlador

**ControladorBiblioteca:** Coordina las interacciones entre VistaBiblioteca y el modelo.

**ControladorCarrito:** Gestiona la lógica del carrito de compras, conectando PanelCarrito con ListaDobleCarrito.

### Ejecutable

**Main:** Clase principal que inicia la aplicación, mostrando primero el Login y luego la VistaBiblioteca.

## Funcionalidades Implementadas

### Módulo de Gestión de Libros

**agregarAlInicio()**: Inserción de un libro al inicio de la lista.

**agregarAlFinal():** Inserción de un libro al final de la lista.

**eliminar():** Elimina un libro por título, usando búsqueda lineal.

**ordenarPorTituloBurbuja():** Ordena los libros por título usando Bubble Sort.

**ordenarPorAutorMerge():** Ordena los libros por autor usando Merge Sort.

**ordenarPorAñoBurbuja():** Ordena los libros por año usando Bubble Sort.

**buscarPorTitulo():** Búsqueda lineal por título exacto.

**buscarLibrosPorCualquierCampo():** Búsqueda flexible por título, autor, género o año.

### Módulo de Carrito de Compras

**agregarItem():** Agrega un libro al carrito.

**eliminarItem():** Elimina un libro del carrito por título.

**actualizarCantidad():** Modifica la cantidad de un libro en el carrito.

**calcularTotal():** Calcula el total a pagar en el carrito.

**procesarPago():** Confirma la compra, vacía el carrito y acumula ventas.

### Módulo de Historial

**push():** Agrega un libro al tope del historial.

**pop():** Elimina y devuelve el libro más reciente.

**peek():** Muestra el libro en el tope sin eliminarlo.

**obtenerHistorialFormateado():** Devuelve el historial ordenado.

### Módulo de Préstamos

**enqueue():** Agrega un libro a la cola de préstamos.

**dequeue():** Atiende el préstamo más antiguo.

**peek():** Muestra el próximo libro a procesar sin eliminarlo.

**obtenerColaFormateada():** Devuelve la cola de préstamos en orden FIFO.

### Módulo de Ciclo de Lectura

**agregarLibro():** Añade un libro al ciclo.

**siguienteLibro():** Avanza al siguiente libro en el ciclo.

**libroActual():** Muestra el libro actual sin avanzar.

**ordenarPorTituloConInsertionSort():** Ordena el ciclo por título usando Insertion Sort.

### Módulo de Búsqueda

**insertar():** Añade un libro al árbol binario de búsqueda.

**buscar():** Encuentra un libro por título.

**obtenerLibrosInorden():** Devuelve los libros ordenados por título.

### Módulo de Categorías

**agregarCategoria():** Añade una categoría o subcategoría al árbol.

**asignarLibro():** Asocia un libro a una categoría.

**obtenerLibrosCategoria():** Devuelve todos los libros de una categoría y sus subcategorías.

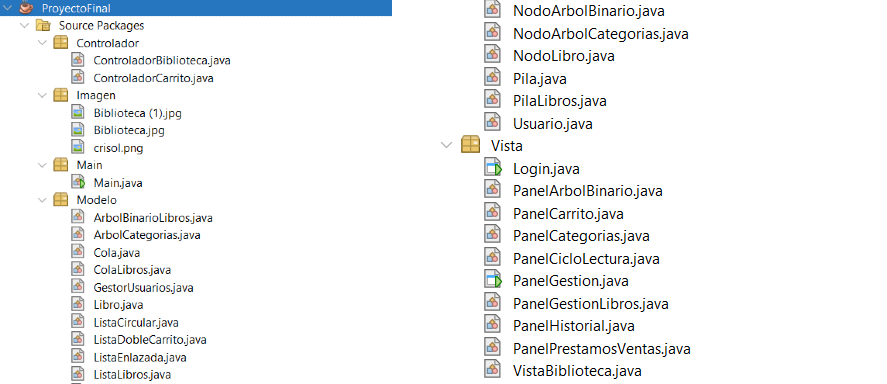
### Módulo de Usuarios (GestorUsuarios)

**agregarUsuario():** Registra un nuevo usuariocon.

**validarCredenciales():** Verifica usuario y contraseña.

## Estructura de Paquetes

**Figura 9**  
Estructura del Sistema de Gestión de Biblioteca



*Nota.* Se observan directorios como Controlador, Imagen, Main, Modelo y Vista, siguiendo el patrón Modelo-Vista-Controlador (MVC).

## Clases Principales

**Clase Libro:**

Almacena atributos básicos (título, autor, género, año).

Getters/Setters para cada campo.

toString() para representación en texto.

**Clase NodoLibro:**

Nodo de la lista enlazada.

Libro libro: Datos del libro.

NodoLibro siguiente: Enlace al próximo nodo.

**Clase ListaLibros:**

Gestiona operaciones sobre la lista.

agregarAlInicio() / agregarAlFinal(): O(1).

ordenarPorAutorMerge(): Divide y vencerás recursivo.

**Clase VistaBiblioteca:**

JTable tabla: Muestra los libros.

JButton btnOrdenarTitulo: Ejecuta Bubble Sort.

Clase ControladorBiblioteca:

Puente entre vistas y modelo

# Capítulo 5

## Conclusiones

* El sistema demuestra que estructuras de datos simples (listas enlazadas) resuelven necesidades reales de Pymes dedicadas a la venta de libros con bajo costo técnico.
* Los algoritmos de ordenamiento elegidos (Bubble Sort y Merge Sort) optimizan operaciones relevantes, manteniendo un código escalable.
* La interfaz Java Swing cumple su propósito al requerir poco tiempo de capacitación para usuarios no técnicos.
* La implementación del carrito con lista doblemente enlazada demostró una mayor eficiencia en operaciones frecuentes (agregar/eliminar ítems) y facilidad para implementar navegación en ambas direcciones
* La inclusión del módulo de ventas amplía el alcance del sistema de gestión bibliográfica a un sistema completo de punto de venta para librerías.

## Recomendaciones

* Migrar a una interfaz gráfica más avanzada (JavaFX) en futuras versiones para interfaces más modernas y responsivas.
* Añadir persistencia usando una base de datos para evitar pérdida de datos al cerrar la aplicación.
* Añadir funcionalidad de "guardar carrito" para compras posteriores
* Implementar búsqueda en el carrito para grandes cantidades de ítems

## Glosario de términos

Estructura de Datos: Organización y almacenamiento de información.

Algoritmo: Secuencia de pasos para resolver un problema.

Complejidad Computacional (Big O): Medida de eficiencia de un algoritmo.

Interfaz Gráfica (GUI): Parte visual del sistema con la que interactúa el usuario (botones, tablas).

Java Swing: Biblioteca de Java para crear interfaces gráficas sencillas.

Modelo-Vista-Controlador (MVC): Patrón de diseño que separa lógica, interfaz y control.

Lista Enlazada Simple: Estructura lineal donde cada nodo contiene datos y una referencia al siguiente nodo. Ideal para inserciones/eliminaciones rápidas.

Lista Doblemente Enlazada: Estructura donde cada nodo tiene referencias al nodo anterior y siguiente, permitiendo recorrido bidireccional.

Pila (LIFO): Estructura donde el último elemento en entrar es el primero en salir, usada para gestión de historiales.

Cola (FIFO): Estructura donde el primer elemento en entrar es el primero en salir, ideal para manejo de préstamos en orden de llegada.

Lista Circular: Variante de lista enlazada donde el último nodo enlaza al primero, formando un ciclo continuo.

Árbol Binario de Búsqueda: Estructura jerárquica donde cada nodo tiene hasta dos hijos, optimizada para búsquedas rápidas.

Persistencia: Guardar datos entre sesiones (no implementado en esta versión).

Validación de Entradas: Verificar que los datos ingresados cumplan formatos esperados

Thread-Safe: Capacidad de funcionar con múltiples usuarios

JTable: Componente de Swing para mostrar datos en formato de tabla

# Referencias

Goodrich, M. T., Tamassia, R., & Goldwasser, M. H. (2014). Data structures and algorithms in Java (6th ed.). Wiley.

Lily, H., & Gabriel, F. (2024, June 10). *Estructura de datos de lista*. StudySmarter. <https://www.studysmarter.es/resumenes/ciencias-de-la-computacion/estructuras-de-datos/estructura-de-datos-de-lista/>

Cyberstream. (2024, May 25). *Guía completa sobre MVC: Definición y funcionamiento detallado*. Byron Vargas ®; Byron Vargas. <https://www.byronvargas.com/web/que-es-mvc-y-como-funciona/>

Zukowski, J. (2005). The definitive guide to Java Swing (3rd ed.). Apress.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design patterns: Elements of reusable object-oriented software. Addison-Wesley.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to algorithms (3rd ed.). MIT Press.

Shneiderman, B., Plaisant, C., Cohen, M., Jacobs, S., Elmqvist, N., & Diakopoulos, N. (2018). Designing the user interface: Strategies for effective human-computer interaction (6th ed.). Pearson.

Pozo, S. (2001). Estructuras de datos. Universidad Nacional Autónoma de México. Recuperado de <https://conclase.net/nosotros>

Cedeño Zambrano, K. C., Lucas Alay, D. D., Verá Polanco, L. A., Vera Zambrano, J. M., & Romero Castro, V. F. (2022). Importancia de árboles binarios en la programación. Journal TechInnovation, 1(2), 33–42. <https://doi.org/10.47230/Journal.TechInnovation.v1.n2.2022.33-42>

Ortega, O. B., & Garro, D. Á. (2021). Guía práctica del curso “Estructuras de datos y algoritmos”: Listas. *INNOVACIÓN UNIVERSITARIA*, *3*(1).

Guerra, Y. A., Graverán, J. Á. L., Domínguez, A. D., & Martínez, Y. V. Documentación de la Biblioteca de Estructuras de Datos Avanzadas (Listas, Pilas, Colas, Tablas Hash y DCEL).