Documentación del Proyecto Java - Sistema de Gestión para Librería

1. Introducción

El presente documento describe el desarrollo y la estructura del proyecto realizado en Java utilizando NetBeans, con el objetivo de crear un sistema básico de gestión para una librería. Este sistema permite a los usuarios administrar clientes, productos y ventas a través de una interfaz gráfica amigable.

El proyecto está orientado a la gestión de datos, con conexión a una base de datos MySQL llamada **librería** donde se almacenan todas las tablas necesarias para la operación del sistema.

2. Tecnologías y Herramientas Utilizadas

- Lenguaje de programación: Java
- Entorno de desarrollo: NetBeans IDE
- Base de datos: MySQL (base de datos llamada `libreria`)
- Interfaz gráfica: Swing, con componentes JForm y JInternalFrame para modularizar las ventanas
- Conexión a base de datos: JDBC (Java Database Connectivity)
- Patrón DAO: Para la gestión de acceso a datos (ClienteDAO, ProductoDAO, VentasDAO)

3. Estructura General del Proyecto

El sistema tiene una estructura modular, con una ventana principal que funciona como menú y desde la cual se abren formularios para diferentes funcionalidades:

- LoginForm: Pantalla de inicio para autenticación.
- Menú Principal: Acceso a formularios de clientes, productos y ventas.
- Formularios: Implementados con JInternalFrame para facilitar la navegación y organización.

Cada formulario está conectado directamente a una tabla de la base de datos, permitiendo realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar).

4. Descripción de Funcionalidades

4.1 LoginForm

- Permite que el usuario ingrese sus credenciales para acceder al sistema.
- Realiza la validación contra la base de datos.
- En caso de autenticación exitosa, el sistema muestra el menú principal.
- En caso de fallo, muestra un mensaje de error y solicita reintentar.

4.2 Menú Principal

- Desde aquí se puede navegar a:
  - Formulario para agregar clientes.
  - Formulario para agregar productos.
  - Formulario para registrar ventas.
- La navegación se realiza mediante botones que abren los formularios correspondientes en JInternalFrames.

4.3 Gestión de Clientes

- El formulario de clientes permite ingresar nuevos clientes, modificar y consultar información.
- Los datos incluyen nombre, apellido, teléfono, dirección, etc.
- Se conecta con la tabla `clientes` en la base de datos.
- La clase `ClienteDAO` implementa métodos para insertar, actualizar, eliminar y listar clientes.
- Uso de constructores, getters y setters para manipular objetos Cliente.

4.4 Gestión de Productos

- Permite agregar productos con información como nombre, categoría, precio y stock.
- La tabla en la base de datos es `productos`.
- `ProductoDAO` maneja la conexión y manipulación de datos en la tabla.
- Se implementan validaciones para evitar inconsistencias como stock negativo.

4.5 Registro de Ventas

- El formulario permite seleccionar clientes y productos para registrar una venta.
- Se registran detalles como cantidad, fecha y total.
- La información se almacena en la tabla `ventas`.
- La clase `VentasDAO` se encarga de la inserción y consulta de ventas.
- Actualmente, no se logró implementar la funcionalidad para generar reportes filtrados por fechas, la cual queda pendiente para futuras versiones.

5. Conexión a la Base de Datos

- Se creó una clase para manejar la conexión JDBC con la base de datos MySQL `libreria`.
- Esta clase se reutiliza en las clases DAO para ejecutar consultas SQL.

- Se utiliza el patrón DAO para separar la lógica de negocio de la manipulación directa de datos.
- Esto facilita el mantenimiento y escalabilidad del código.

6. Uso de Constructores, Getters y Setters

Para mantener una estructura clara y ordenada en el manejo de los datos, se implementaron en las clases modelo (Cliente, Producto, Venta) los siguientes elementos clave de la programación orientada a objetos:

- **Constructores:**
  Se definieron constructores en cada clase para facilitar la creación de objetos con atributos inicializados, permitiendo instanciar un objeto con los datos necesarios desde el momento de su creación.
- **Getters y Setters:**
  Se utilizaron métodos getters y setters para encapsular los atributos de cada clase. Esto significa que los atributos son privados y solo pueden ser accedidos o modificados mediante estos métodos, asegurando un control sobre cómo se acceden o cambian los datos, manteniendo la integridad del objeto.

Este enfoque permite manejar los datos de forma segura y ordenada, facilitando además la integración con las clases DAO, que utilizan estos objetos para transferir datos entre la aplicación y la base de datos.

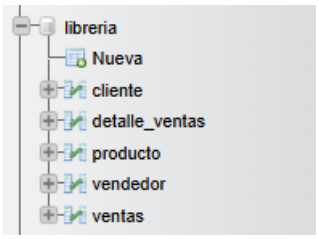
7. Principios y Metodologías Aplicadas

- Programación orientada a objetos, con uso de clases, objetos, encapsulación y abstracción.
- Separación de responsabilidades: interfaz gráfica, lógica de negocio y acceso a datos.
- Manejo de excepciones para robustez y control de errores en conexión y consultas.

8. Limitaciones y Funcionalidades Pendientes

- La generación de reportes por rango de fechas no fue terminada.
- Se planea incluir esta funcionalidad para poder visualizar y exportar reportes detallados de ventas.
- Mejoras en la interfaz para mayor usabilidad.
- Validaciones más completas y manejo de sesiones.

9. Diagrama de la Base de Datos


libreria
Nueva
cliente
detalle_ventas
producto
vendedor
ventas