

Instituto Tecnológico Superior de Xalapa

Tópicos Avanzados de la Programación

Proyecto Final

* Baizabal Acosta Ismael
* Cruz Mendoza Lucero
* Trujillo Trujillo Juan Pablo
* Velásquez López Guadalupe

Docente: Juan Manuel Olguín Medina

04/ Junio/ 2025

**Sistema de Gestión de Estudiantes**

**Proyecto Final**

## **Índice**

[Índice 1](#_Toc618436079)

[Descripción del Proyecto 2](#_Toc486279297)

[Objetivo 2](#_Toc749485251)

[Funcionalidades Principales 2](#_Toc465152436)

[Arquitectura del Sistema 2](#_Toc499038268)

[Patrón de Diseño 2](#_Toc1061096830)

[Componentes del Sistema 3](#_Toc1640761543)

[Tecnologías Utilizadas 3](#_Toc845136890)

[Lenguajes y Frameworks 3](#_Toc1315743702)

[Base de Datos 3](#_Toc487113773)

[Herramientas de Desarrollo 3](#_Toc1078746923)

[Requisitos del Sistema 3](#_Toc1581718427)

[Requisitos de Software 3](#_Toc1767622404)

[Requisitos de Hardware 4](#_Toc2013417257)

[Instalación y Configuración 4](#_Toc796192791)

[Paso 1: Configuración de MySQL 4](#_Toc1764114629)

[Paso 2: Configuración de la Aplicación Java 5](#_Toc1843320480)

[Paso 3: Compilar y Ejecutar 5](#_Toc377581199)

[Estructura del Proyecto 5](#_Toc228732646)

[Manual de Usuario 6](#_Toc1091656508)

[Inicio de la Aplicación 6](#_Toc426039778)

[Funcionalidades Disponibles 6](#_Toc2050208776)

[1. Agregar Estudiante 6](#_Toc1257343084)

[2. Listar Estudiantes 7](#_Toc333058740)

[3. Actualizar Estudiante 8](#_Toc404071662)

[4. Eliminar Estudiante 8](#_Toc61959311)

[5. Salir 9](#_Toc1780193658)

[Documentación Técnica 9](#_Toc1179841744)

[Clase Estudiante 10](#_Toc297626596)

[Clase EstudianteDAO 11](#_Toc1226940889)

[Clase ConexionBD 12](#_Toc1780283642)

[Interfaces Gráficas 13](#_Toc1278541715)

[Vulnerabilidades Identificadas 14](#_Toc1206258890)

[Recomendaciones de Seguridad 14](#_Toc1970549769)

[Configuración Segura para Producción 14](#_Toc860455947)

[Mantenimiento Preventivo 14](#_Toc1987157388)

## **Descripción del Proyecto**

### **Objetivo**

Este proyecto integra los conocimientos de programación adquiridos durante el semestre en una aplicación Stand-Alone desarrollada en Java que implementa un sistema completo de gestión de estudiantes con operaciones CRUD (Crear, Leer, Actualizar, Eliminar).

### **Funcionalidades Principales**

* **Gestión completa de estudiantes**: Registro, consulta, actualización y eliminación
* **Interfaz gráfica intuitiva**: Desarrollada con Java Swing
* **Validación de datos**: Verificación de campos obligatorios y rangos válidos
* **Base de datos persistente**: Almacenamiento en MySQL
* **Manejo de errores**: Sistema robusto de captura y manejo de excepciones

## **Arquitectura del Sistema**

### **Patrón de Diseño**

El sistema implementa el patrón **MVC (Modelo-Vista-Controlador)** junto con el patrón **DAO (Data Access Object)**:

┌─────────────────┐ ┌─────────────────┐ ┌─────────────────┐  
│ VISTA │ │ CONTROLADOR │ │ MODELO │  
│ │ │ │ │ │  
│ VentanaOpciones │◄──►│ App.java │◄──►│ Estudiante.java │  
│VentanaResultados│ │ │ │ │  
└─────────────────┘ └─────────────────┘ └─────────────────┘  
 │  
 ▼  
 ┌─────────────────┐ ┌─────────────────┐  
 │ ACCESO DATOS │ │ BASE DATOS │  
 │ │ │ │  
 │EstudianteDAO.java│◄──►│ MySQL │  
 │ ConexionBD.java │ │gestion\_estudiantes│  
 └─────────────────┘ └─────────────────┘

### **Componentes del Sistema**

**Capa de Presentación (Vista)**

* VentanaOpciones.java: Interfaz principal con menú de opciones
* VentanaResultados.java: Ventana para mostrar resultados y mensajes

**Capa de Lógica de Negocio (Controlador)**

* App.java: Punto de entrada y coordinación de componentes

**Capa de Datos (Modelo)**

* Estudiante.java: Entidad que representa un estudiante
* EstudianteDAO.java: Operaciones CRUD
* ConexionBD.java: Gestión de conexiones a base de datos

## **Tecnologías Utilizadas**

### **Lenguajes y Frameworks**

* **Java SE 8+**: Lenguaje principal de desarrollo
* **Java Swing**: Framework para interfaz gráfica de usuario
* **JDBC**: API para conectividad con base de datos

### **Base de Datos**

* **MySQL 8.0+**: Sistema de gestión de base de datos relacional
* **MySQL Connector/J**: Driver JDBC para MySQL

### **Herramientas de Desarrollo**

* **NetBeans IDE**: Entorno de desarrollo integrado

## **Requisitos del Sistema**

### **Requisitos de Software**

* Java Development Kit (JDK) 8 o superior
* MySQL Server 8.0 o superior
* MySQL Connector/J (incluido en el proyecto)
* Sistema operativo: Windows

### **Requisitos de Hardware**

* Procesador: 1.5 GHz o superior
* Memoria RAM: 2 GB mínimo, 4 GB recomendado
* Espacio en disco: 500 MB para la aplicación y base de datos
* Resolución de pantalla: 1024x768 mínimo

## **Instalación y Configuración**

### **Paso 1: Configuración de MySQL**

1. **Instalar MySQL Server**

-- Crear la base de datos  
CREATE DATABASE gestion\_estudiantes;  
  
-- Usar la base de datos  
USE gestion\_estudiantes;  
  
-- Crear la tabla estudiantes  
CREATE TABLE estudiantes (  
 id INT PRIMARY KEY,  
 nombre VARCHAR(100) NOT NULL,  
 edad INT NOT NULL CHECK (edad >= 16 AND edad <= 100),  
 carrera VARCHAR(100) NOT NULL  
);  
  
-- Insertar datos de prueba (opcional)  
INSERT INTO estudiantes (id, nombre, edad, carrera) VALUES  
(1, 'Juan Pérez', 20, 'Ingeniería en Sistemas'),  
(2, 'María García', 22, 'Administración'),  
(3, 'Carlos López', 19, 'Ingeniería Civil');

1. **Configurar Usuario y Permisos**

-- Crear usuario (opcional, se puede usar root)  
CREATE USER 'estudiantes\_user'@'localhost' IDENTIFIED BY 'password123';  
  
-- Otorgar permisos  
GRANT ALL PRIVILEGES ON gestion\_estudiantes.\* TO 'estudiantes\_user'@'localhost';  
FLUSH PRIVILEGES;

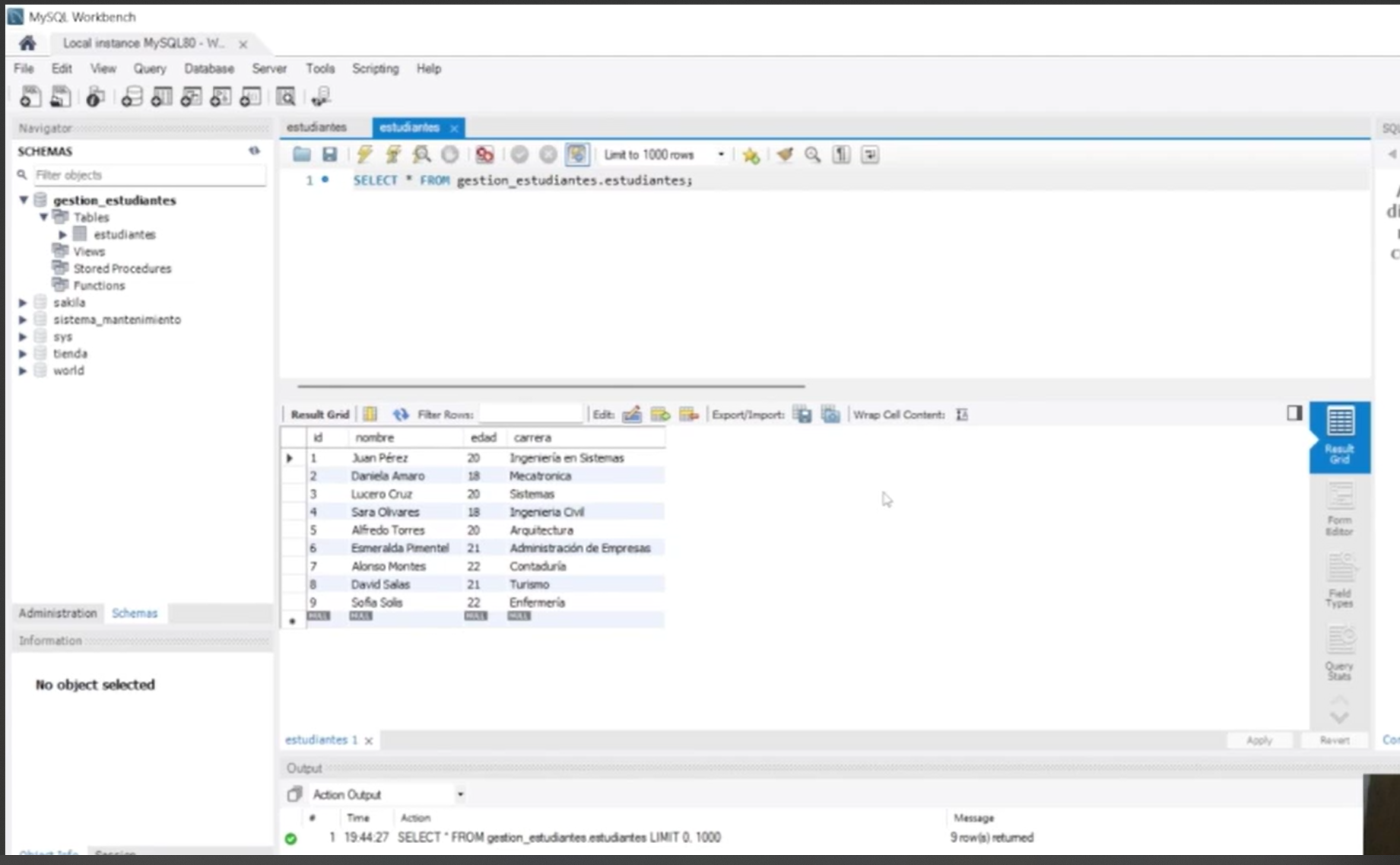
### **Paso 2: Configuración de la Aplicación Java**

1. **Clonar o descargar el proyecto**
2. **Configurar la conexión a base de datos** en ConexionBD.java:

private static final String URL = "jdbc:mysql://localhost:3306/gestion\_estudiantes";  
private static final String USER = "root"; // Cambiar según configuración  
private static final String PASSWORD = "35741"; // Cambiar según configuración

1. **Agregar MySQL Connector/J al classpath**
   1. Descargar desde: <https://dev.mysql.com/downloads/connector/j/>
   2. Agregar el JAR al classpath del proyecto

### **Paso 3: Compilar y Ejecutar**

# Compilar el proyecto  
javac -cp ".:mysql-connector-java.jar" com/mycompany/proyectobd/\*.java  
  
# Ejecutar la aplicación  
java -cp ".:mysql-connector-java.jar" com.mycompany.proyectobd.App  
 

## **Estructura del Proyecto**

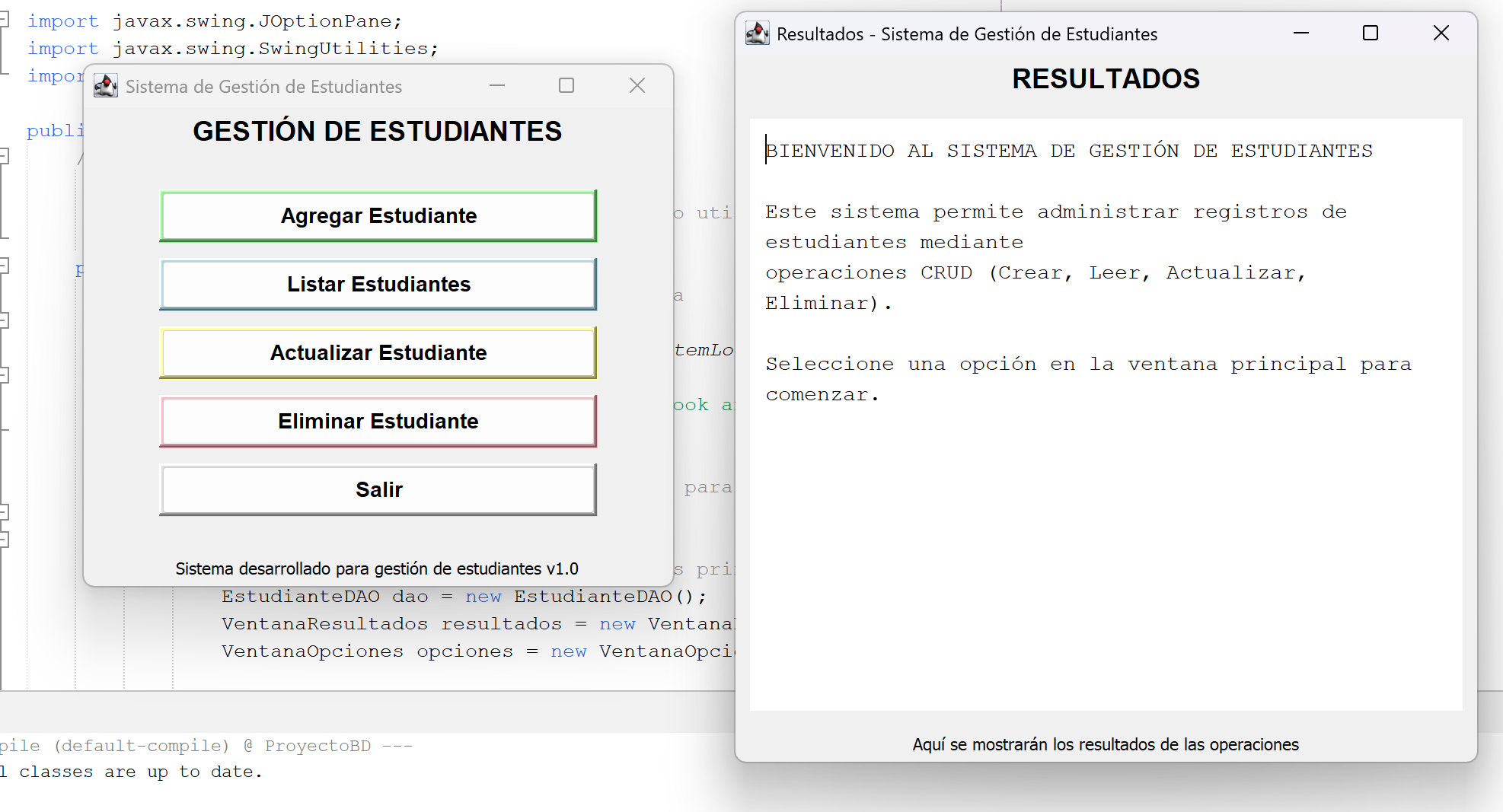
proyectobd/  
│  
├── src/  
│ └── main/  
│ └── java/  
│ └── com/  
│ └── mycompany/  
│ └── proyectobd/  
│ ├── App.java # Punto de entrada  
│ ├── Estudiante.java # Modelo de datos  
│ ├── EstudianteDAO.java # Acceso a datos  
│ ├── ConexionBD.java # Conexión BD  
│ ├── VentanaOpciones.java # Interfaz principal  
│ ├── VentanaResultados.java # Ventana resultados  
│ └── ProyectoBD.java # Archivo base  
│  
├── lib/  
│ └── mysql-connector-java.jar # Driver MySQL  
│  
├── sql/  
│ └── database\_setup.sql # Script de BD  
│  
└── README.md # Documentación

## **Manual de Usuario**

### **Inicio de la Aplicación**

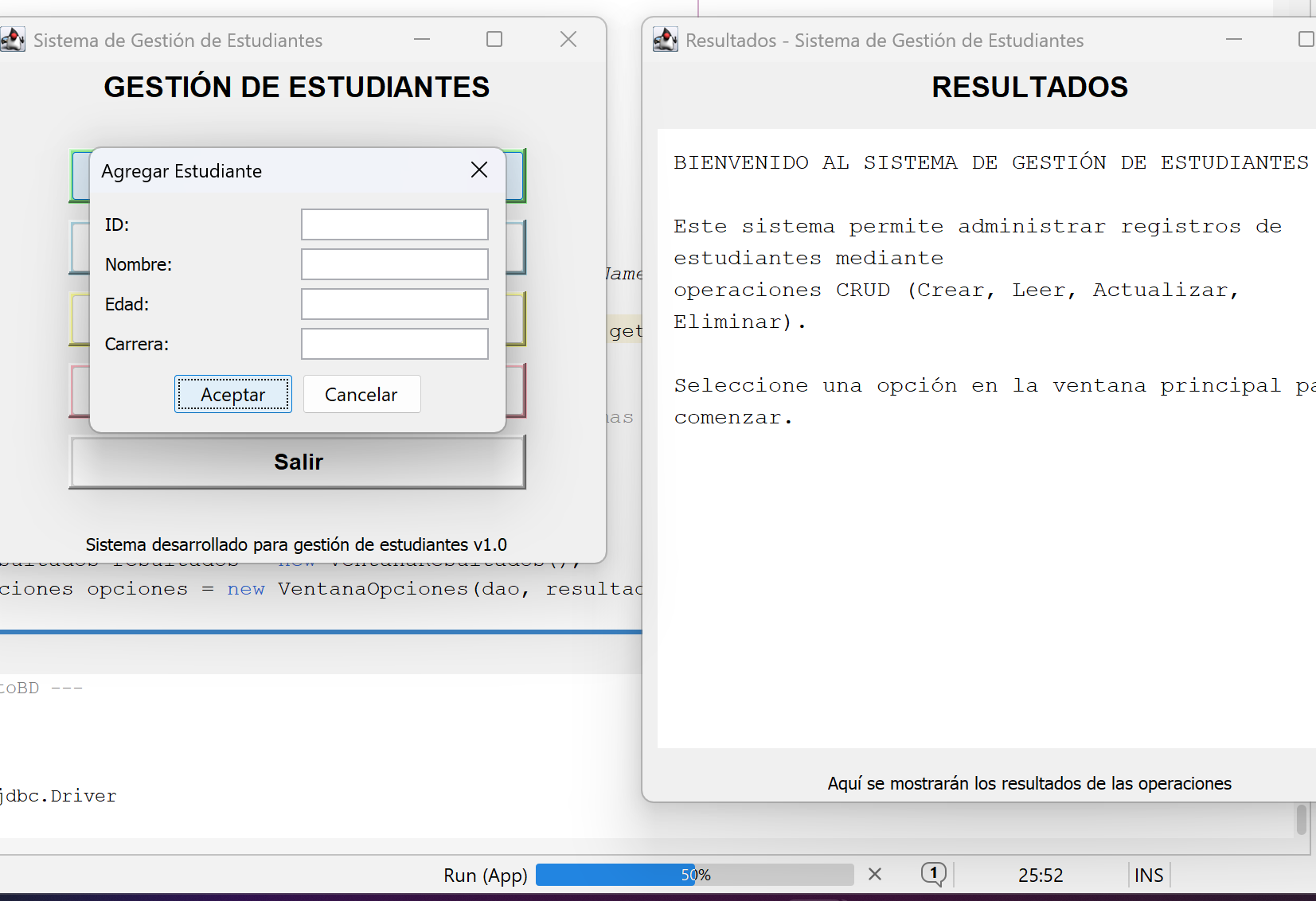
1. **Ejecutar la aplicación**: La aplicación muestra dos ventanas principales
   1. **Ventana de Opciones**: Menú principal con botones de acción
   2. **Ventana de Resultados**: Muestra los resultados de las operaciones
2. **Mensaje de bienvenida**: Al iniciar, se muestra información sobre el sistema

### **Funcionalidades Disponibles**

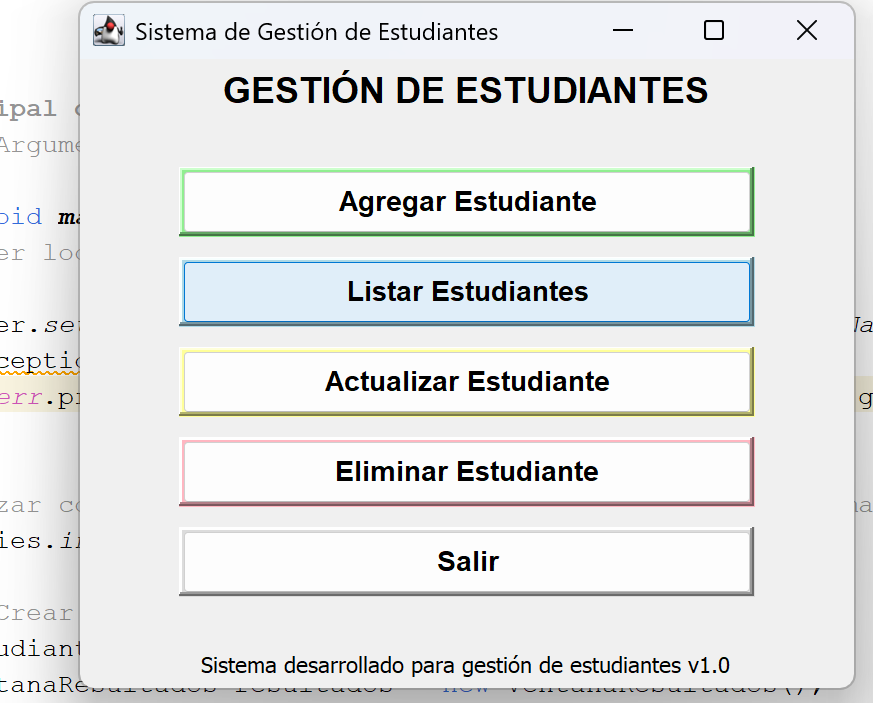


#### **1. Agregar Estudiante**

* **Acción**: Clic en "Agregar Estudiante"
* **Proceso**:
  + Se abre un formulario con campos: ID, Nombre, Edad, Carrera
  + Completar todos los campos obligatorios
  + Hacer clic en "OK" para confirmar
* **Validaciones**:
  + ID debe ser único y numérico
  + Nombre no puede estar vacío
  + Edad debe estar entre 16 y 100 años
  + Carrera no puede estar vacía

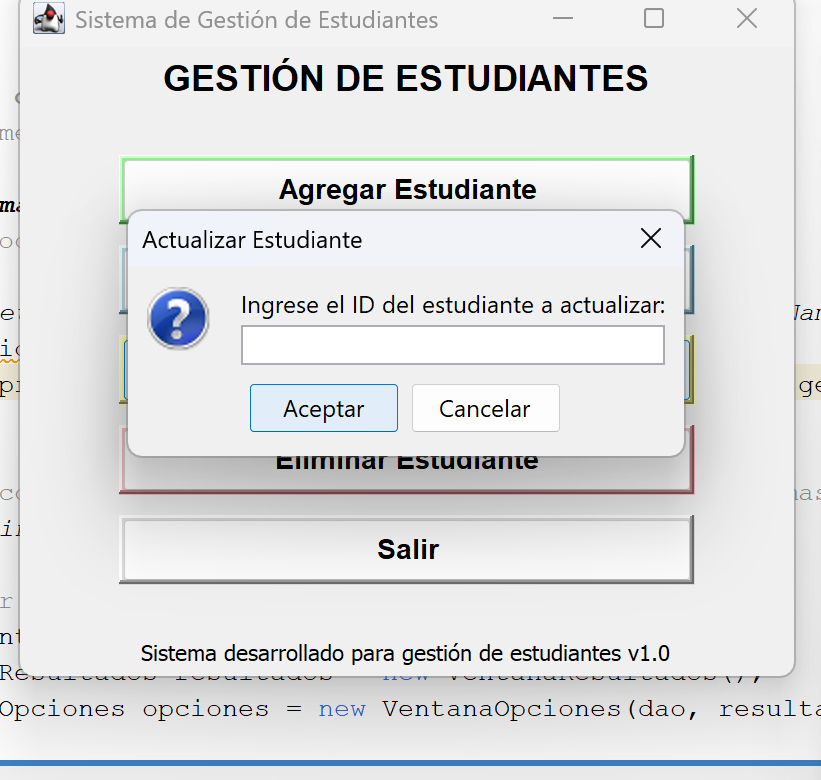


#### **2. Listar Estudiantes**

* **Acción**: Clic en "Listar Estudiantes"
* **Resultado**: Muestra todos los estudiantes registrados con formato: ID: 1Nombre: Juan PérezEdad: 20Carrera: Ingeniería en Sistemas...  
  

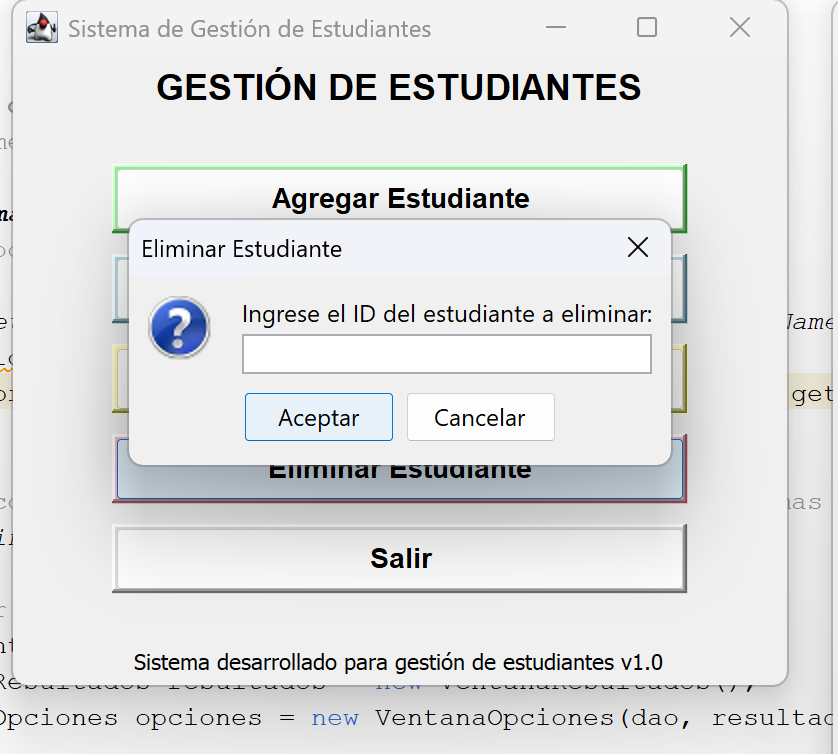
#### **3. Actualizar Estudiante**

* **Acción**: Clic en "Actualizar Estudiante"
* **Proceso**:
  + Ingresar el ID del estudiante a actualizar
  + Si existe, se muestra un formulario con datos actuales
  + Modificar los campos deseados
  + Confirmar cambios
* **Nota**: No se puede cambiar el ID



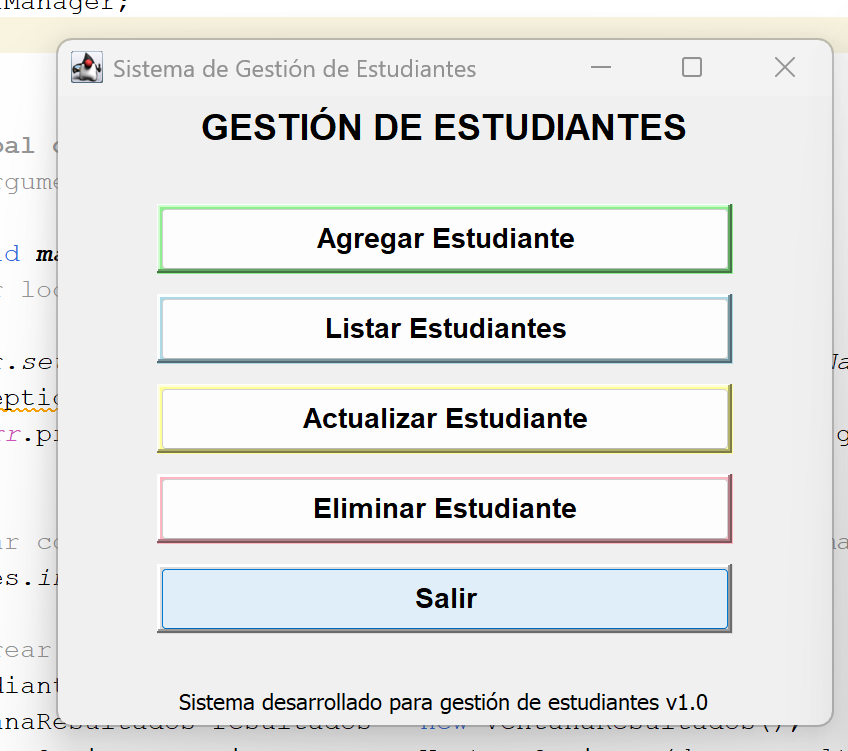
#### **4. Eliminar Estudiante**

* **Acción**: Clic en "Eliminar Estudiante"
* **Proceso**:
  + Ingresar el ID del estudiante a eliminar
  + Confirmar la eliminación en el diálogo de seguridad
  + El registro se elimina permanentemente



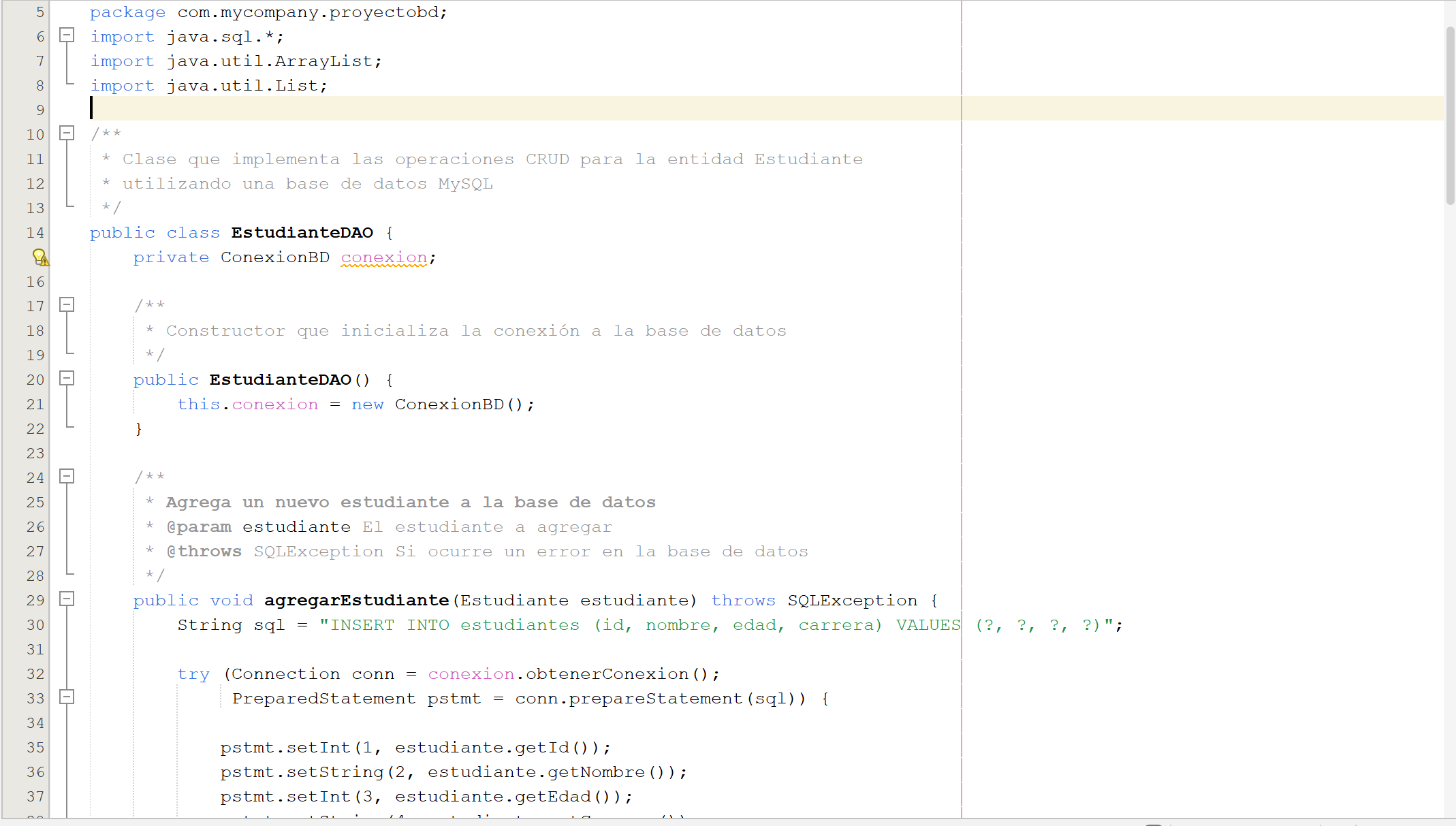
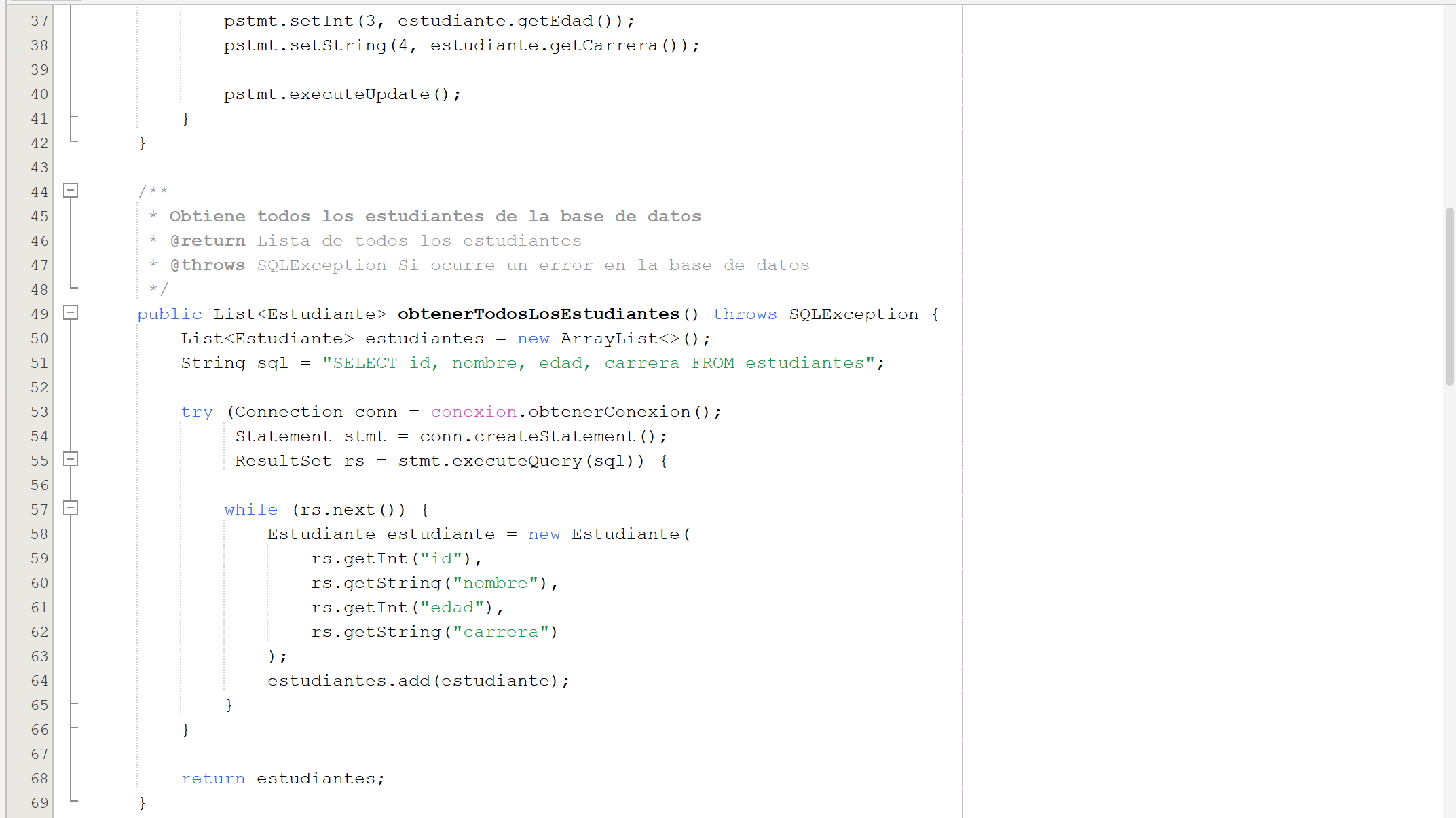
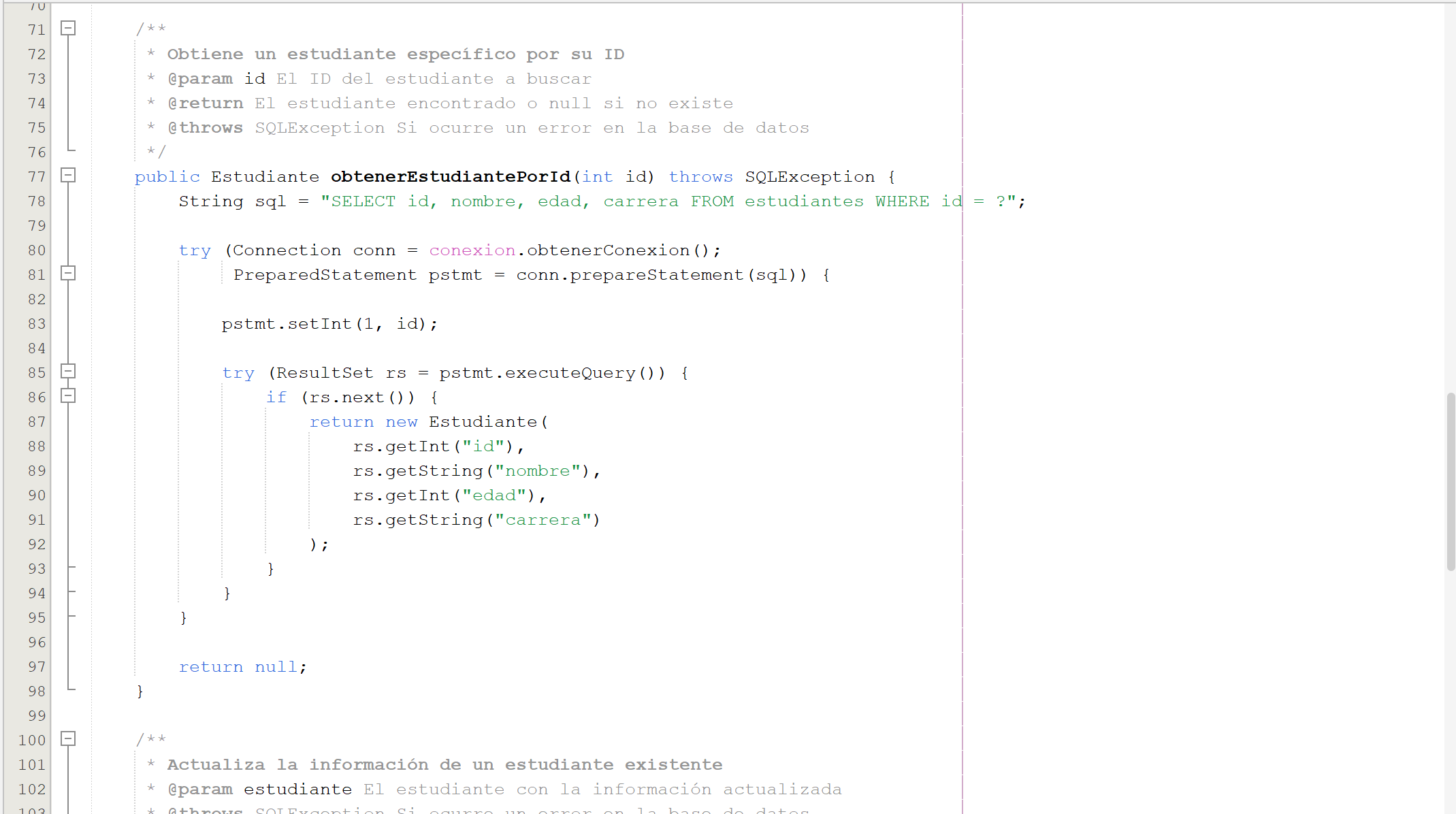
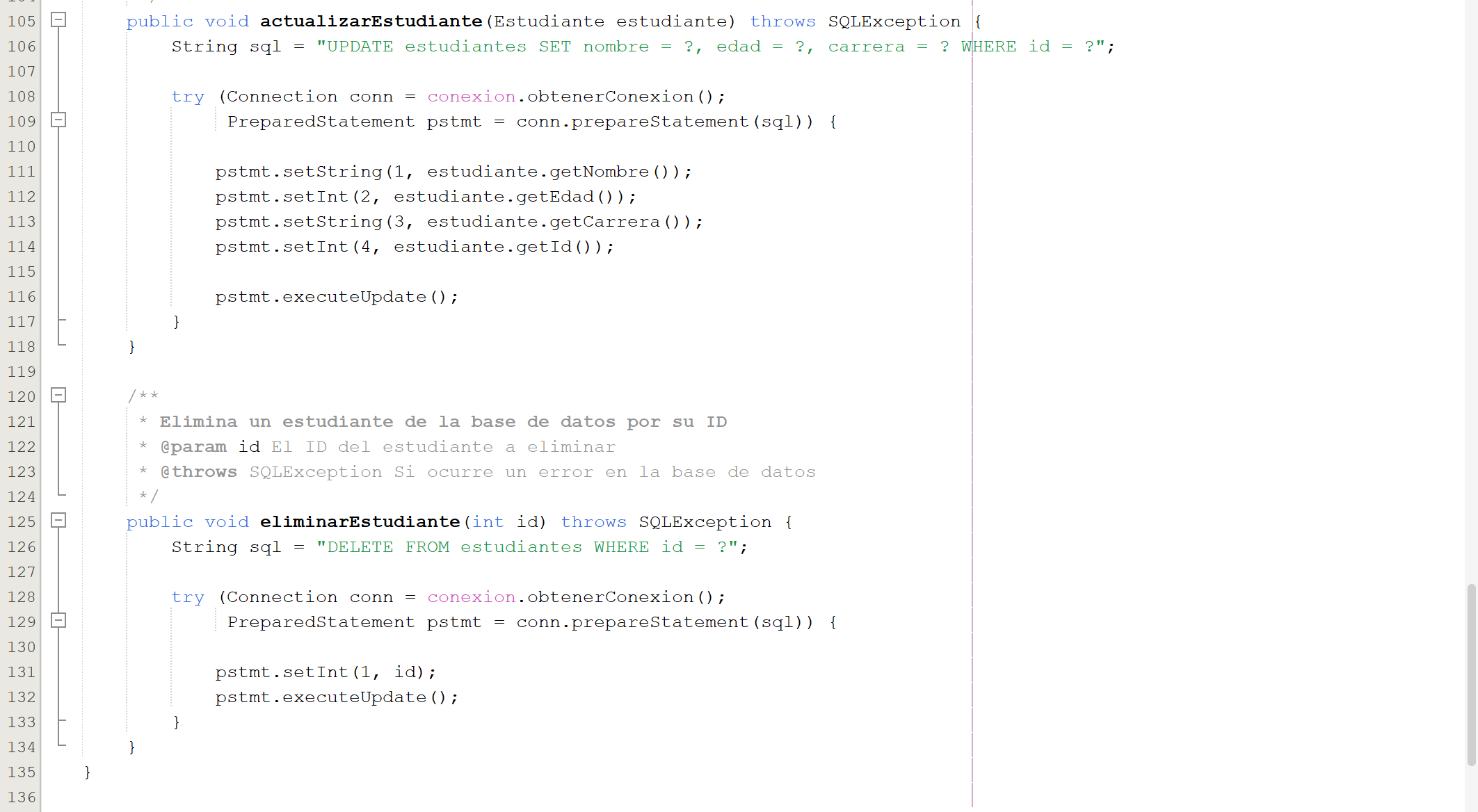
#### **5. Salir**

* **Acción**: Clic en "Salir"
* **Resultado**: Cierra la aplicación completamente



## **Documentación Técnica**

### **Clase Estudiante**

public class Estudiante {  
 private int id; // Identificador único  
 private String nombre; // Nombre completo  
 private int edad; // Edad (16-100)  
 private String carrera; // Programa académico  
}  
    

### **Clase EstudianteDAO**

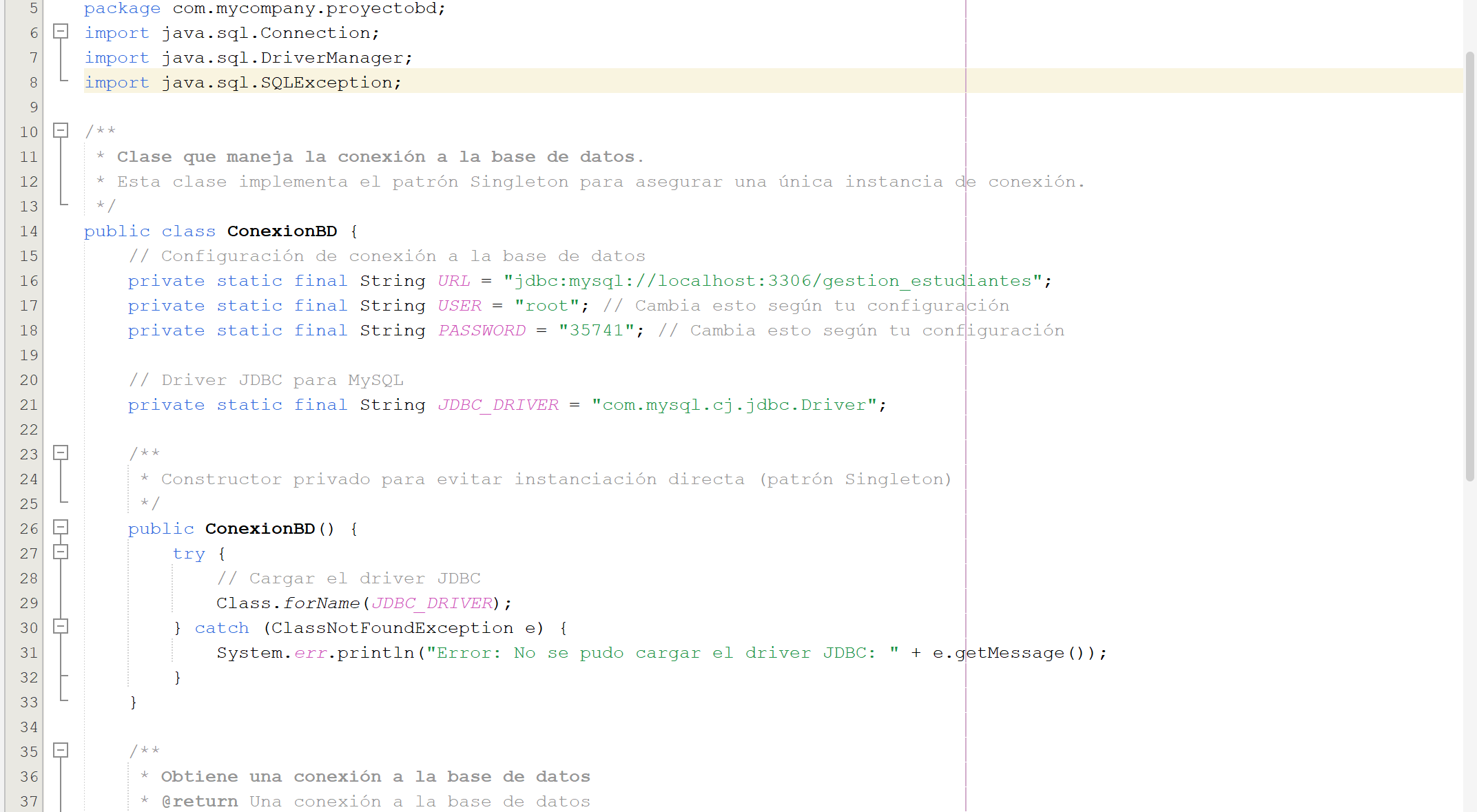
Implementa las operaciones CRUD:

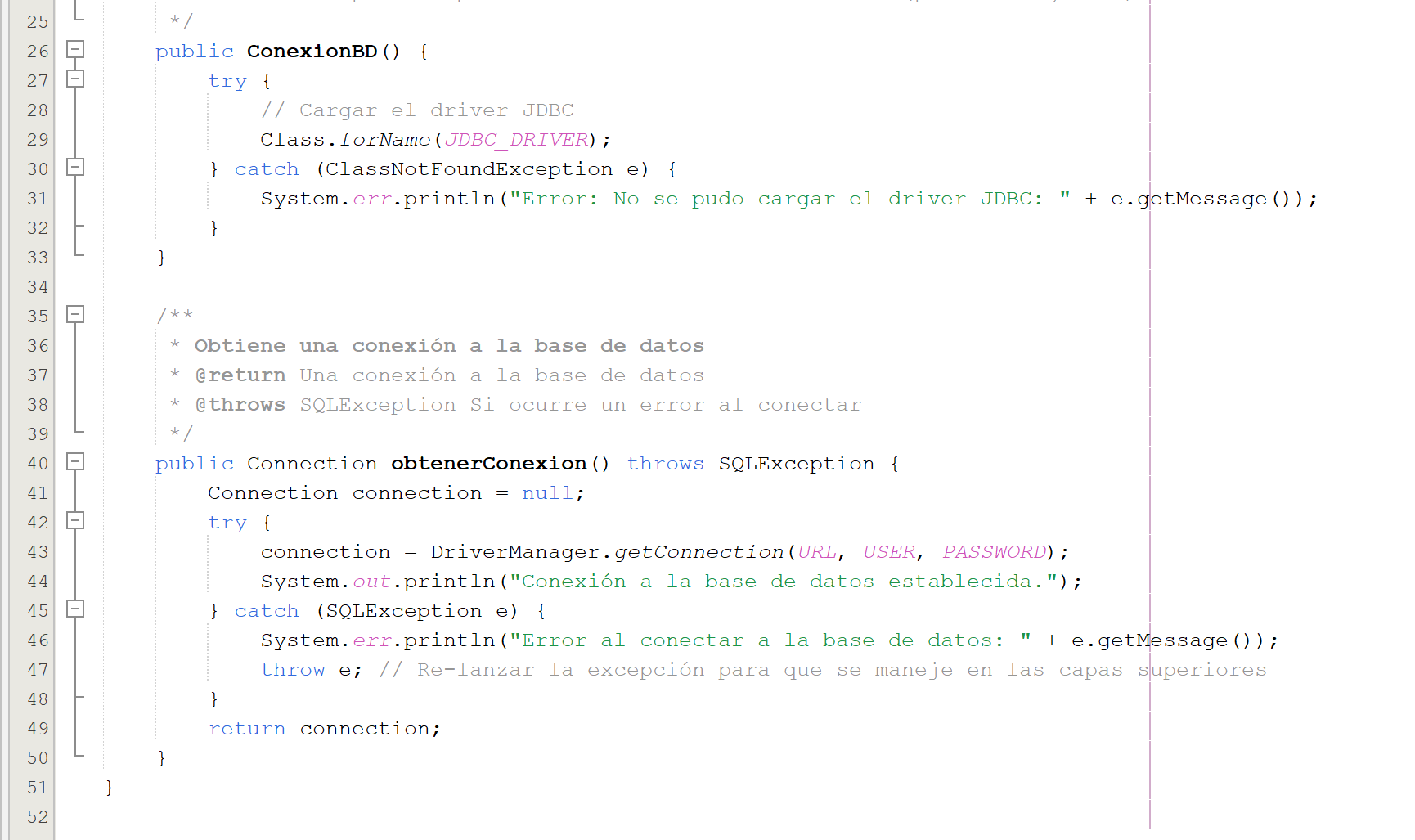
* agregarEstudiante(Estudiante): INSERT
* obtenerTodosLosEstudiantes(): SELECT ALL
* obtenerEstudiantePorId(int): SELECT BY ID
* actualizarEstudiante(Estudiante): UPDATE
* eliminarEstudiante(int): DELETE



### **Clase ConexionBD**

* **Patrón Singleton**: Una sola instancia de conexión
* **Configuración centralizada**: URL, usuario y contraseña
* **Manejo de excepciones**: Captura errores de conexión

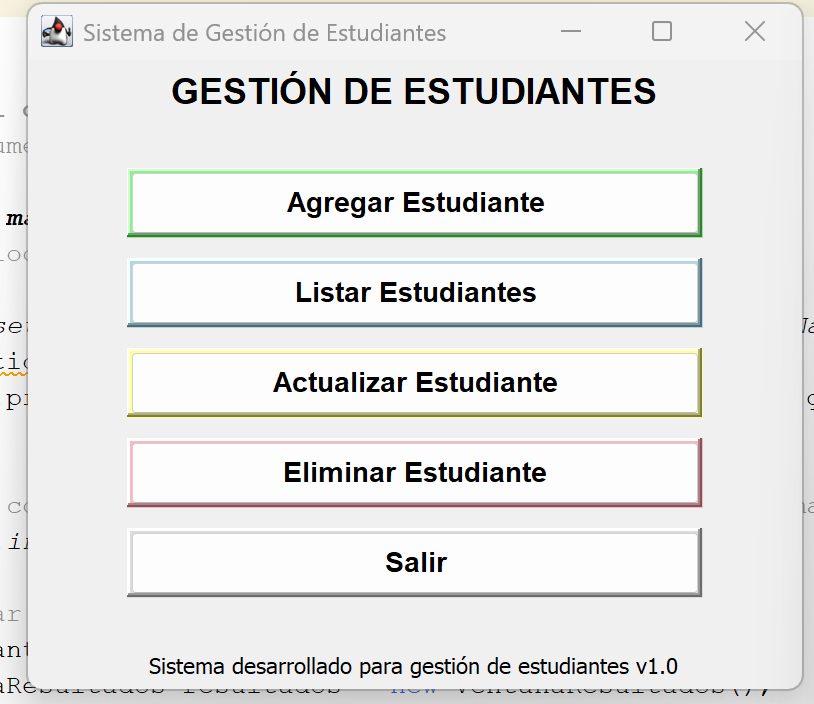




### **Interfaces Gráficas**

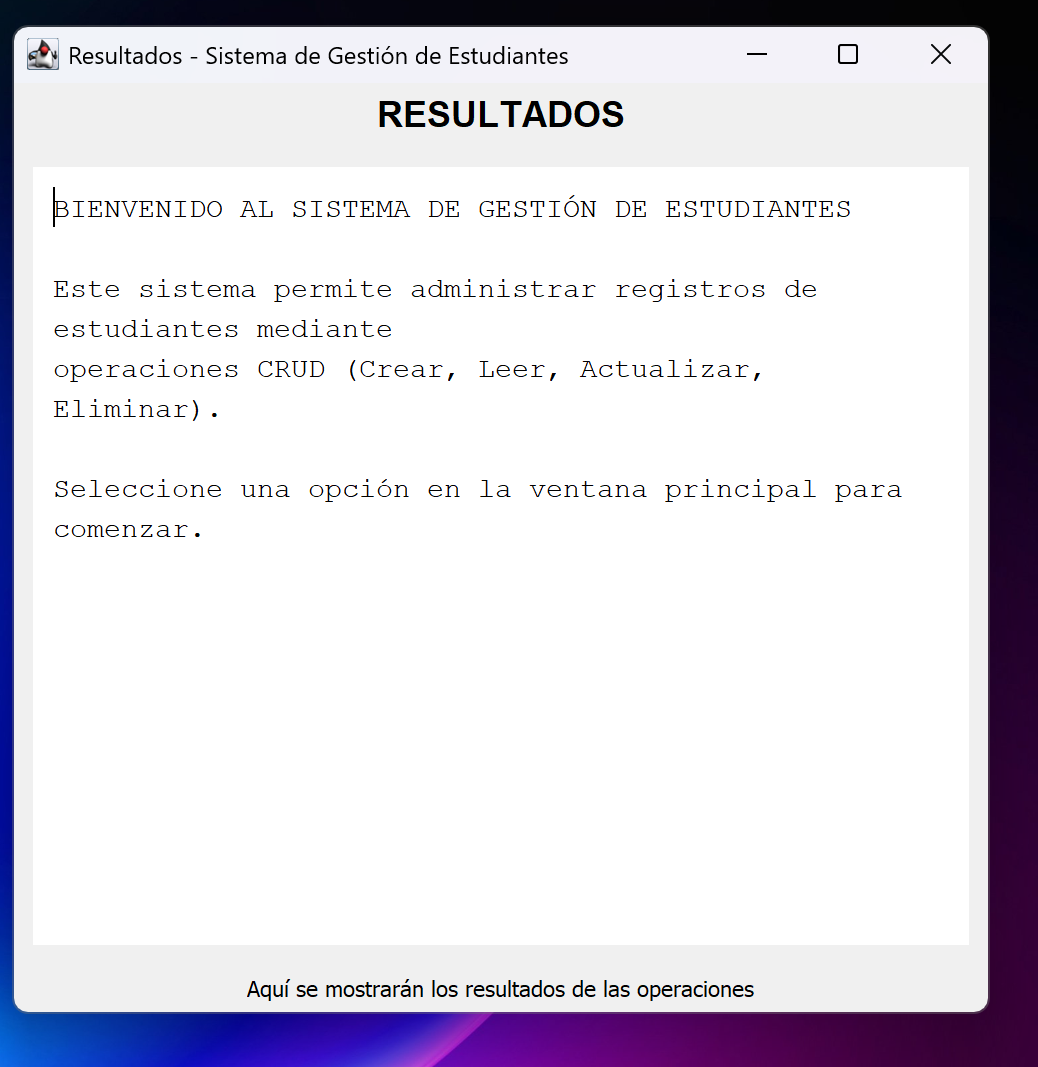
**VentanaOpciones**:

* Layout: BorderLayout con panel de botones central
* Colores personalizados para cada botón
* Validaciones en tiempo real
* Diálogos modales para entrada de datos



**VentanaResultados**:

* Área de texto no editable
* Scroll automático
* Formato monoespaciado para mejor legibilidad



### **Vulnerabilidades Identificadas**

1. **Credenciales hardcodeadas**: Usuario y contraseña en código fuente
2. **Inyección SQL**: Mitigada con PreparedStatements
3. **Conexiones no cifradas**: Conexión local sin SSL

### **Recomendaciones de Seguridad**

1. **Externalizar configuración**: Usar archivos properties
2. **Cifrar contraseñas**: Implementar hash de contraseñas
3. **Validación estricta**: Sanitizar entradas del usuario
4. **Logs de auditoría**: Registrar operaciones críticas

### **Configuración Segura para Producción**

// Ejemplo de configuración externa  
Properties config = new Properties();  
config.load(new FileInputStream("config.properties"));  
String url = config.getProperty("db.url");  
String user = config.getProperty("db.user");  
String password = config.getProperty("db.password");

### **Mantenimiento Preventivo**

* **Respaldos regulares**: Base de datos y código fuente
* **Monitoreo de rendimiento**: Conexiones y consultas
* **Actualizaciones**: Drivers y dependencias

*Esta documentación corresponde al proyecto final de integración de conocimientos en programación, desarrollado como aplicación Stand-Alone en Java con base de datos MySQL.*