Universidad de las Fuerzas Armadas - ESP

Departamento de las Ciencias de Computación

Carrera de Ingeniería de Software

Curso:

NRC 22412

Estudiantes:

Ronny Joel Ibarra Gaona

Carlos Sebastián Rivera Espín

Angelo Patricio Sacnhez Sarabia

Instructor:

Ing. Jenny Ruiz

Fecha:

12-06-2025

Contenido

1. Resumen

En este proyecto se implementó un sistema CRUD para gestionar estudiantes utilizando Java y el patrón de arquitectura en 3 capas: presentación, lógica de negocio y datos. Se utilizó Java Swing para la interfaz gráfica, aplicando principios de encapsulamiento, separación de responsabilidades y reutilización de código. Este sistema permite registrar, buscar, modificar, listar y eliminar estudiantes de forma sencilla y dinámica desde una interfaz visual.

2. Objetivo

Objetivo General:

Desarrollar una aplicación de escritorio en Java que implemente operaciones CRUD de estudiantes aplicando el patrón de arquitectura en 3 capas.

Objetivos Específicos:

- Diseñar una clase modelo que represente la entidad Estudiante.
- Implementar una capa de datos que maneje almacenamiento en memoria.
- Crear una capa de lógica de negocio que gestione las operaciones.
- Desarrollar una interfaz gráfica amigable con Java Swing.

3. Marco Teórico

Arquitectura en 3 Capas

Este patrón divide el software en tres niveles:

- **Capa de presentación:** Interacción con el usuario.
- **Capa de lógica de negocio:** Procesa las reglas del sistema.
- **Capa de datos:** Maneja el almacenamiento o persistencia de datos.

CRUD

CRUD representa las operaciones básicas que puede realizar un sistema de gestión de datos: Crear nuevos registros, Leer o consultar información existente, Actualizar datos ya almacenados y Eliminar registros que ya no se necesiten. Estas acciones son fundamentales en el desarrollo de aplicaciones que manejan bases de datos, ya que permiten mantener la información organizada y actualizada.

Java Swing

- Librería gráfica utilizada para crear interfaces de usuario en Java.

4. Desarrollo

4.1. Capa Modelo: Estudiante.java

Contiene los atributos id, nombre, edad y métodos get/set. Se sobrescribe toString() para representación textual.

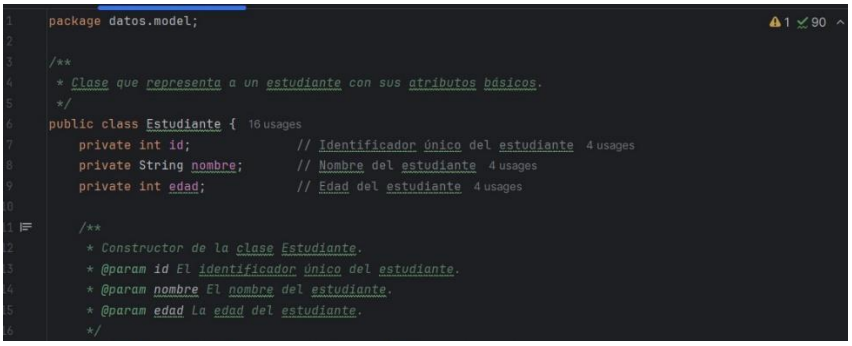

Figure1. datos/Estudiante.java

4.2. Capa Repositorio: EstudianteRepository.java

La Capa Repositorio representada por la clase EstudianteRepository.java se encarga del almacenamiento de los datos en memoria utilizando una lista (List). Esta capa actúa como intermediaria entre la lógica de negocio y la fuente de datos, permitiendo realizar las operaciones básicas CRUD. Entre sus métodos se encuentran: crear() para agregar nuevos estudiantes, leerTodos() para obtener la lista completa, leerPorId() para buscar un estudiante específico por su identificador, actualizar() para modificar datos existentes y eliminar() para remover registros de la lista.

Se encarga del almacenamiento en memoria usando una List. Define métodos:

- crear()
- leerTodos()
- leerPorId()
- actualizar()
- eliminar()

4.3. Capa Lógica de Negocio: EstudianteService.java

La Capa de Lógica de Negocio, implementada en la clase EstudianteService.java, funciona como intermediaria entre la interfaz de usuario y la capa de repositorio. Su principal responsabilidad es encapsular las reglas de validación y coordinar las operaciones CRUD, delegando las tareas específicas al repositorio. Esta capa garantiza que se cumplan las condiciones necesarias antes de acceder o modificar los datos, asegurando así la integridad y coherencia del sistema.

4.4. Capa Presentación: EstudianteUI.java

La Capa de Presentación, implementada en la clase EstudianteUI.java, corresponde a la interfaz gráfica desarrollada con Java Swing. Esta interfaz permite la interacción del usuario mediante campos de entrada (txtId, txtNombre, txtEdad), una tabla (JTable) para mostrar los estudiantes y botones para ejecutar las operaciones de Agregar, Modificar, Eliminar, Buscar y Listar. Además, se incorporaron validaciones básicas de tipo de datos utilizando bloques try-catch para evitar errores durante la ejecución.

Se construyó una interfaz gráfica con Java Swing. Incluye:

- Campos de entrada (txtId, txtNombre, txtEdad)
- Tabla con estudiantes (JTable)
- Botones para Agregar, Modificar, Eliminar, Buscar, Listar
- Validaciones básicas de tipo de datos con try-catch

5. Resultados

El sistema permite gestionar estudiantes de manera eficiente, reflejando los cambios inmediatamente en la tabla. Las acciones están correctamente encapsuladas en cada capa, lo que facilita el mantenimiento y la escalabilidad del software.

6. Conclusiones

La adopción de la arquitectura en tres capas fue fundamental para la organización clara y estructurada del proyecto. Separar la lógica del negocio, la presentación y el acceso a datos permitió distribuir adecuadamente las responsabilidades de cada componente, facilitando el desarrollo, la comprensión del flujo del sistema y la posibilidad de realizar modificaciones sin afectar otras partes del código. Cada capa fue diseñada con una

finalidad específica: la capa modelo definió la entidad Estudiante, la capa repositorio gestionó el almacenamiento en memoria, la capa de lógica de negocio coordinó las operaciones y validaciones, y la capa de presentación permitió la interacción del usuario a través de una interfaz amigable construida con Java Swing.

7. Recomendaciones

- Agregar validaciones más robustas (por ejemplo, evitar IDs duplicados).
- Implementar persistencia con base de datos (usando JDBC o Hibernate).
- Utilizar patrones como DAO para mayor escalabilidad.
- Modularizar el UI en componentes reutilizables.

8. Anexos

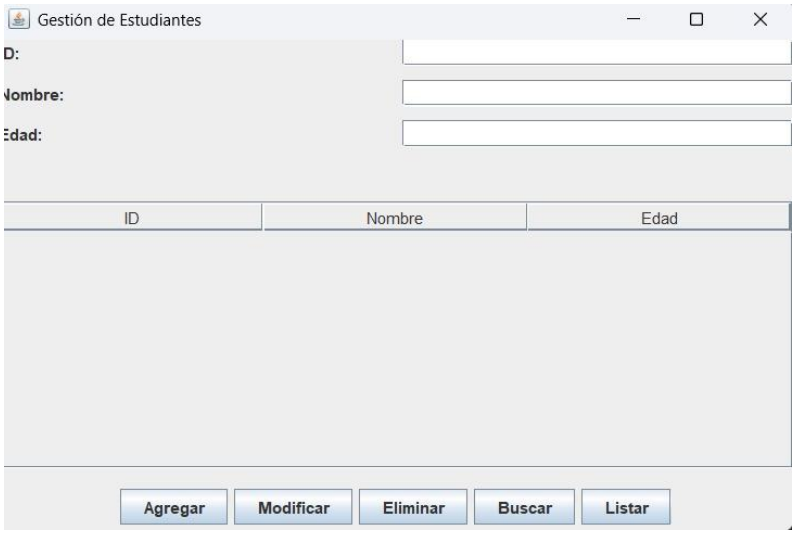

Figura3. Ejecución CRUD