Informe Proyecto App 2

Lenguaje y Paradigmas de la Programación (TICS 200)

Juan Pablo Salum, Julian Epple, Lucas Colombo Echenique, Matilda Vasquez, Alvaro Astudillo

1. Arquitectura del Sistema

1.1. Estructura General

El proyecto está desarrollado en Java y sigue una arquitectura modular basada en el Paradigma Orientado a Objetos (POO). Está organizado en los siguientes paquetes:

- models: contiene las clases de dominio como Cultivo, Parcela, Actividad y la clase abstracta ElementoAgricola.
- services: agrupa la lógica de negocio, como los servicios para CultivoService, ParcelaService y ActividadService.
- io: contiene CSVHandler, responsable de la lectura y escritura del archivo cultivos.csv.
- ui: contiene los menús y la lógica de interfaz por consola (como MenuCultivos, MenuParcelas, etc.).
- Archivo principal: App2.java, que implementa el menú principal y coordina el flujo del programa.

1.3. Modificadores de Acceso

Se utilizaron modificadores private para encapsular atributos, y public para métodos de acceso y constructores. Las listas se manejan con ArrayList y se evita el acceso directo a los atributos.

2. Justificación de Diseño

2.1. Uso de Colecciones

Se usó ArrayList para almacenar listas de cultivos, actividades y parcelas, por su eficiencia, facilidad de uso y dinamismo.

2.2. Patrones de Diseño (implícitos)

- DAO simple: CSVHandler actúa como un Data Access Object para la persistencia de Cultivo.
- Separation of Concerns: separamos claramente modelos, lógica de negocio, persistencia y presentación.

3. Reflexiones Finales

¿Qué fue lo más desafiante?

Implementar una arquitectura limpia en POO y lograr que los objetos se comuniquen de forma coherente sin romper la separación de responsabilidades. También fue desafiante el manejo correcto de fechas, estructuras anidadas y modularidad.

¿Cómo se controló la lectura y escritura del CSV?

Se creó la clase CSVHandler con dos métodos estáticos: leerCultivosDesdeCSV() y escribirCultivosEnCSV(), que utilizan BufferedReader y BufferedWriter. Se definen los headers y se asegura que los datos se lean/salven en el formato correcto.

¿Qué aprendizajes surgieron?

- Aplicar POO con propósito real
- Usar colecciones, herencia, encapsulación y estructuras de menú
- Trabajar en equipo usando Git y ramas

4. Uso de Inteligencia Artificial

En este proyecto, se utilizó herramientas de IA (Copilot, Chat-GPT, Gemini) para obtener sugerencias sobre la estructura del código y la implementación de ciertas funcionalidades.

Se utilizó una IA para:

- Explicar conceptos de Java
- Dar ejemplos de código para el manejo de archivos y estructuras POO
- Sugerir estructuras de menú y organización de clases
- Implementación de funciones.

¿Cómo se validó?

- Cada sugerencia fue contrastada:
- Compilando localmente y probando en consola
- Ajustando el código a las necesidades específicas del enunciado
- Discutiendo en equipo antes de incorporar

5. Compilación y Ejecución

Requisitos

- Java instalado (Java 16+)
- Archivo cultivos.csv en la raíz del proyecto

Instrucciones:

- Clonar el repositorio: git clone https://github.com/matildavasquezdevi/App2.git
- Compilar (desde la raíz del proyecto): “javac -cp src -d bin src/models/*.java src/services/*.java src/io/*.java App2.java”
- Ejecutar: “java -cp bin App2 cultivos.csv”
- Al ejecutar, se mostrará un menú por consola para:
    - Listar, crear, editar y eliminar cultivos
    - Registrar actividades
    - Buscar cultivos y generar reportes
    - Salvar los cambios al salir