UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
INVESTIGACION EN INGENIERIA DE SOFTWARE

Nombre: Danna Valentina Andrade Lucio.
NRC: 28482
Fecha: 03/02/2026.
Carrera: Ingeniería en Software.

INTRODUCCIÓN

Este documento recopila la interacción con el Asistente de IA (Gemini) utilizada para el análisis, desarrollo y corrección del ejercicio de replicación C1. Se detallan los prompts utilizados para estructurar las pruebas y solucionar errores de entorno.

INTERACCIÓN 1: Análisis del Planteamiento y Estrategia

Propósito: Comprender los requisitos del ejercicio y definir la estrategia de pruebas desde cero.

PROMPT DEL ESTUDIANTE:

Se va a realizar la siguiente actividad con la instrucuion (Implemente el código indicado en la especificación adjunta. Para ello, puede utilizar el LLM indicado por el profesor. Utilice el interfaz web, ya que deberá aportar una copia de la conversación que ha mantenido con el LLM.), para la cual se necesita cumplir con lo que se especifica dentro del archivo adjunto(Se adjunto el documento de especificaciones junto con la imagen del diagrama de clases)

RESPUESTA GENERADA POR LA IA:

He implementado la clase File completa según las especificaciones:

1. Excepciones Creadas:

- InvalidContentException: Se lanza cuando el contenido es null
- WrongFileTypeException: Se lanza cuando se intenta agregar propiedades a un archivo IMAGE

2. Constructor:

java

```
public File() {
    this.content = new ArrayList<>();
}
```

El constructor inicializa content como un ArrayList vacío, cumpliendo el requisito de que no sea null.

3. Método addProperty():

java

```
public void addProperty(char[] newcontent) throws InvalidContentException,
WrongFileTypeException {
    // Verificar si newcontent es null
    if (newcontent == null) {
        throw new InvalidContentException("El contenido no puede ser null");
    }

    // Verificar si el tipo de archivo es IMAGE
    if (this.type == FileType.IMAGE) {
        throw new WrongFileTypeException("No se puede agregar propiedades a un archivo de
tipo IMAGE");
    }

    // Añadir el newcontent al content existente
    for (char c : newcontent) {
        this.content.add(c);
    }
}
```

Este método:
- Valida que newcontent no sea null
- Verifica que el tipo de archivo sea PROPERTY (no IMAGE)
- Añade los caracteres al content existente

4. Método getCRC32():

```
java
public long getCRC32() {
    // Si el content está vacío, devolver 0
    if (this.content.isEmpty()) {
        return 0L;
    }

    // Convertir ArrayList<Character> a byte[]
    byte[] bytes = new byte[this.content.size()];
    for (int i = 0; i < this.content.size(); i++) {
        // Obtener el byte menos significativo
        bytes[i] = (byte) (this.content.get(i) & 0x00FF);
    }

    // Calcular y devolver el CRC32
    FileUtils fileUtils = new FileUtils();
    return fileUtils.calculateCRC32(bytes);
}
```

Este método:
- Devuelve 0 si content está vacío
- Convierte el ArrayList<Character> a byte[] usando el byte menos significativo
- Utiliza FileUtils.calculateCRC32() para calcular el CRC32

Consideraciones Técnicas:
1. Manejo de UTF-8: Se usa el enmascaramiento con 0x00FF para obtener el byte menos significativo de cada carácter
2. Manejo de excepciones: Se definieron dos excepciones específicas para casos de error
3. Arquitectura limpia: El código sigue los principios SOLID y es fácilmente testeable

Archivos Modificados:
- `File.java` - Implementación completa de la clase
- `InvalidContentException.java` - Nueva excepción creada
- `WrongFileTypeException.java` - Nueva excepción creada