DOCUMENTACIÓN TÉCNICA

PRÁCTICA - Tema 02

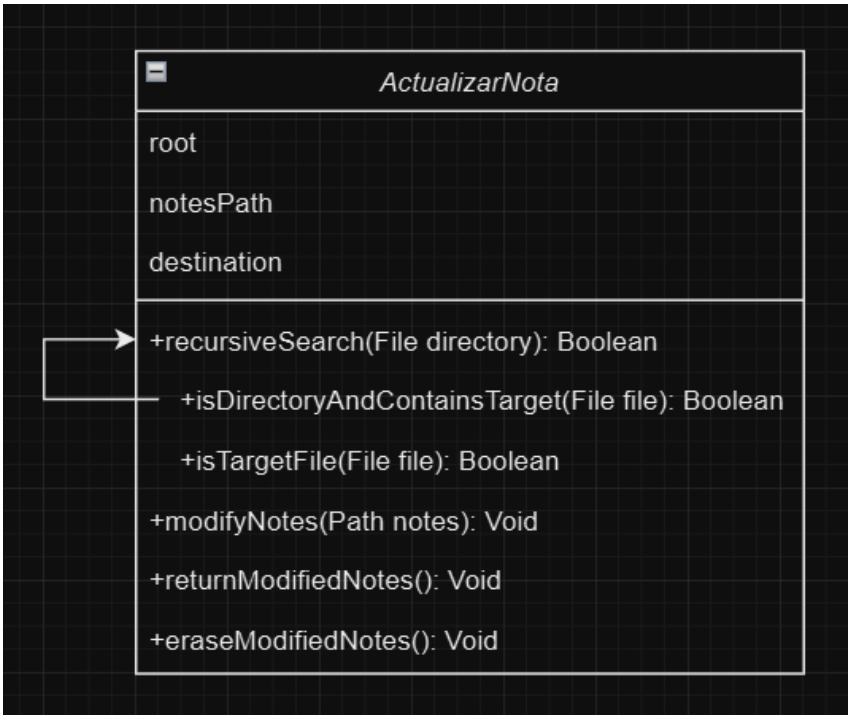
PROYECTO:

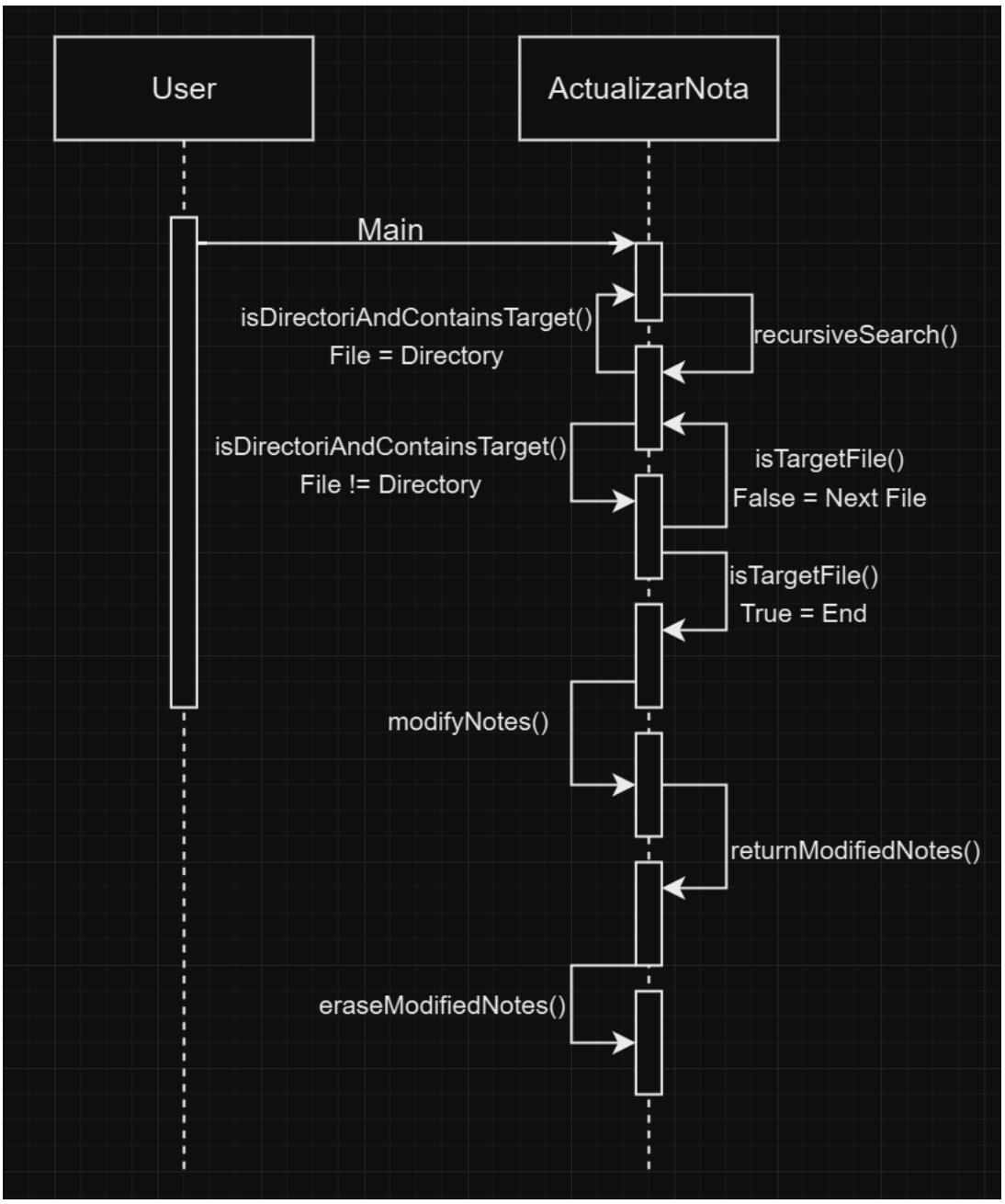
"ActualizarNota"

1. Descripció del sistema

Localizar el archivo NotasDAM2.txt y modificar la nota de un alumno en concreto.

2. Diagrama de classes i seqüències





3. Detall del codi

Se ha intentado mantener una práctica en la cual se intenta anidar un máximo de 3 veces, por eso se ha separado la búsqueda recursiva en un total de 3 métodos y a su vez, que sea más fácil la lectura.

a. Descripció de les funcions de cada mètode, paràmetres, i valors de retorn

i. main(String[] args)

1. Se encarga de ejecutar los 4 métodos que llevan a cabo la funcionalidad del programa.

ii. **recursiveSearch(File directory)**
    1. Crea un array con .listFiles(), primero confirma que el array creado tenga algo de contenido (no sea null), y una vez que tenemos datos, empieza a iterar sobre cada File en el método siguiente isDirectoryAndContainsTarget(File file) el cual si devuelve true en el caso de que haya encontrado el archivo.

iii. **isDirectoryAndContainsTarget(File file)**
    1. Si el file que se analiza es una carpeta con .isDirectory() se hace una llamada al método recursiveSearch(File file) para seguir buscando de manera recursiva. En el caso de que NO sea un directorio, se pasa al método isTargetFile(File file).

iv. **isTargetFile(File file)**
    1. Con un .getName().equals() comprueba el nombre del archivo, return false si no coincide, si coincide realiza la copia del archivo en un destino cualquiera (puesto que luego se borrara), obtenemos la ruta del archivo y devuelve true.

v. **modifyNotes(Path notes, String studentName, String studentNote)**
    1. Crea 2 List, una se llena con .readAllLines() y la otra se queda vacía, se llenará con las líneas una vez estén modificadas. Iteramos sobre las líneas en busca del nombre del alumno, cuando lo tengamos separamos la línea en un array de Strings, y con un .lenght-1 localizamos siempre la ubicación de la nota, se modifica, volvemos a construir la línea con un .join, añadimos la línea a la lista de líneas modificadas y finalmente lo escribimos con un .write.

vi. **returnModifiedNotes()**
    1. Simplemente usamos el método .copy(origen, destino, opciones de copia) de la clase Files.

vii. **eraseModifiedNotes()**
    1. Con un Files.delete() eliminamos la copia que hemos creado modificada, ya que ha sido reemplazada por la original y no la necesitamos.

b. Comportament esperat del codi.

Que no devuelva nada por pantalla ni mensaje, simplemente modifique el archivo y borre la copia generada para su modificación.

c. Possibles excepcions que es gestionen

Se tiene que ejecutar como administrador debido a las rutas y posibles errores de seguridad de Windows. Es posible que tampoco pueda acceder a carpetas dentro de la carpeta del usuario.

4. Entorn de desenvolupament

Todo hecho en Java, en el IDE Intellij, como proyecto Intellij **NO** como Maven.
Librerias:

```
import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardCopyOption;
import java.util.ArrayList;
import java.util.List;
```

5. Tests

Se han preparado un total de 3 tests, verificar que el archivo que se encuentra es el correcto **testIsTargetFile()**, Confirmar que la nota se ha modificado correctamente **testModifyNotes()** y modificar la nota de otro alumno **testModifyNotesForDifferentStudent().** Antes de cada test se crea una carpeta temporal y un archivo que serán para dichas pruebas con un @BeforeEach. Se usa carpeta temporal y no solo el archivo porque si creamos solo el archivo temporal windows le asigna un número aleatorio para validar que es único. De esta manera el número aleatorio se le asigna al directorio y no al .txt

- **testIsTargetFile()**
  - Se comprueba el archivo mediante un assertTrue()
- **testModifyNotes()**
  - Se modifican las notas y comprueban mediante un assertTrue()
- **testModifyNotesForDifferentStudent()**
  - Se modifican las notas con el nombre de otro alumno y otra nota, y a su vez que otros alumnos no han sido modificados mediante 2 assertTrue().

6. Información adicional sobre el proyecto

El apartado de control de versiones con GitHub no se ha realizado debido a la brevedad del proyecto, puesto que ha llevado más tiempo su correspondiente documentación que la programación del código en sí.

También agradecería el feedback del trabajo sobre los tiempos verbales, es decir, es posible encontrar un “pupurri” a la hora de identificarme. Frases como si de un equipo se tratase “**creamos** el método x”, también creo haberlas escrito como “**creo** el método x”, o en neutro como “**se crea** el método X”. Me gustaría saber para este tipo de documentaciones que forma verbal sería la adecuada, en equipo, en solitario o en impersonal o ninguna de estas claro está.