Internship Week (1) Day 2 Task

Step 1: Setting Up IntelliJ IDEA

1. **Open IntelliJ IDEA**.

2. **Create a New Project**:

   o Click on File > New > Project.

   o Select Maven on the left side.

   o Check Create from archetype if you need a predefined structure.

3. **Advanced settings Details**:

   o GroupId: com.tripillar.filehandling

   o ArtifactId: FileHandlingProject

   o Version: Leave default (or 1.0-SNAPSHOT).

   o Click Finish.

Step 2: Adding Apache POI Dependency for Excel Handling

1. **Open the pom.xml file** from the project root.

2. Inside <dependencies>, add the following dependencies for Apache POI (for Excel handling):

Pom.xml

```
<dependencies>

  <!-- Apache POI dependencies -->

  <dependency>

    <groupId>org.apache.poi</groupId>

    <artifactId>poi</artifactId>

    <version>5.2.3</version>

  </dependency>

  <dependency>

    <groupId>org.apache.poi</groupId>

    <artifactId>poi-ooxml</artifactId>

    <version>5.2.3</version>

  </dependency>

  <!-- For parsing XSSF files (Excel 2007+) -->
```

```xml
<dependency>

    <groupId>org.apache.xmlbeans</groupId>

    <artifactId>xmlbeans</artifactId>

    <version>5.1.1</version>

  </dependency>

</dependencies>
```

3. **Reload Maven** to download the dependencies: You can press the Reload button in the Maven tool window.

Step 3: Creating Package Structure

1. In the src/main/java folder, right-click and create two packages:

   o com.tripillar.filehandling.text

   o com.tripillar.filehandling.excel

Step 4: Writing Code for Text File Handling

a. WriteTextFile.java

1. Inside the text package, create a new Java class WriteTextFile.java.

2. Write code to create and write to a text file using BufferedWriter and FileWriter:

java

```java
package com.tripillar.filehandling.text;


import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;


public class WriteTextFile {

  public static void main(String[] args) {

    String fileName = "example.txt";

    try (BufferedWriter writer = new BufferedWriter(new FileWriter(fileName))) {

      writer.write("Hello, this is a sample text.");

      System.out.println("Text file written successfully.");
```

```java
} catch (IOException e) {

        e.printStackTrace();

    }

  }

}
```

b. ReadTextFile.java

1. Create another Java class ReadTextFile.java.

2. Write code to read from a text file using BufferedReader and FileReader:

java

```java
package com.tripillar.filehandling.text;


import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;


public class ReadTextFile {
  public static void main(String[] args) {
    String fileName = "example.txt";
    try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {
      String line;
      while ((line = reader.readLine()) != null) {
        System.out.println(line);
      }
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

Step 5: Writing Code for Excel File Handling

a. WriteExcelFile.java

1. Inside the excel package, create a Java class WriteExcelFile.java.

2. Write code to create an Excel file using Apache POI:

java

```java
package com.tripillar.filehandling.excel;


import org.apache.poi.ss.usermodel.*;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;


import java.io.FileOutputStream;

import java.io.IOException;


public class WriteExcelFile {

  public static void main(String[] args) {

    String fileName = "example.xlsx";

    Workbook workbook = new XSSFWorkbook();

    Sheet sheet = workbook.createSheet("SampleSheet");


    // Creating a row and adding data

    Row row = sheet.createRow(0);

    row.createCell(0).setCellValue("Name");

    row.createCell(1).setCellValue("Age");


    Row row1 = sheet.createRow(1);

    row1.createCell(0).setCellValue("John Doe");

    row1.createCell(1).setCellValue(25);
```

```java
    // Writing to file

    try (FileOutputStream fileOut = new FileOutputStream(fileName)) {

        workbook.write(fileOut);

        System.out.println("Excel file written successfully.");

    } catch (IOException e) {

        e.printStackTrace();

    }

  }

}
```

b. ReadExcelFile.java

1. Create another Java class ReadExcelFile.java in the excel package.

2. Write code to read from an Excel file:

java

```java
package com.tripillar.filehandling.excel;


import org.apache.poi.ss.usermodel.*;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;


import java.io.FileInputStream;

import java.io.IOException;


public class ReadExcelFile {

  public static void main(String[] args) {

    String fileName = "example.xlsx";

    try (FileInputStream fis = new FileInputStream(fileName);

        Workbook workbook = new XSSFWorkbook(fis)) {


      Sheet sheet = workbook.getSheetAt(0);
```

```
 for (Row row : sheet) {

         for (Cell cell : row) {

            switch (cell.getCellType()) {

               case STRING:

                  System.out.print(cell.getStringCellValue() + "\t");

                  break;

               case NUMERIC:

                  System.out.print(cell.getNumericCellValue() + "\t");

                  break;

            }

         }

         System.out.println();

      }

   } catch (IOException e) {

      e.printStackTrace();

   }

  }

}
```

Step 6: Running the Project

1. Right-click on each file and choose Run.

2. Ensure that both the text file and Excel file operations work as expected.

Step 7: GitHub Submission

1. **Initialize Git in your project directory**:

git init

git add .

git commit -m "Initial commit - File Handling Project"

2. **Create a new repository on GitHub**.

3. **Push your project to GitHub**:

git remote add origin <your-repo-url>

git push -u origin main

Challenges faced :

1. While adding maven dependencies I got some error then I resolved that problems.
2. While coding and giving naming conventions we faced some problem then we resolved it.