**Exercise 1: Implementing the Singleton Pattern**

**Scenario:**

You need to ensure that a logging utility class in your application has only one instance throughout the application lifecycle to ensure consistent logging.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **SingletonPatternExample**.
2. **Define a Singleton Class:**
   * Create a class named Logger that has a private static instance of itself.
   * Ensure the constructor of Logger is private.
   * Provide a public static method to get the instance of the Logger class.
3. **Implement the Singleton Pattern:**
   * Write code to ensure that the Logger class follows the Singleton design pattern.
4. **Test the Singleton Implementation:**
   * Create a test class to verify that only one instance of Logger is created and used across the application.

**PROCEDURE :**

1. **Create a New Java Project :**

• Create Java Project and select it.

• Choose No Build Tools when prompted.

• Select a location on your file system to create the project.

• Name the project SingletonPatternExample.

1. **Open the Project :** Once the project is created, it will open in a new window.
2. **Define the Singleton Class :**

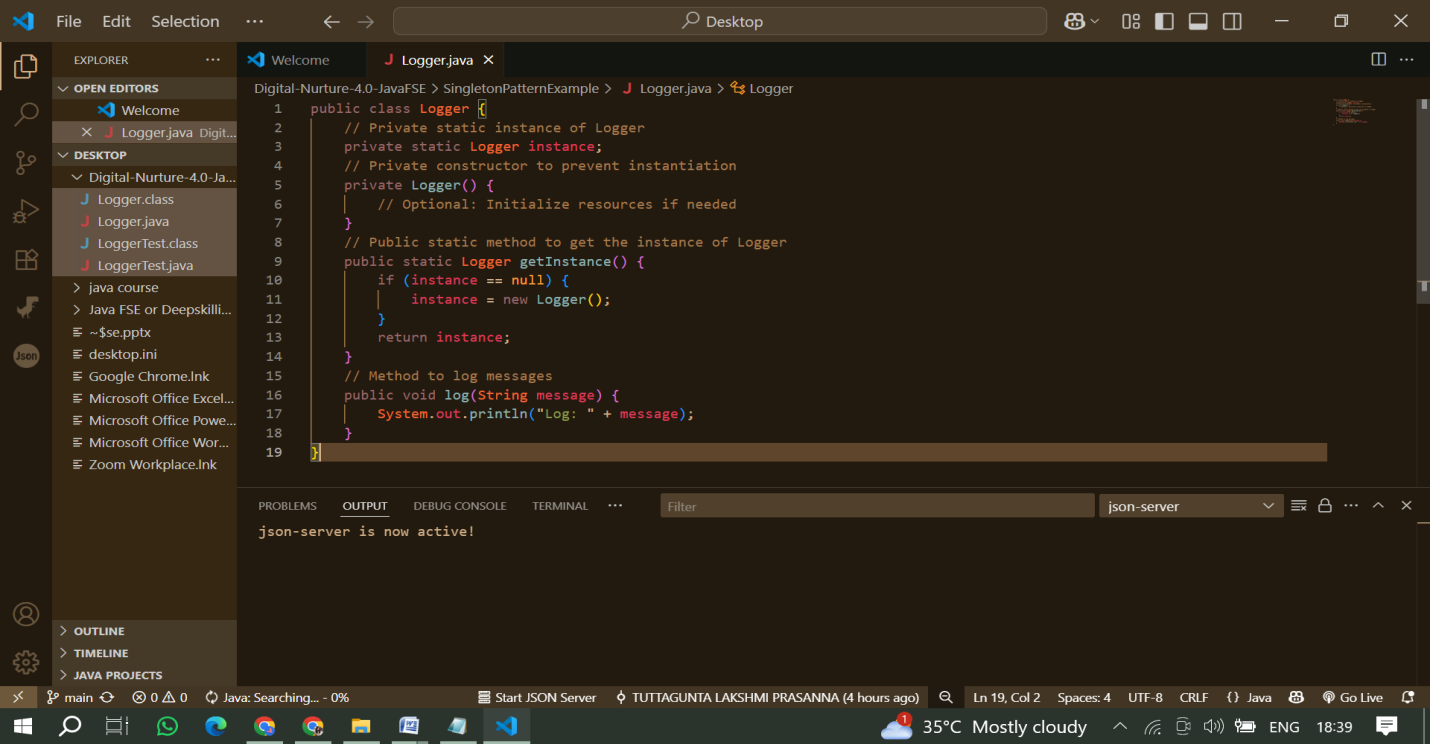
1. Create a New Java Class:

• In the Explorer view, right-click on the source folder and select New File.

• Name the file Logger.java.

2. Implement the Logger Class:

• Open Logger.java and add the following code:



1. **Create the Test Class :**

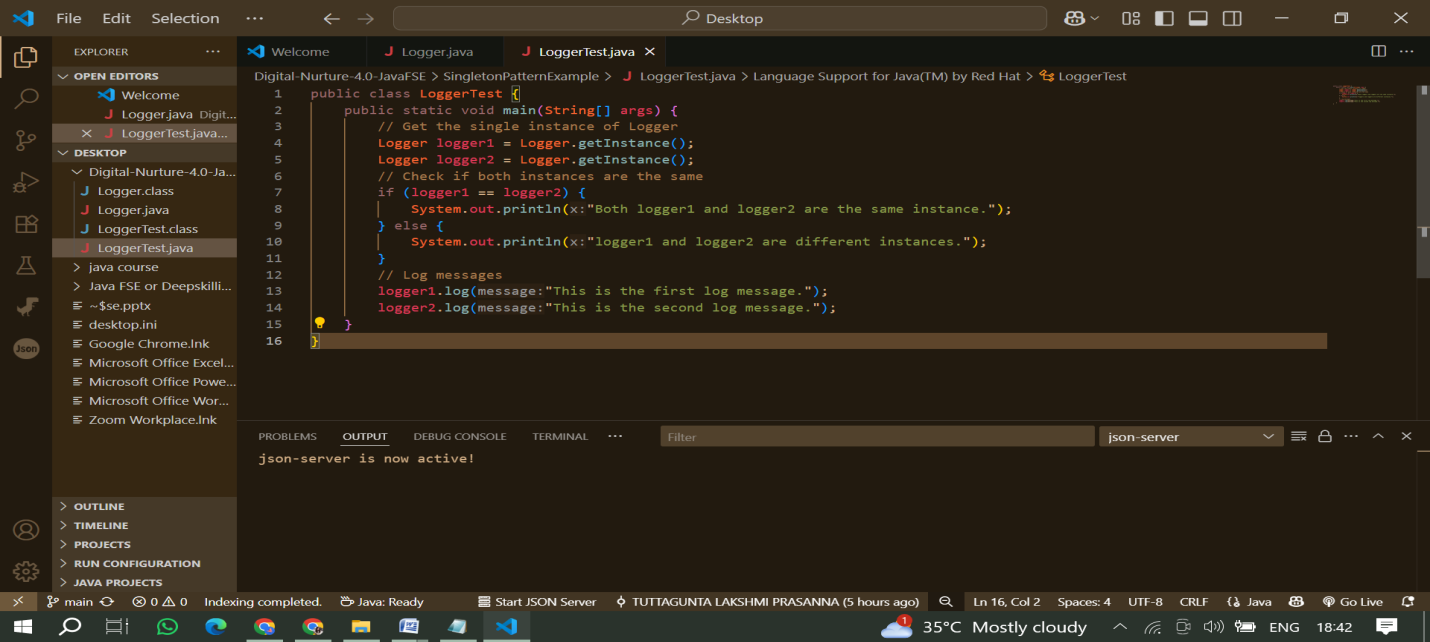
1.Create a New Java Class for Testing:

• Right-click on the source folder again and select New File.

• Name the file LoggerTest.java.

2. Implement the Test Class:

• Open LoggerTest.java and add the following code:



1. Compile the Files & Run the Application :

Compile the Logger.java and LoggerTest.java. Then, Run the LoggerTest.java file.

**OUTPUT :**

