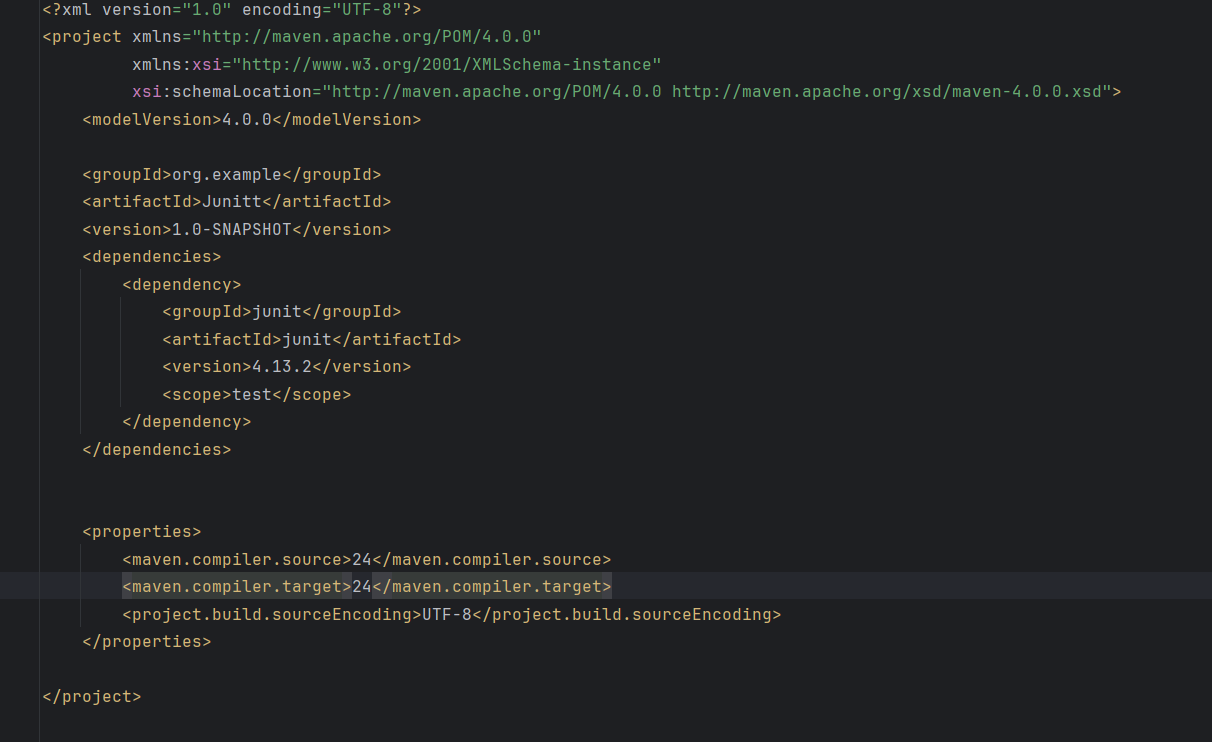
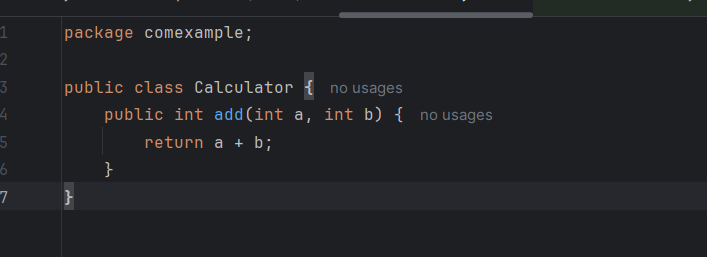
**Cognizant Deep Skilling Mandatory Hands-On Questions**

**TOPIC- JUnit Testing Exercises**

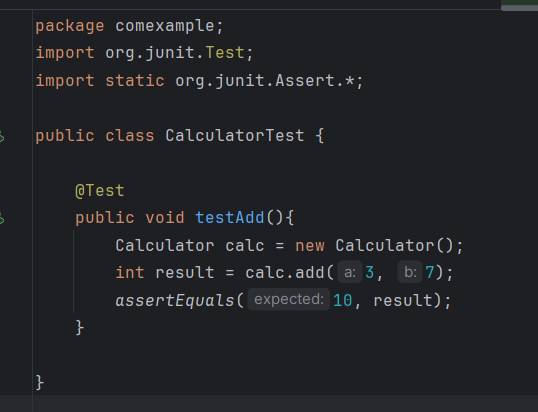
Exercise 1: Setting Up JUnit

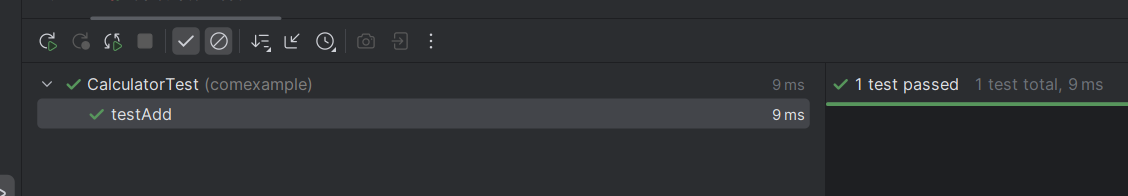
Scenario: You need to set up JUnit in your Java project to start writing unit tests.

Ans) pom.xml:

CalculatorClass:

NewTest Class:



OUTPUT:

Q.3 You need to use different assertions in JUnit to validate your test results.

Steps:

1. Write tests using various JUnit assertions.

Solution Code:

public class AssertionsTest {

@Test

public void testAssertions() {

// Assert equals

assertEquals(5, 2 + 3);

// Assert true

assertTrue(5 > 3);

// Assert false

assertFalse(5 < 3);

// Assert null

assertNull(null);

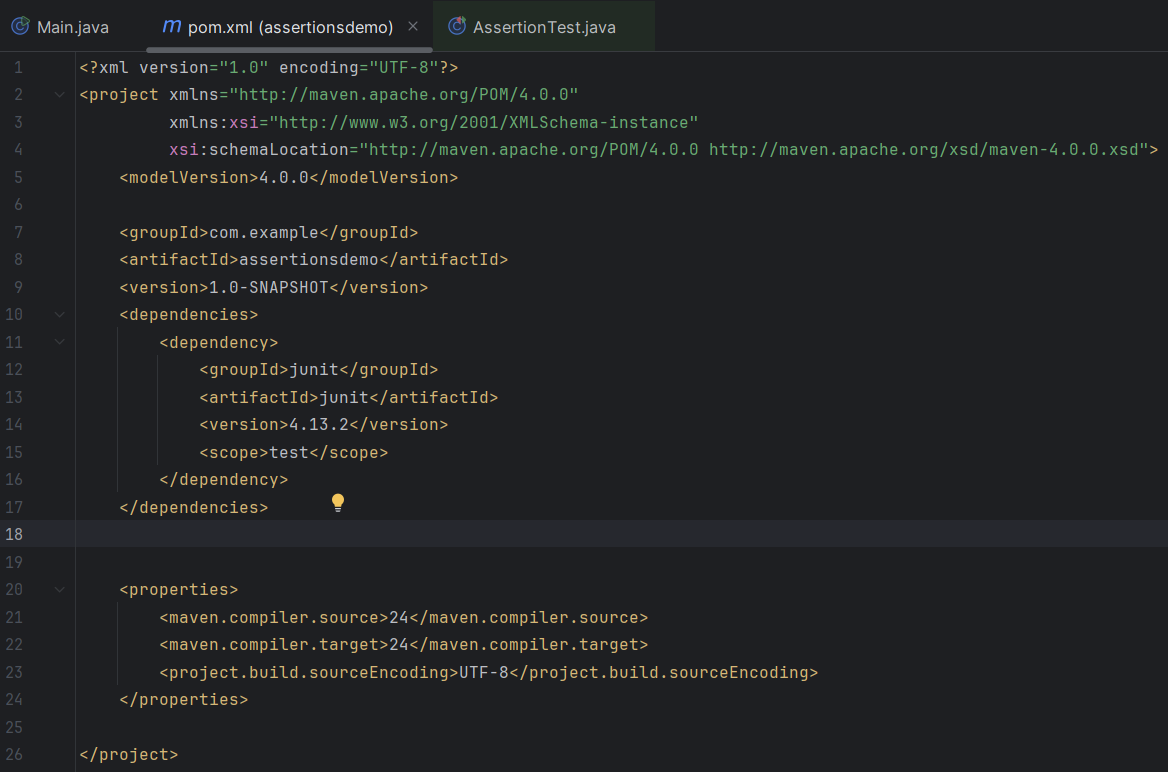
// Assert not null

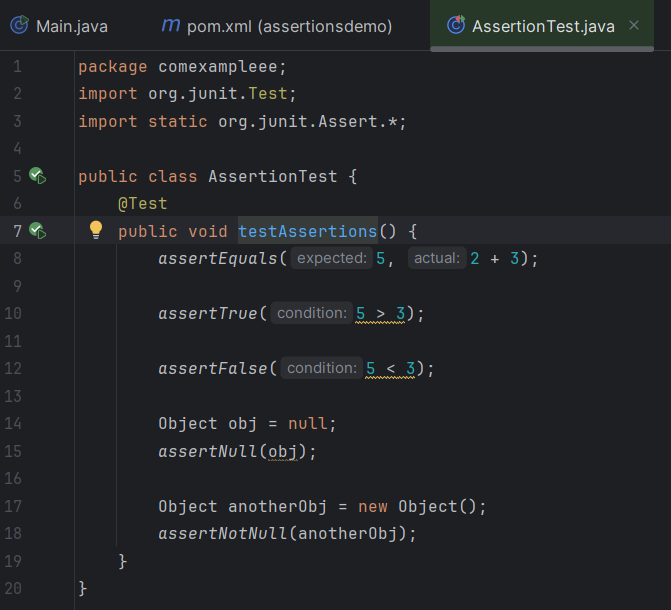
assertNotNull(new Object());

}

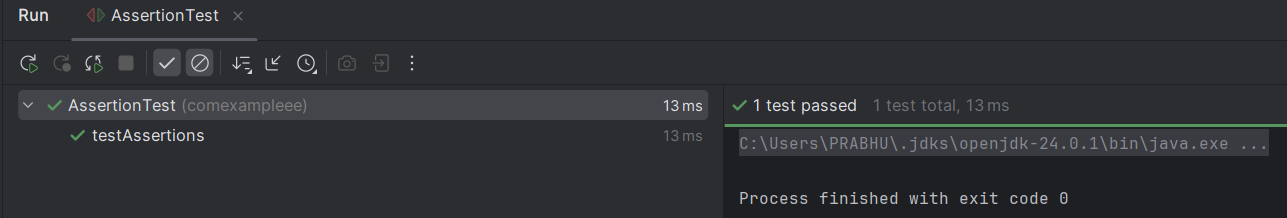
} Give the code for this with detailed steps

**Ans)**

Step1: Create a maven project with pom.xml file and dependencies to it:

Step2: Create the Test Class:

Output:



Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and

Teardown Methods in JUnit

Scenario:

You need to organize your tests using the Arrange-Act-Assert (AAA) pattern and use setup

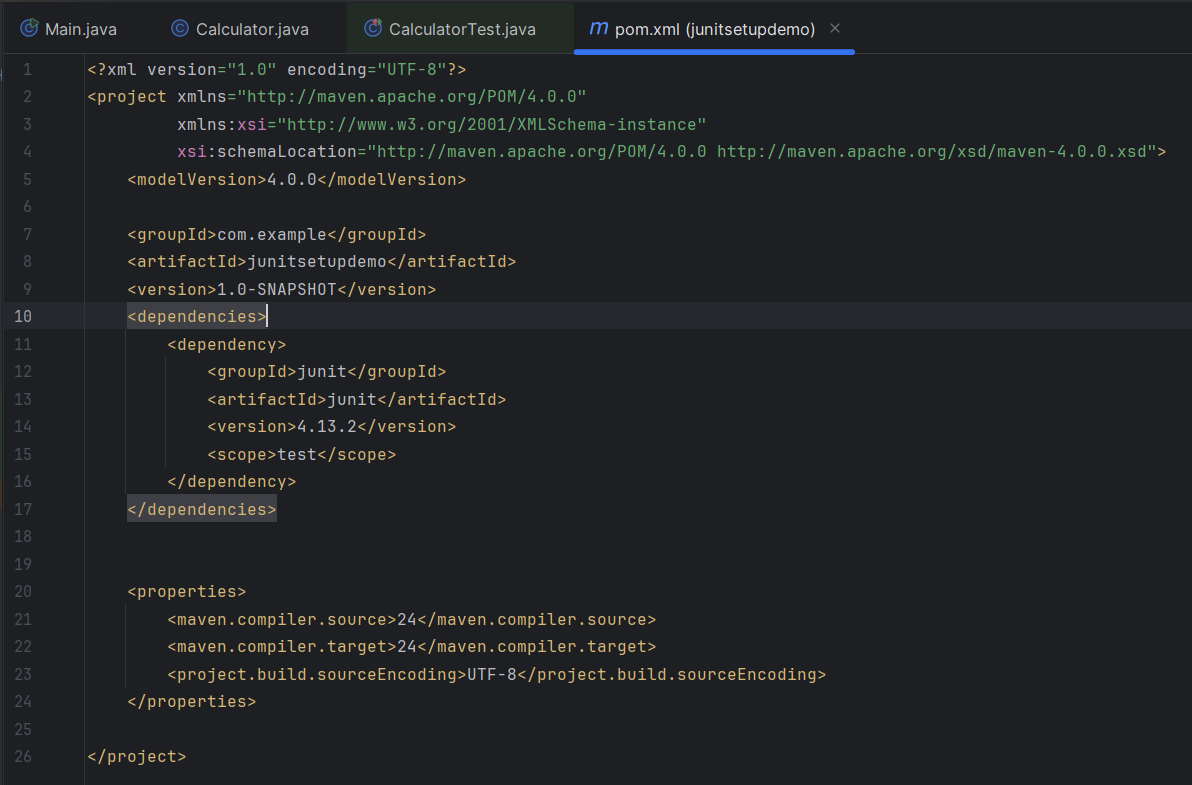
and teardown methods.

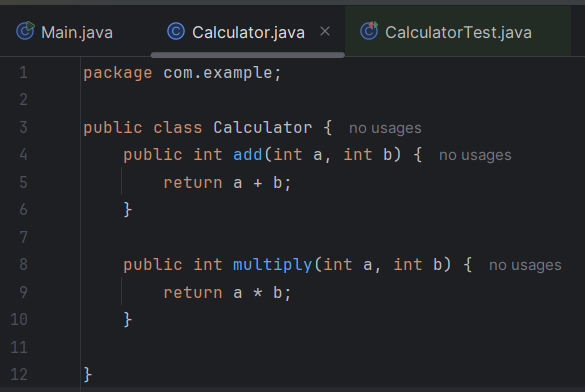
Steps:

1. Write tests using the AAA pattern.

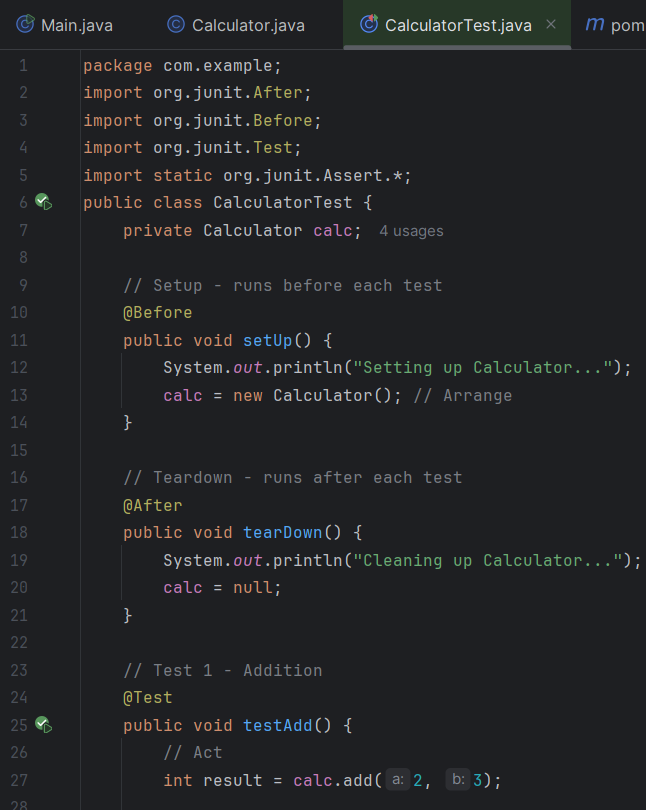
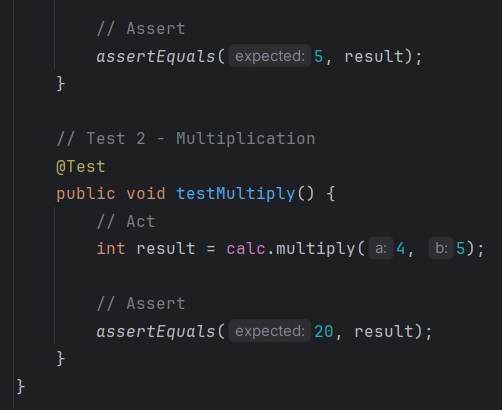
2. Use @Before and @After annotations for setup and teardown methods.

Ans)

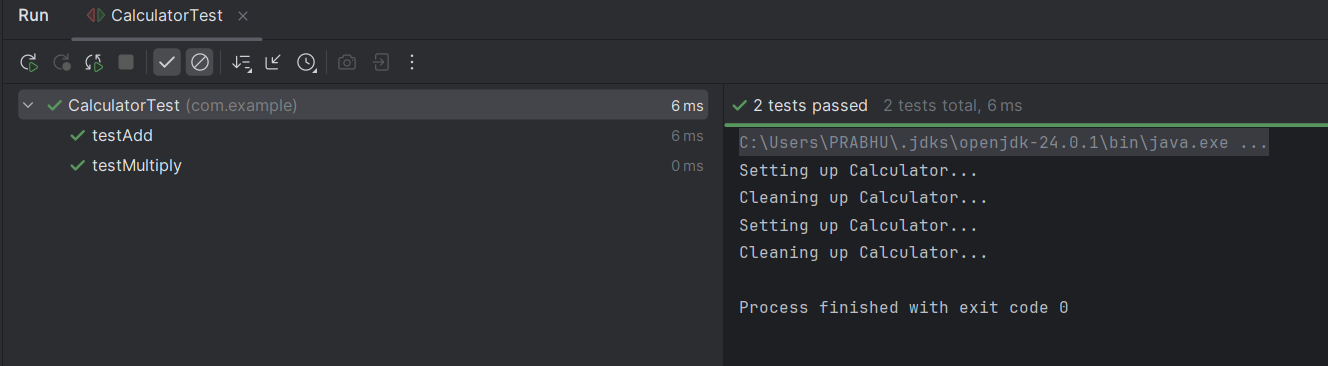
Step1:

Step-2:

Step-3:



OUTPUT



**Cognizant Deep Skilling Mandatory Hands-On Questions**

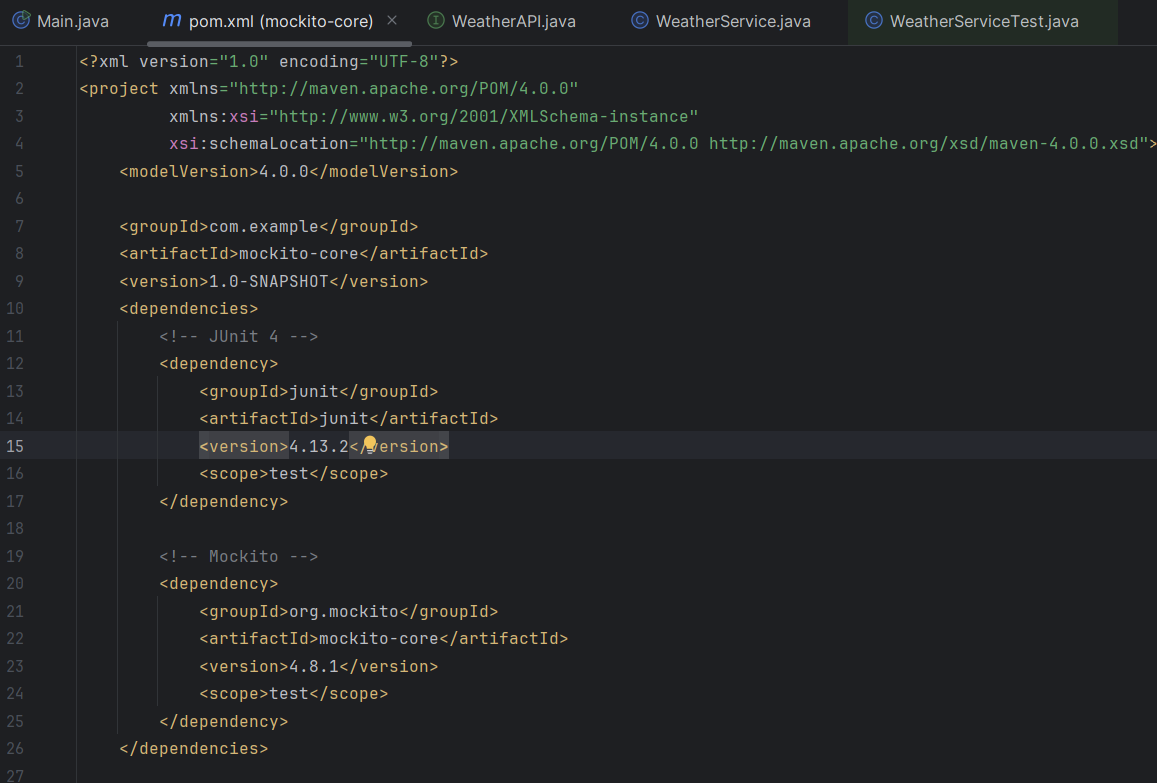
**TOPIC- Mockito Hands-On Exercises**

Exercise 1: Mocking and Stubbing

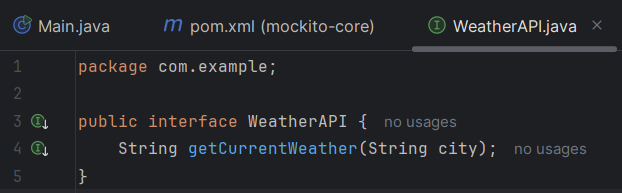
Scenario: You need to test a service that depends on an external API. Use Mockito to mock the external API and stub its methods.

Ans)

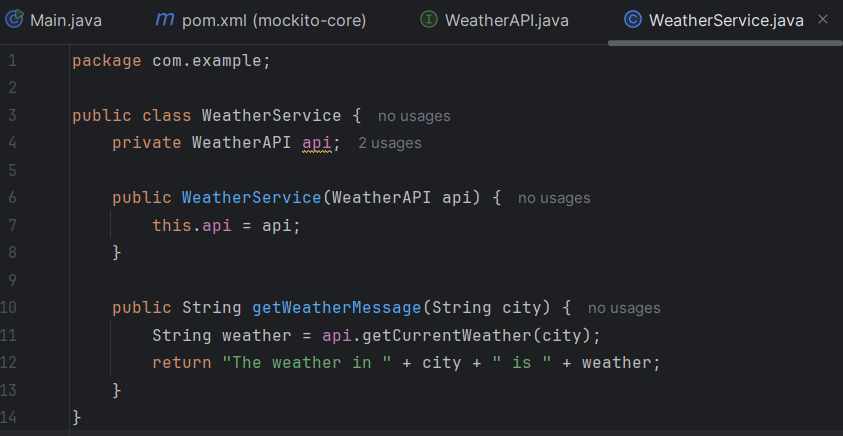
Step1: Create a Maven Project and Add Dependencies:



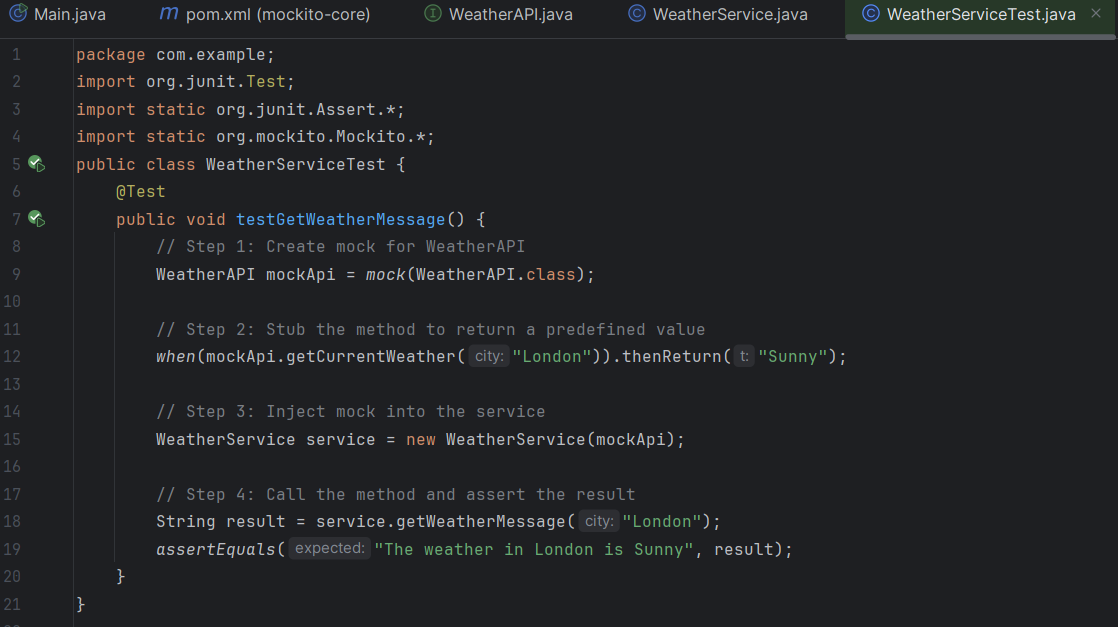
Step2: Create the External API Interface:



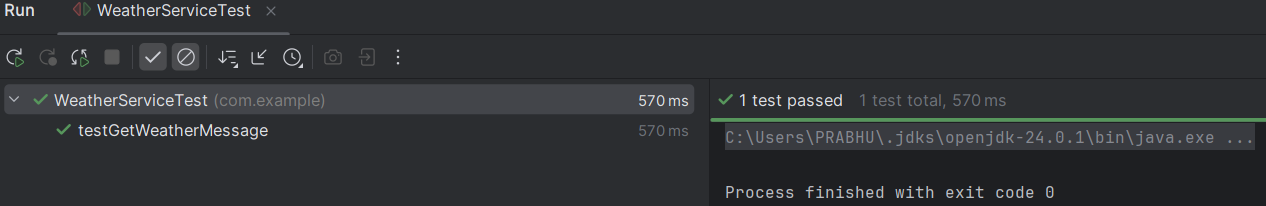
Step3: Create the Service That Uses the API:



Step4: Create the Test Class Using Mockito:



OUTPUT:



Exercise 2: Verifying Interactions

Scenario:

You need to ensure that a method is called with specific arguments.

Steps:

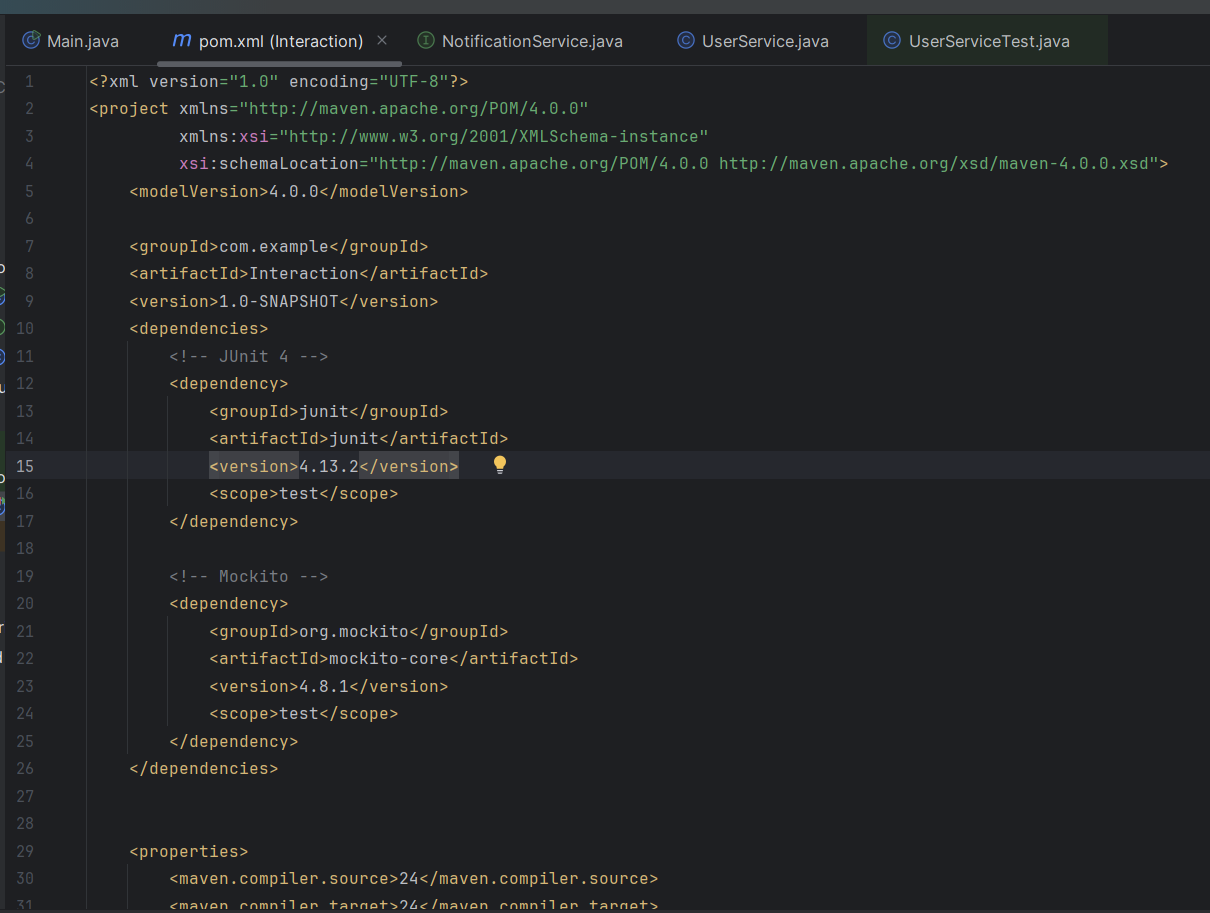
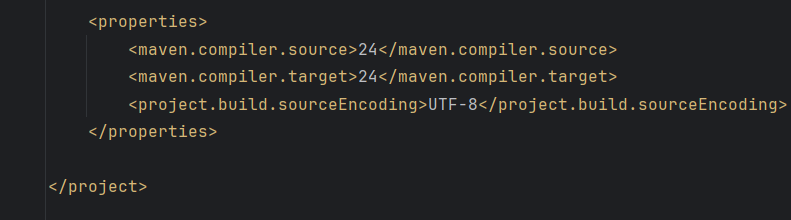
1. Create a mock object.

2. Call the method with specific arguments.

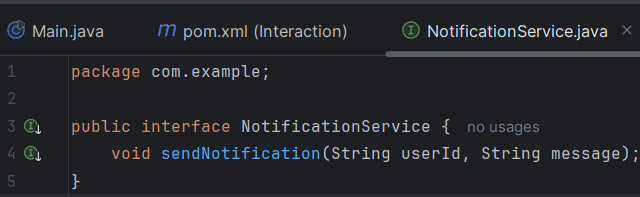
3. Verify the interaction.

Ans)

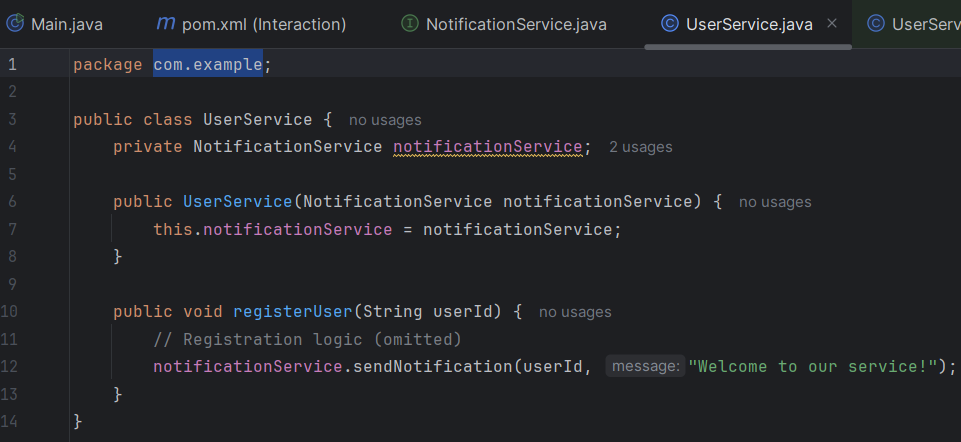
Step1: Create a Maven Project and Add Dependencies:



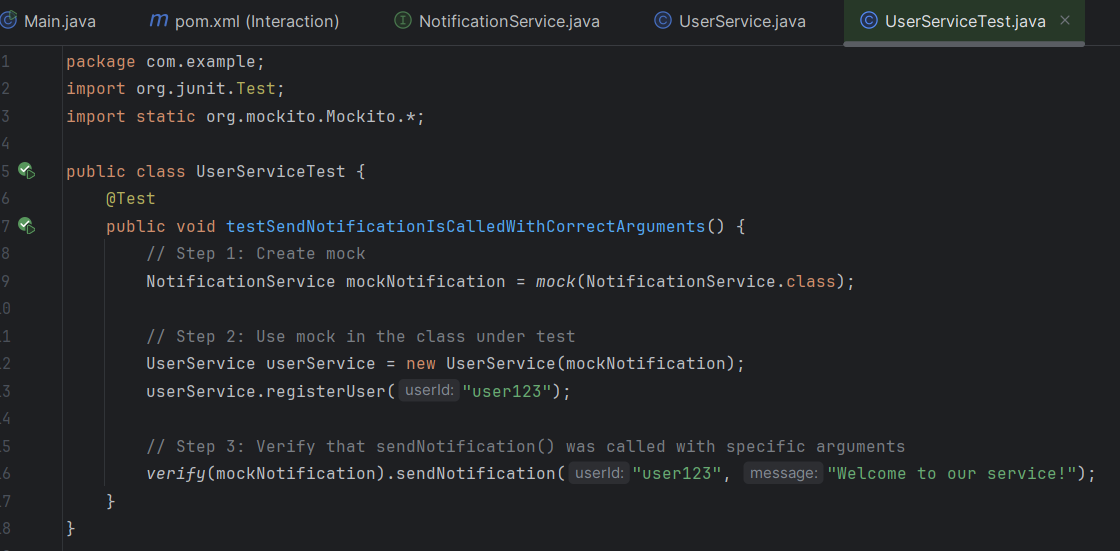
Step2: Define the External Dependency (to Mock):



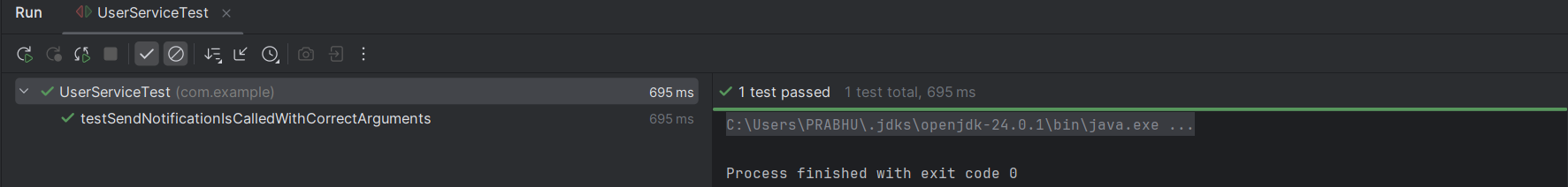
Step3: Create the Class Under Test:



Step4: Create the Test Class to Verify Interactions:



OUTPUT:



**Cognizant Deep Skilling Mandatory Hands-On Questions**

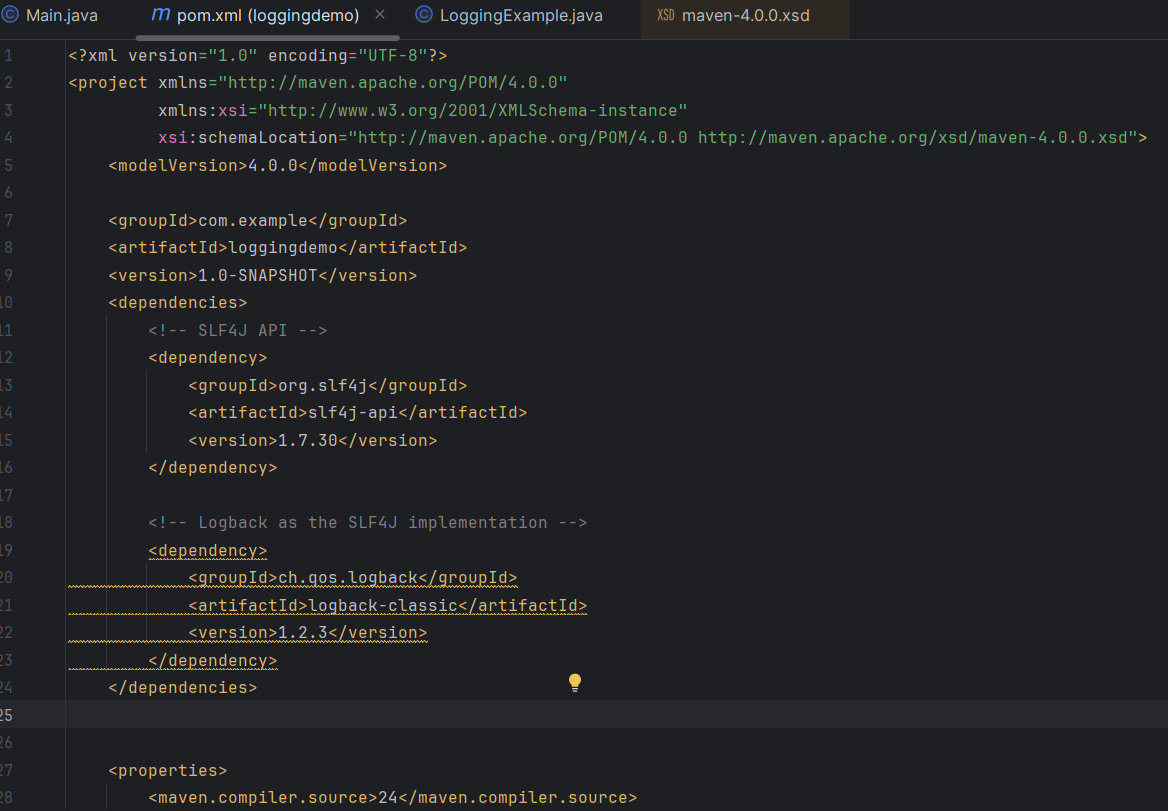
**TOPIC- Logging using SLF4J**

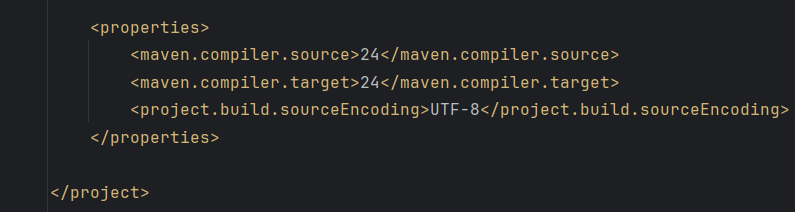
Exercise 1: Logging Error Messages and Warning Levels

Task: Write a Java application that demonstrates logging error messages and warning levels using SLF4J.

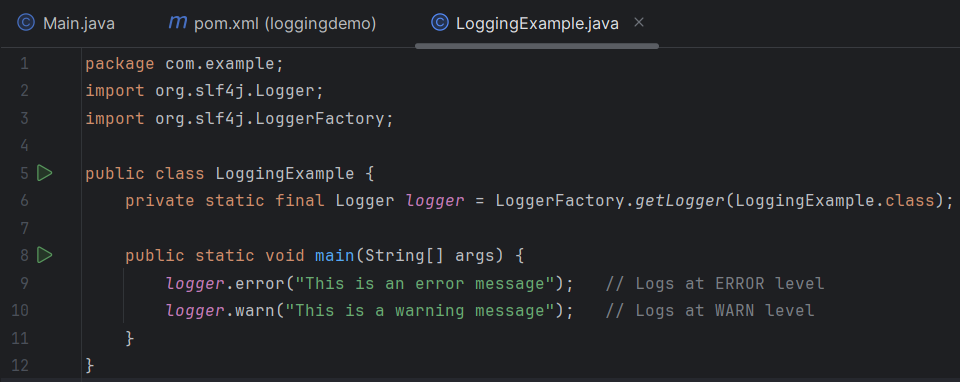
Ans)

STEP 1: Create a Maven Project and add SLF4J and Logback Dependencies in pom.xml:





STEP 2: Create the Java Logging Class



OUTPUT:

