**1.JUnit\_Basic Testing Exercises**

**Exercise 1: Setting Up Junit**

**Code:**

**Pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>JUnitDemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**CalculatorTest.java:**

import static org.junit.Assert.\*;

import org.junit.Test;

public class CalculatorTest {

@Test

public void testAdd() {

int a = 2, b = 3;

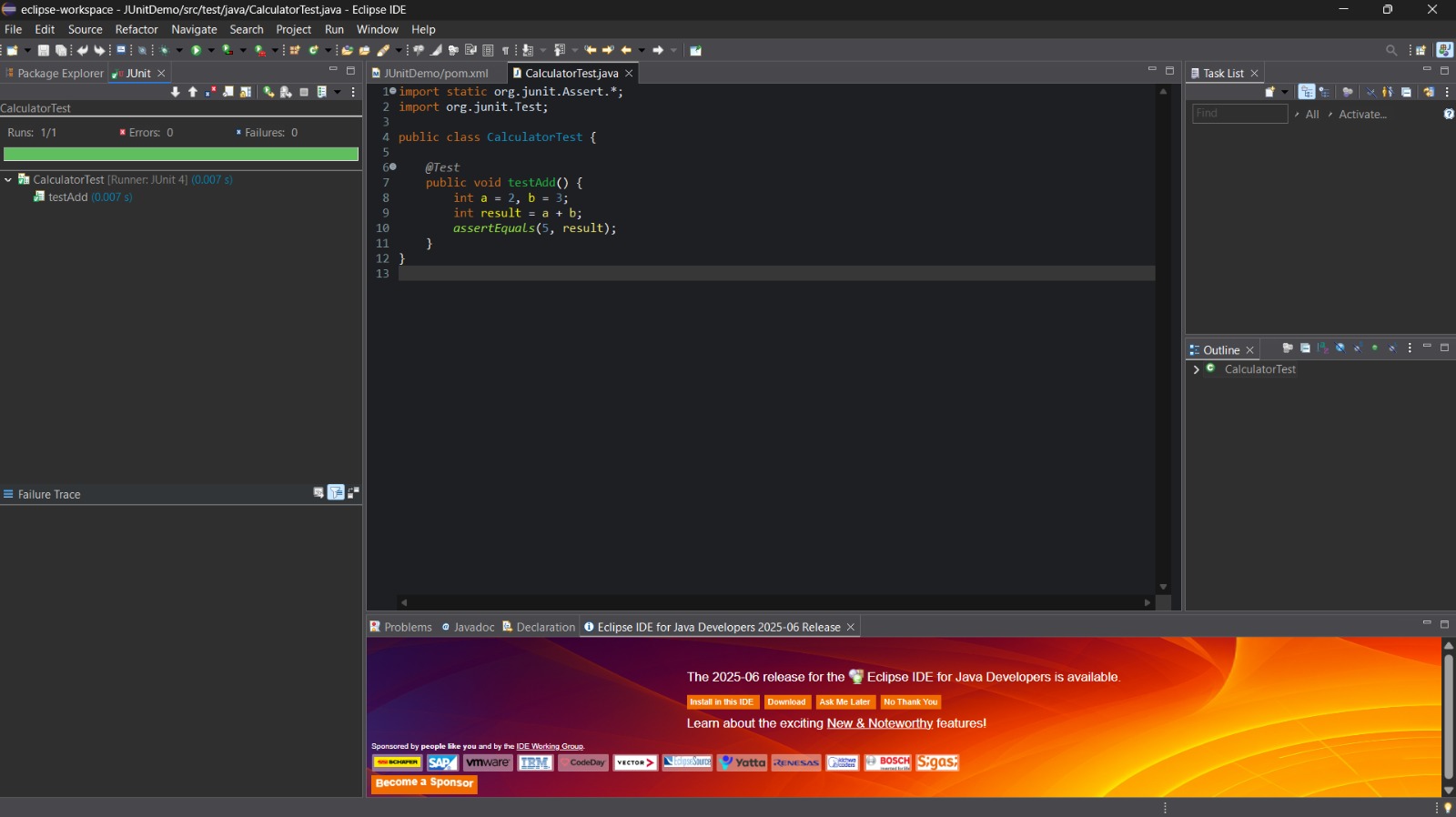
int result = a + b;

assertEquals(5, result);

}

}

**Output:**



**Exercise 3: Assertions in JUnit**

**Code:**

**Pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>JUnitDemo</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<!-- JUnit 5 dependency -->

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Maven Surefire Plugin for running JUnit 5 tests -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0</version>

</plugin>

</plugins>

</build>

</project>

**AssertionsTest:**

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

public class AssertionsTest {

@Test

public void testAssertions() {

// assertEquals

assertEquals(5, 2 + 3);

// assertTrue

assertTrue(5 > 3);

// assertFalse

assertFalse(5 < 3);

// assertNull

assertNull(null);

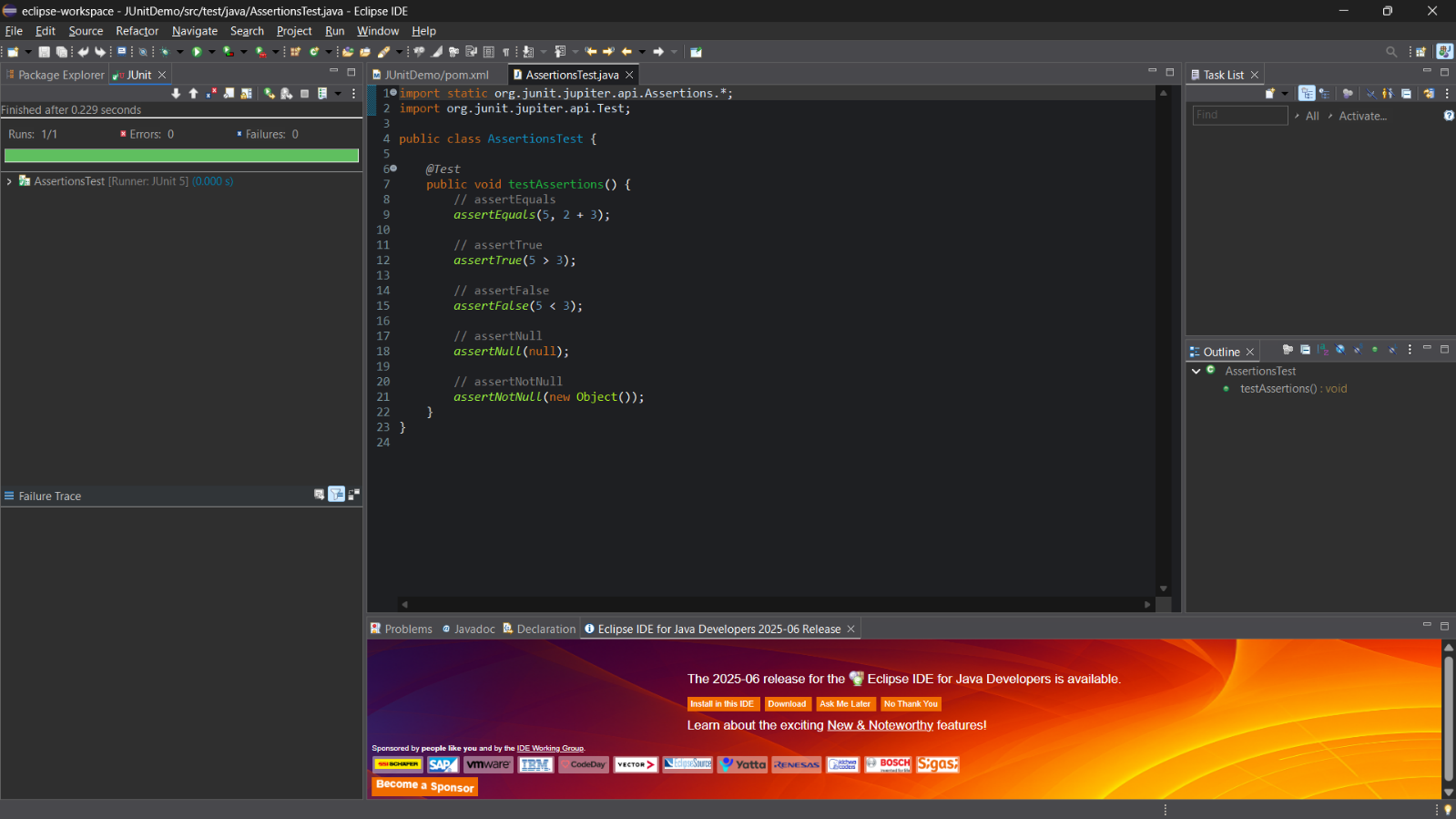
// assertNotNull

assertNotNull(new Object());

}

}

**Output:**

****

**Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit.**

**Code:**

**Pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>JUnitDemo</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0</version>

</plugin>

</plugins>

</build>

</project>

**Calculator.java:**

package com.example;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

**AAAExampleTest.java:**

package com.example;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class AAAExampleTest {

private Calculator calculator;

// Setup method: This runs before each test

@BeforeEach

public void setup() {

// Arrange: Initialize objects or variables

calculator = new Calculator();

}

// Teardown method: This runs after each test

@AfterEach

public void teardown() {

// Clean up resources after the test

calculator = null;

}

@Test

public void testAddition() {

// Arrange: (already done in setup)

int a = 5;

int b = 3;

// Act: Perform the action or call the method

int result = calculator.add(a, b);

// Assert: Verify the outcome

assertEquals(8, result);

}

@Test

public void testSubtraction() {

// Arrange: (already done in setup)

int a = 5;

int b = 3;

// Act: Perform the action or call the method

int result = calculator.subtract(a, b);

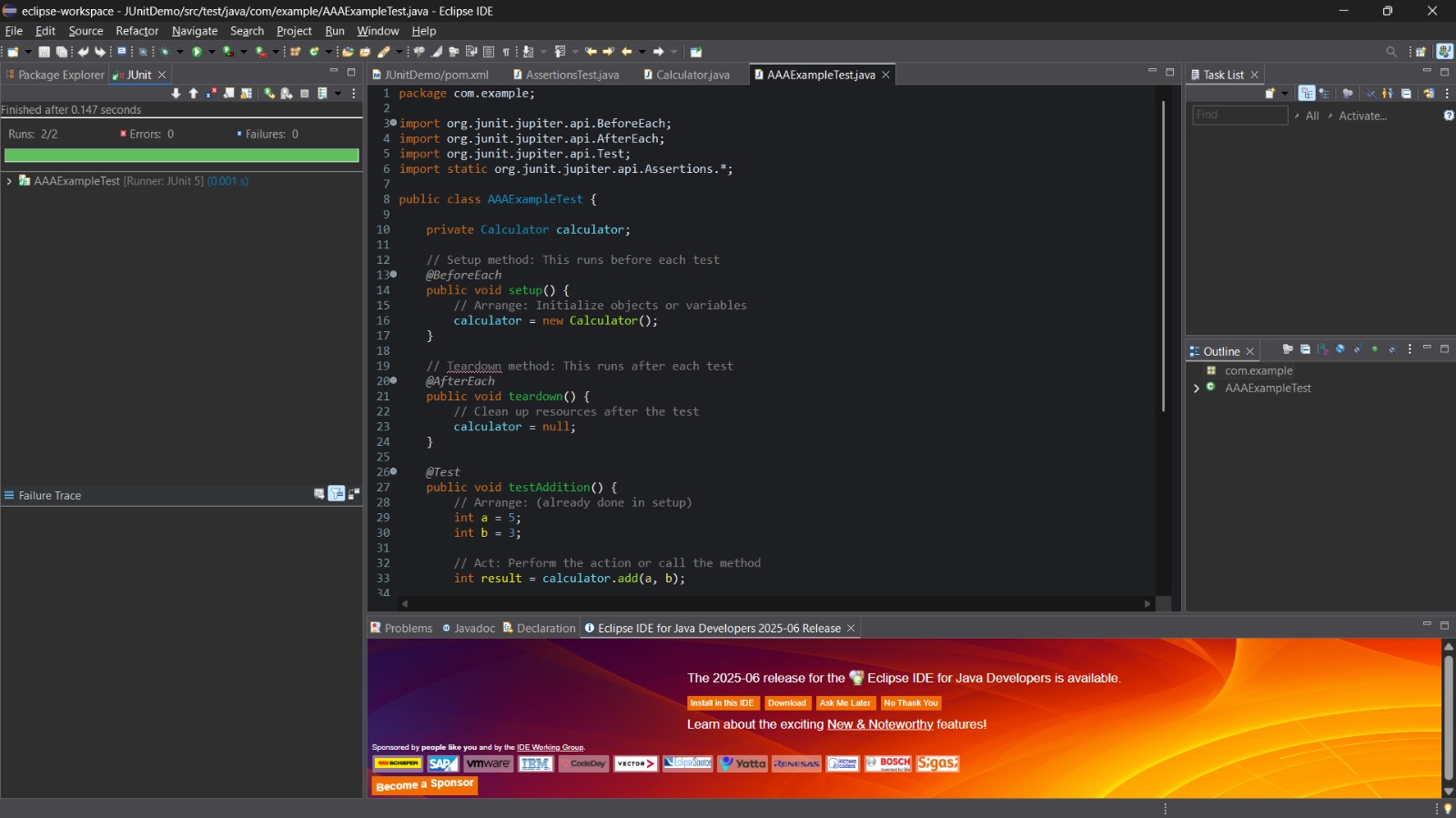
// Assert: Verify the outcome

assertEquals(2, result);

}

}

**Output:**



**3. Mockito exercises**

**Exercise 1: Mocking and Stubbing**

**Code:**

**Pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>MokitDemo1</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- JUnit 5 dependency -->

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

<!-- Mockito dependency for mocking -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.2.0</version> <!-- Use latest version -->

<scope>test</scope>

</dependency>

</dependencies>

</project>

**MyServiceTest.java:**

package com.example;

import static org.mockito.Mockito.\*;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testExternalApi() {

// 1. Create a mock object for the ExternalApi

ExternalApi mockApi = Mockito.mock(ExternalApi.class);

// 2. Stub the method to return a predefined value

when(mockApi.getData()).thenReturn("Mock Data");

// 3. Create the service, passing in the mocked API

MyService service = new MyService(mockApi);

// 4. Call the method you want to test

String result = service.fetchData();

// 5. Assert the expected value

assertEquals("Mock Data", result);

}

}

**MyService.java:**

package com.example;

public class MyService {

private final ExternalApi externalApi;

// Constructor to inject the dependency

public MyService(ExternalApi externalApi) {

this.externalApi = externalApi;

}

// Method that calls the external API

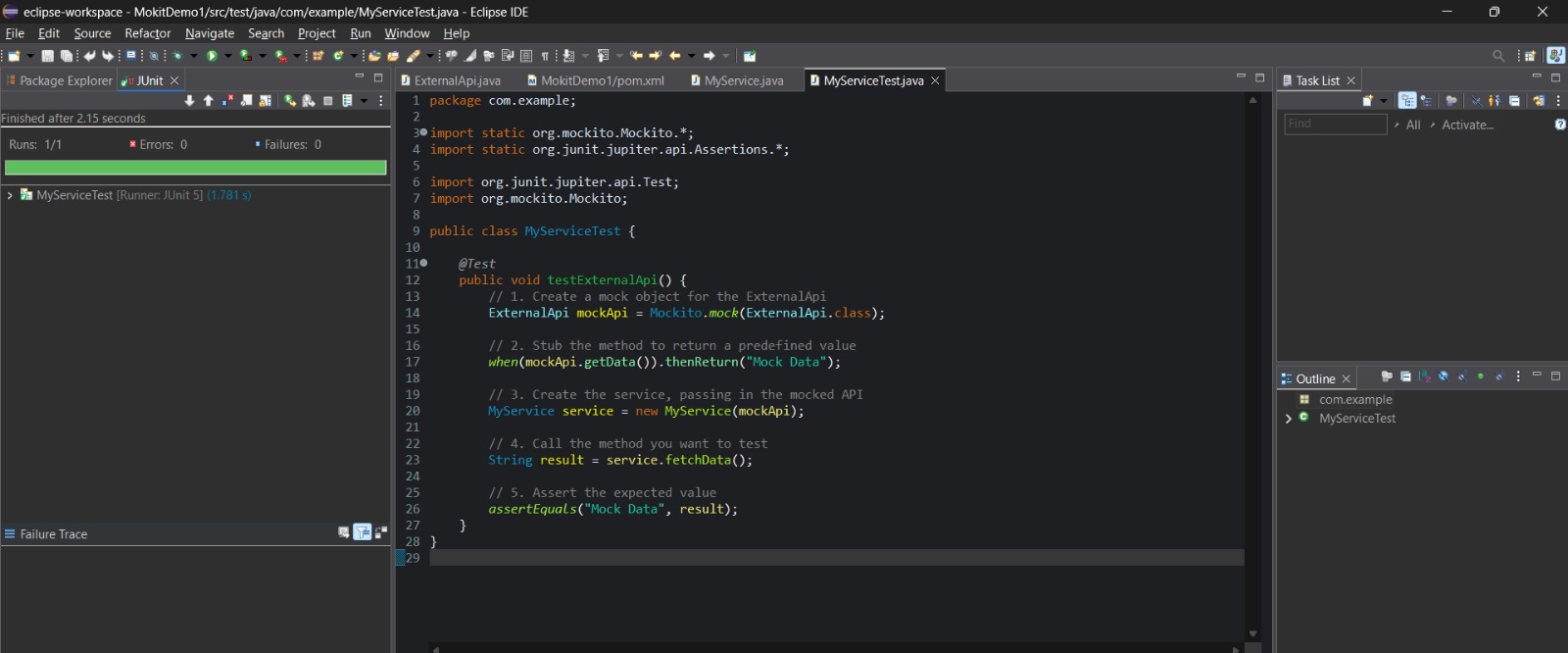
public String fetchData() {

return externalApi.getData();

}

}

**Output:**



**Exercise 2: Verifying Interactions**

**Code:**

**Pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>MokitDemo1</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- JUnit 5 -->

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

<!-- Mockito -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.12.0</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**MyServiceTest.java:**

package com.example;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

// Step 1: Mock ExternalApi

ExternalApi mockApi = Mockito.mock(ExternalApi.class);

// Step 2: Inject mock into service

MyService service = new MyService(mockApi);

// Step 3: Call method

service.fetchData();

// Step 4: Verify interaction

verify(mockApi).getData();

}

}

**MyService.java:**

package com.example;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**ExternalApi.java:**

package com.example;

public class ExternalApi {

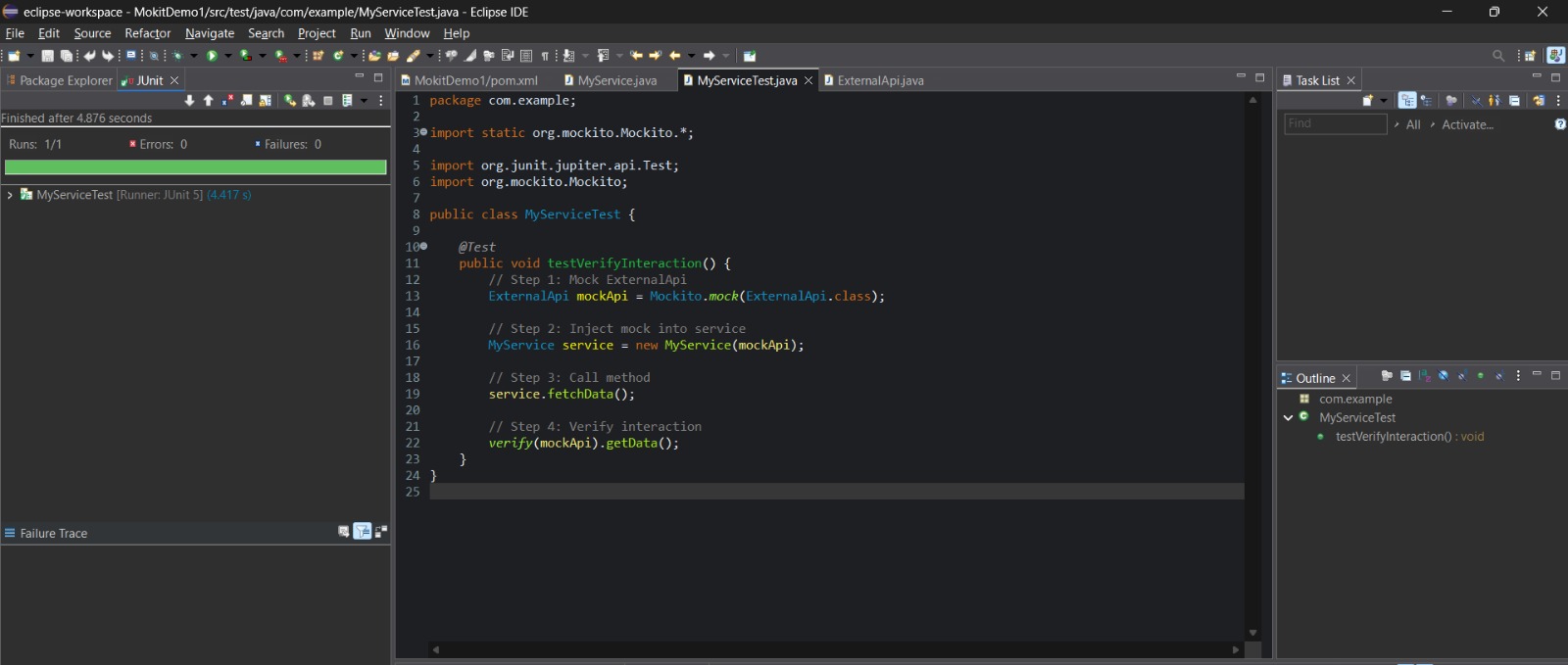
public String getData() {

return "Real Data";

}

}

**Output:**



**6. SL4J Logging exercises**

**Exercise 1: Logging Error Messages and Warning Levels**

**Code:**

**Pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>LoggingDemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.7.30</version>

</dependency>

<dependency>

<groupId>ch.qos.logback</groupId>

<artifactId>logback-classic</artifactId>

<version>1.2.3</version>

</dependency>

</dependencies>

</project>

**LoggingExample.java:**

package com.example;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class LoggingExample {

private static final Logger logger = LoggerFactory.getLogger(LoggingExample.class);

public static void main(String[] args) {

logger.error("This is an error message");

logger.warn("This is a warning message");

}

}

**Output:**

