**WEEK-02**

**MODULE-04:** **TEST DRIVEN DEVELOPMENT AND LOGGING FRAMEWORK**

**MANDATORY HANDSON**

**JUnit TESTING EXERCISE**

**EXERCISE-01:** **Setting Up Junit**

Calculator.java

**package** app;

**public** **class** Calculator {

**public** **int** add(**int** a, **int** b) {

**return** a + b;

}

}

CalculatorTest.java (JUnit Test Class)

**package** app;

**import** **static** org.junit.Assert.*assertEquals*;

**import** org.junit.Test;

**import** app.Calculator;

**public** **class** CalculatorTest {

@Test

**public** **void** testAdd() {

Calculator calc = **new** Calculator();

**int** result = calc.add(10, 20);

System.***out***.println("Running testAdd: 10 + 20 = " + result);

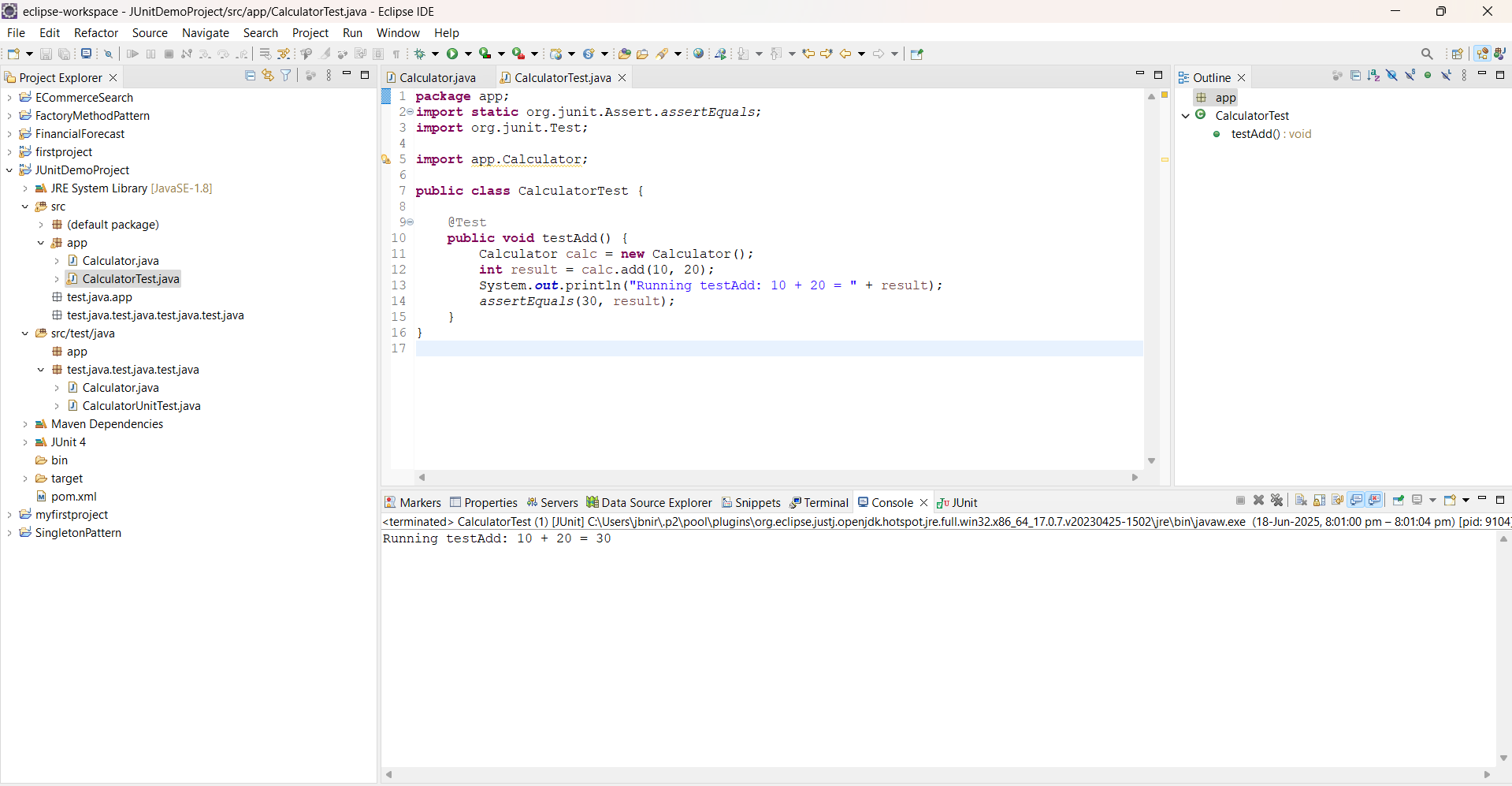
*assertEquals*(30, result);

}

}

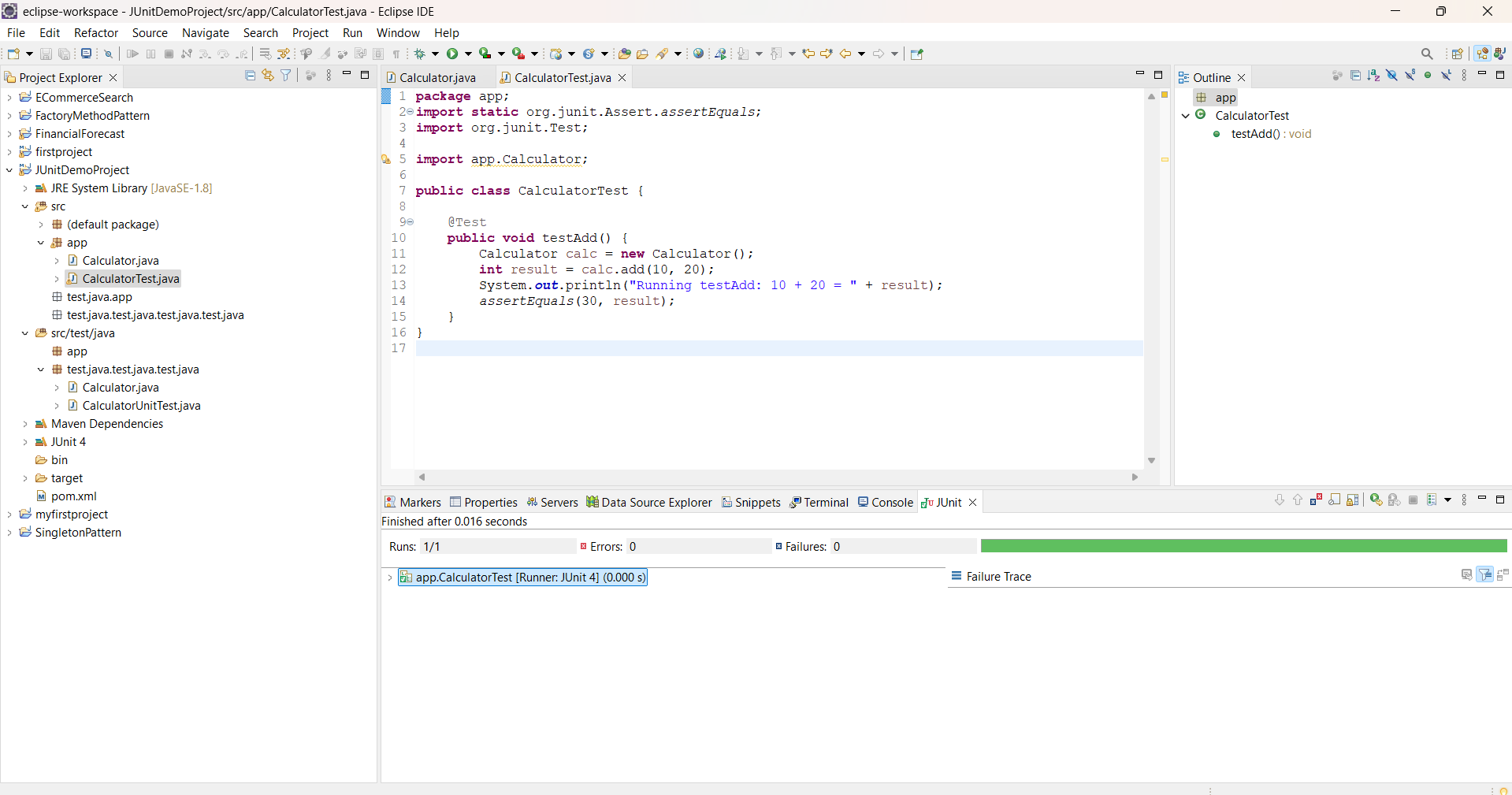
**OUTPUT**

**CONSOLE**

****



**Junit :**

****



**EXERCISE-03:** :**Assertions in Junit**

AssertionsTest.java

**package** app;

**import** **static** org.junit.Assert.\*;

**import** org.junit.Test;

**public** **class** AssertionsTest {

@Test

**public** **void** testAssertions() {

// Assert equals

System.***out***.println("Checking assertEquals...");

*assertEquals*(5, 2 + 3);

// Assert true

System.***out***.println("Checking assertTrue...");

*assertTrue*(5 > 3);

// Assert false

System.***out***.println("Checking assertFalse...");

*assertFalse*(5 < 3);

// Assert null

System.***out***.println("Checking assertNull...");

*assertNull*(**null**);

// Assert not null

System.***out***.println("Checking assertNotNull...");

*assertNotNull*(**new** Object());

// Final Success Message

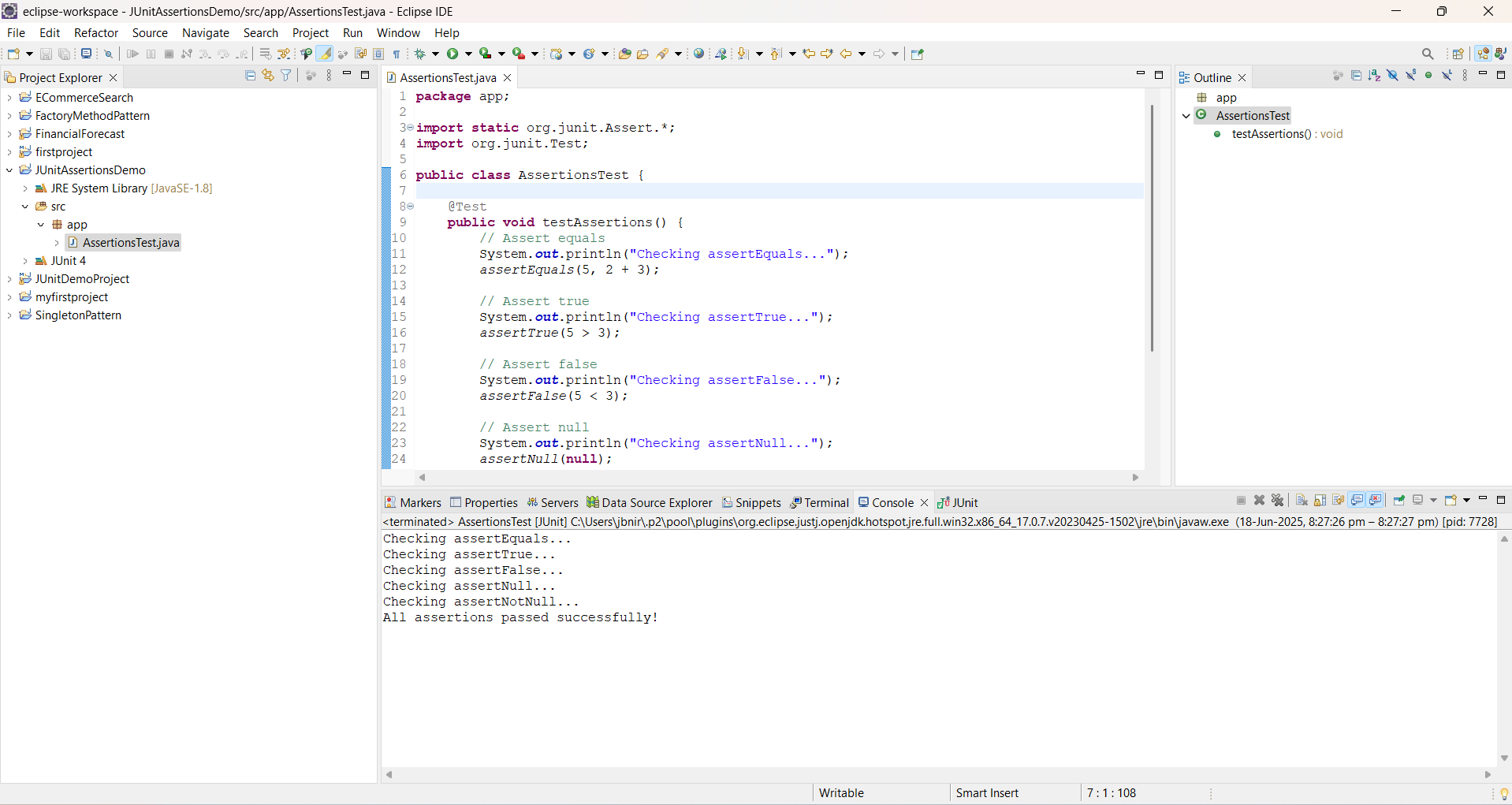
System.***out***.println("All assertions passed successfully!");

}

}

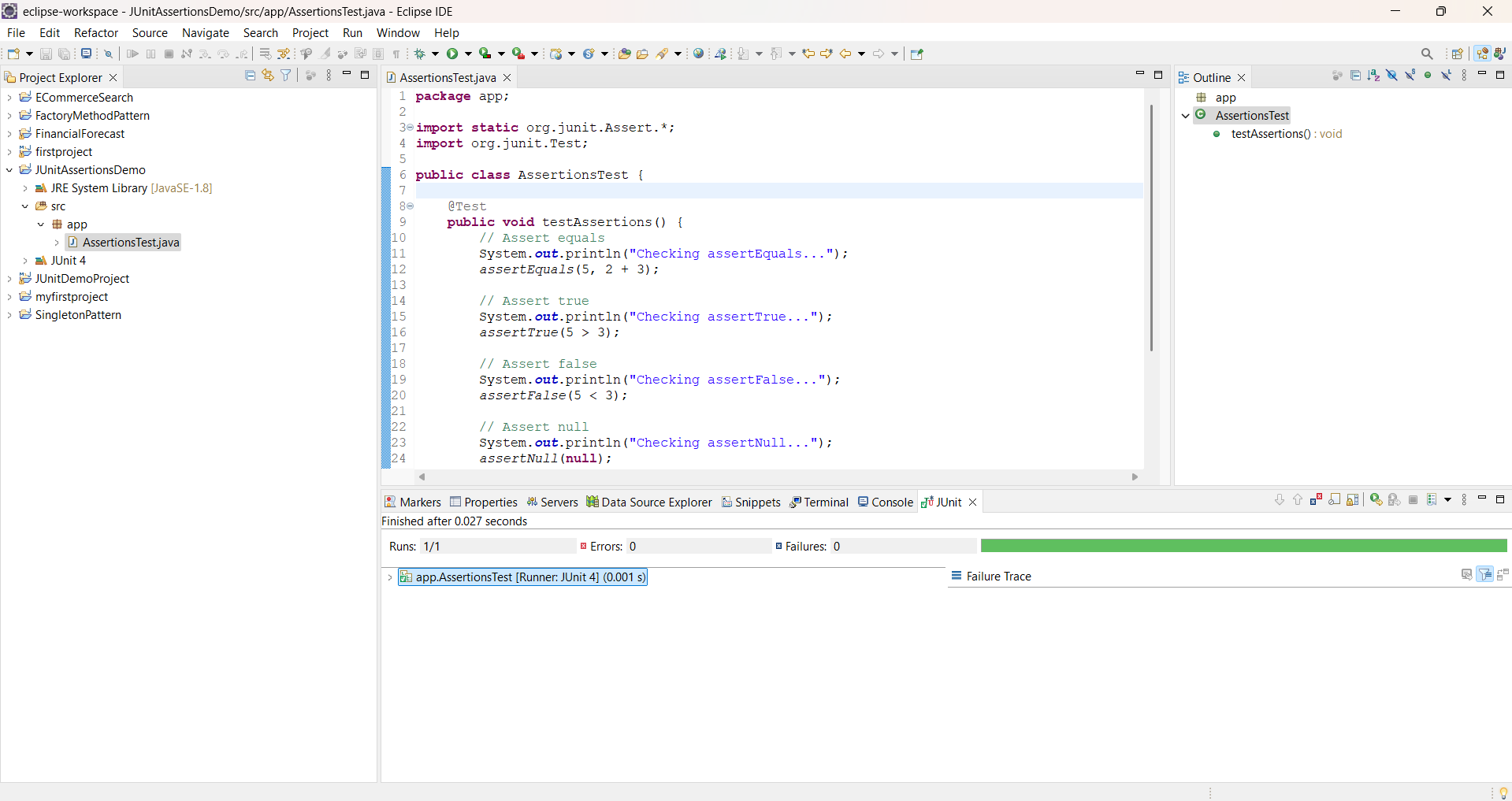
**OUTPUT**

**CONSOLE**

****



**JUNIT**

****



**EXERCISE-04:** **Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit**

Calculator.java

package app;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

CalculatorTest.java

package app;

import static org.junit.Assert.\*;

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

public class CalculatorTest {

private Calculator calc;

@Before

public void setUp() {

calc = new Calculator();

System.out.println("Setup: New Calculator Created");

}

@After

public void tearDown() {

calc = null;

System.out.println("Teardown: Calculator Destroyed");

}

@Test

public void testAdd() {

int a = 5;

int b = 3;

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

System.out.println("Running testAdd: " + a + " + " + b + " = " + result);

assertEquals(8, result);

}

@Test

public void testSubtract() {

int a = 10;

int b = 4;

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

System.out.println("Running testSubtract: " + a + " - " + b + " = " + result);

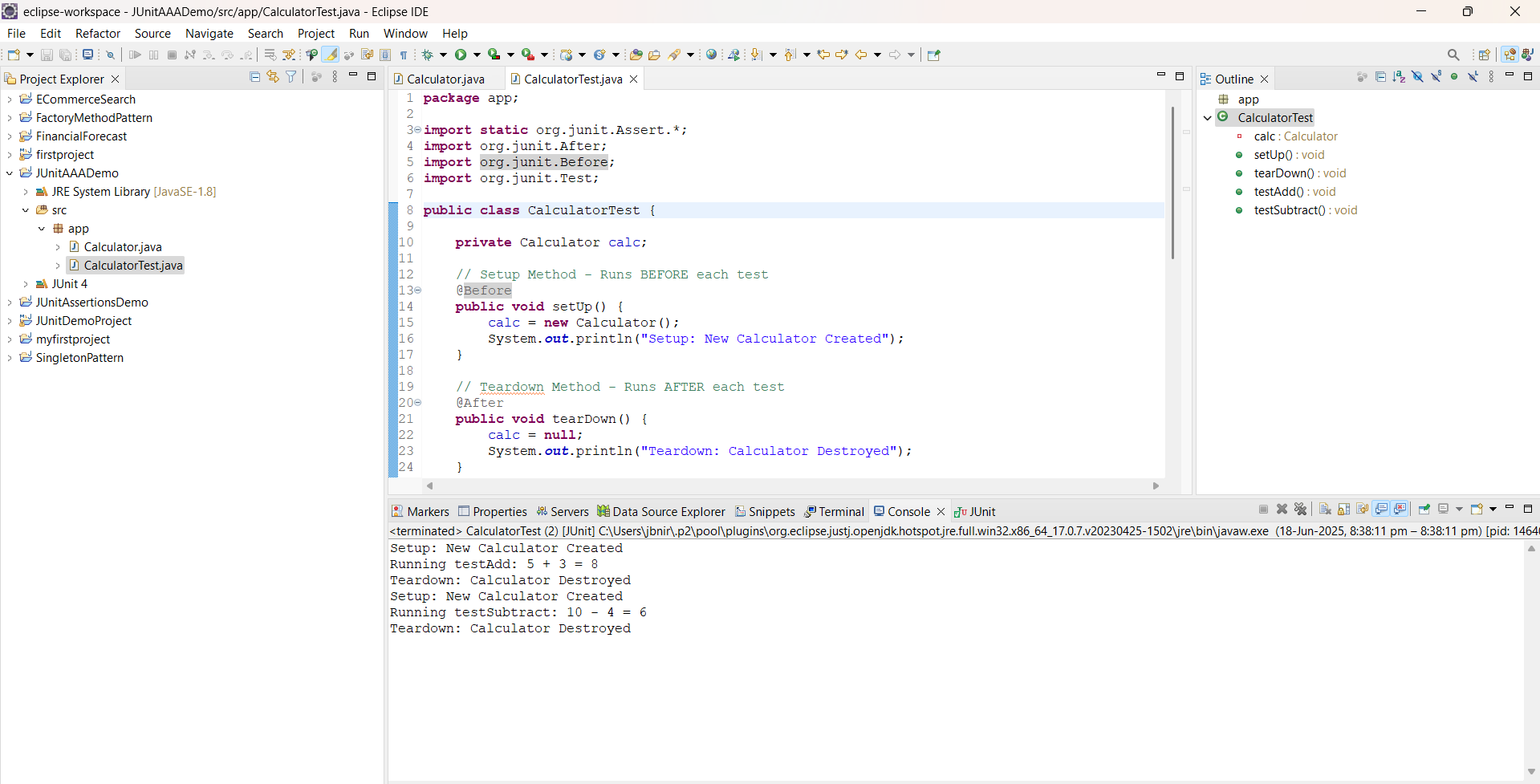
assertEquals(6, result);

}

}

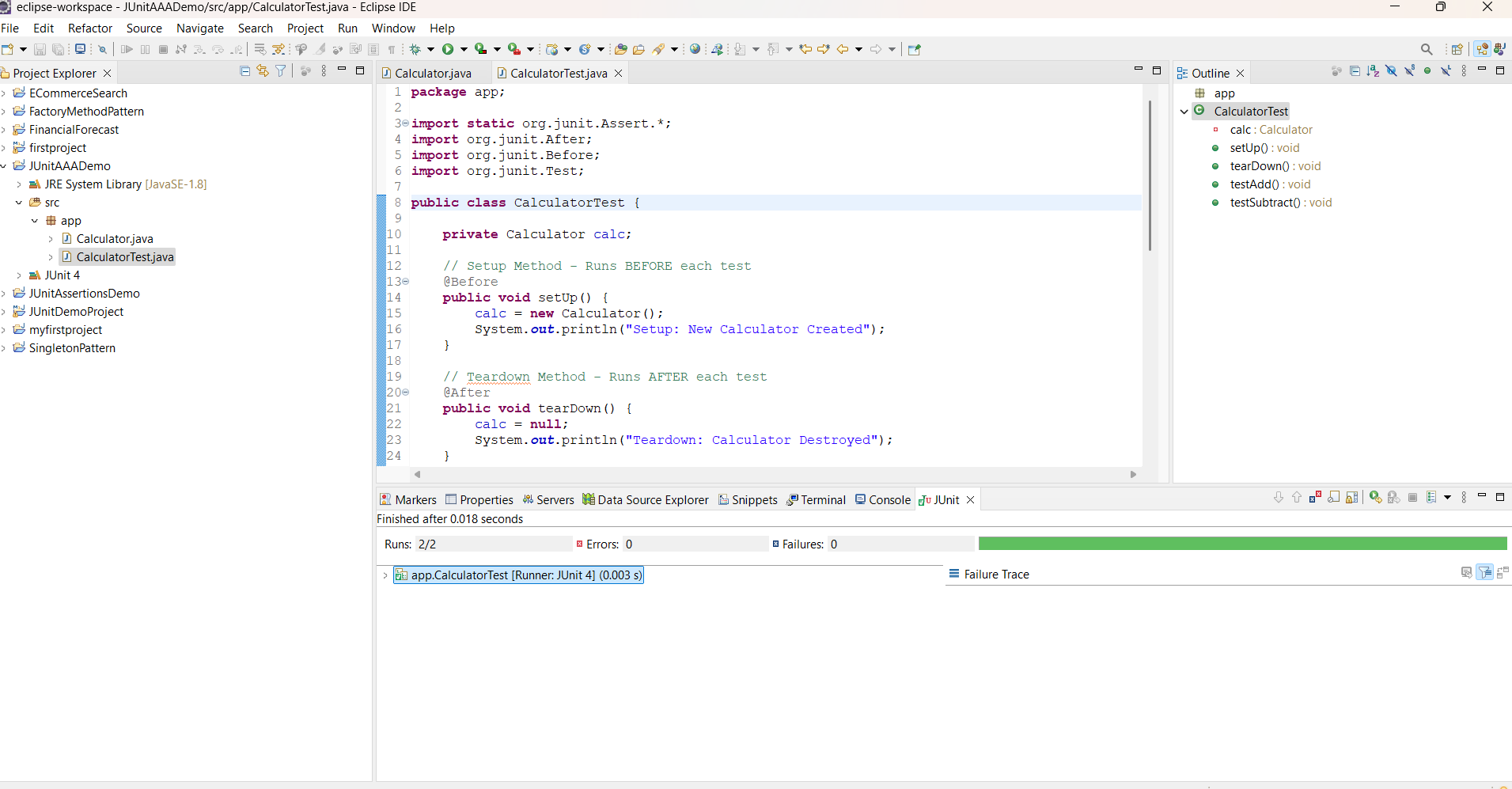
**OUTPUT**

**CONSOLE**

****



**JUNIT**

****



**MOCKITO EXERCISE**

**EXERCISE-01:** **Mocking and Stubbing**

pom.xml

<!-- JUnit 5 Dependency -->

<dependency>

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

<artifactId>junit-jupiter</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

<!-- Mockito Dependency -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.11.0</version>

<scope>test</scope>

</dependency>

ExternalApi

package com.example;

public interface ExternalApi {

String getData();

}

MyService.java

package com.example;

public class MyService {

private ExternalApi externalApi;

public MyService(ExternalApi externalApi) {

this.externalApi = externalApi;

}

public String fetchData() {

return externalApi.getData();

}

}

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() {

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

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

MyService service = new MyService(mockApi);

String result = service.fetchData();

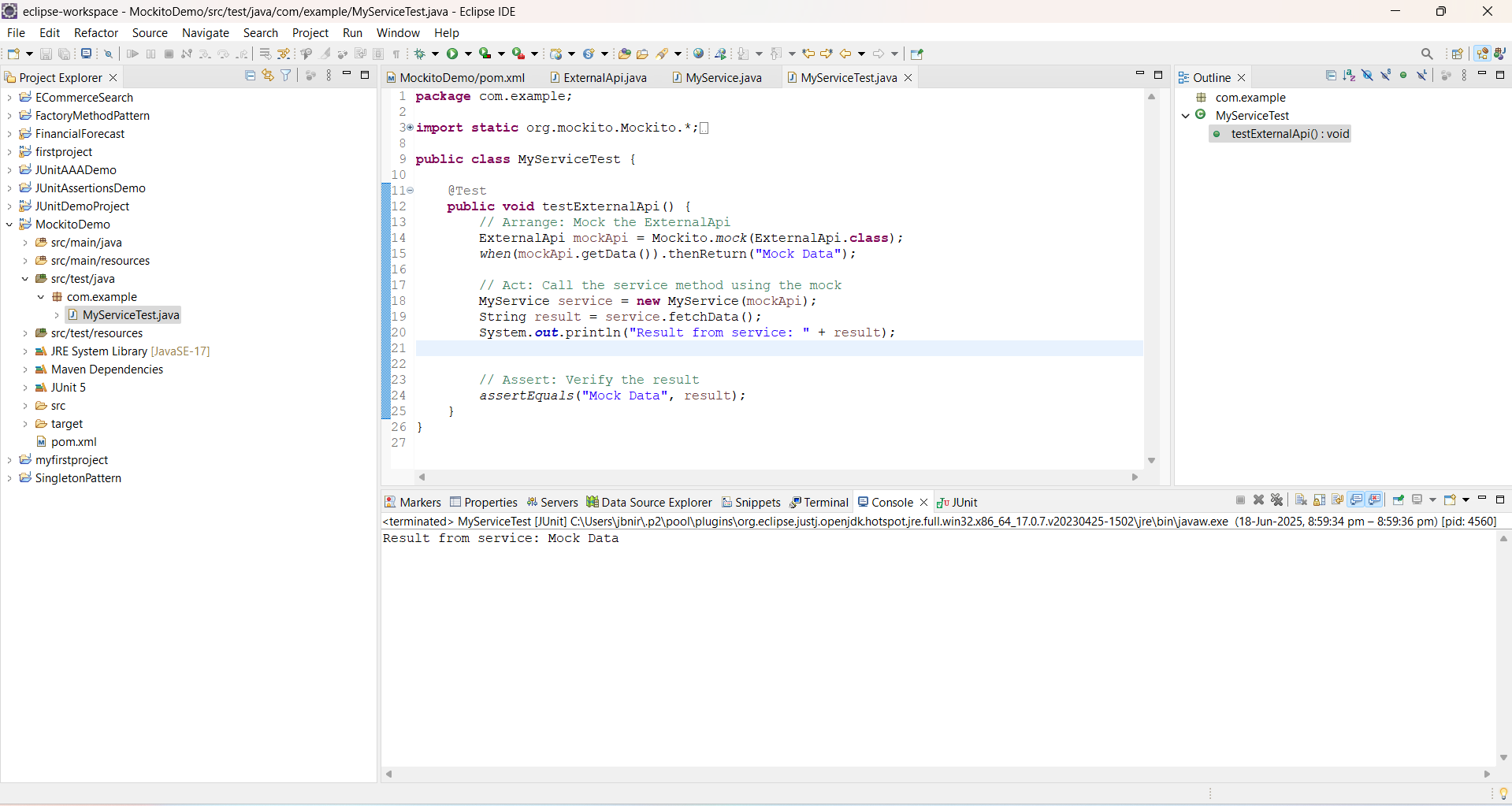
System.out.println("Result from service: " + result);

assertEquals("Mock Data", result);

} }

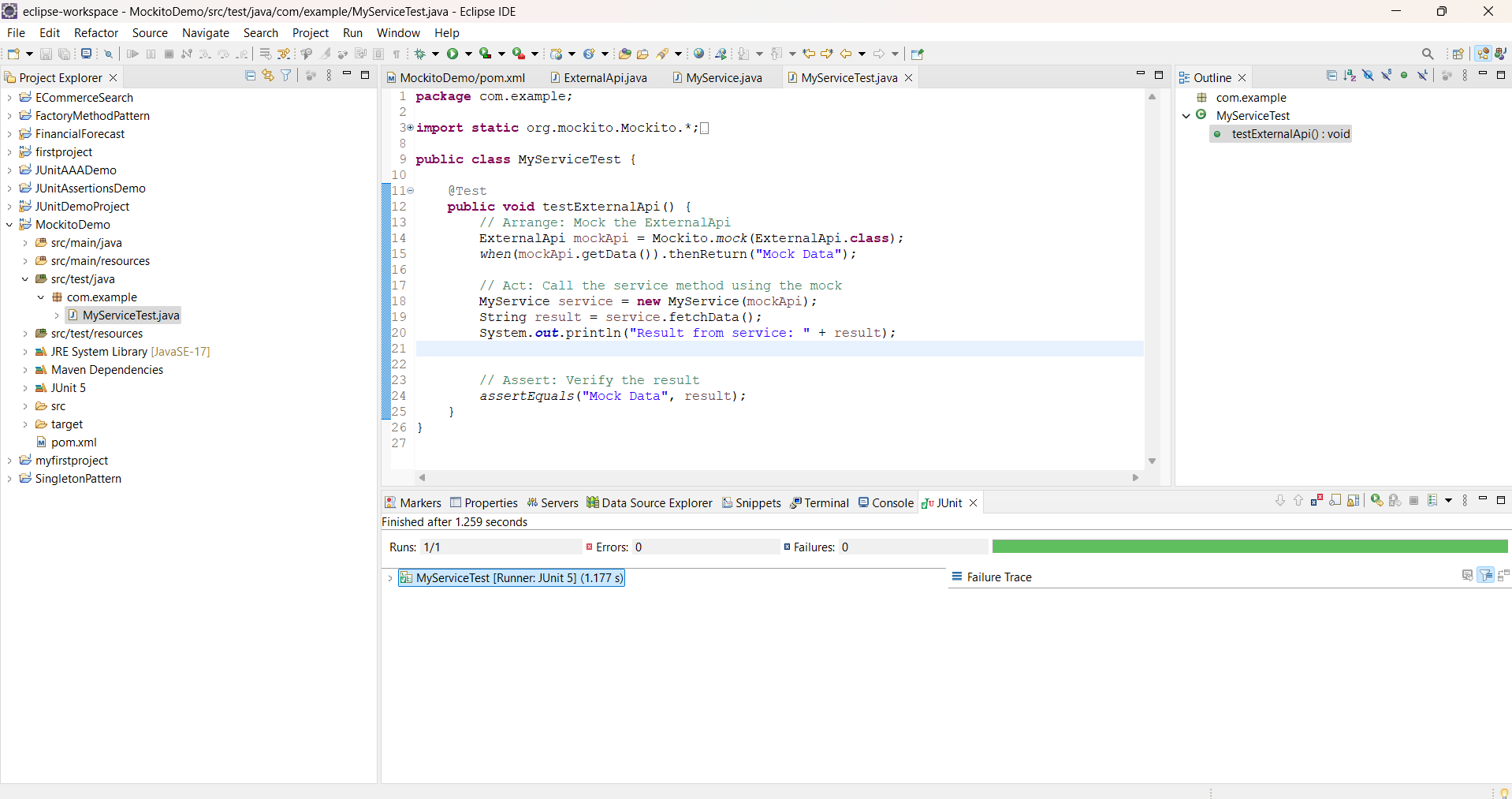
**OUTPUT**

**CONSOLE**

****



**JUNIT**

****



**EXERCISE-02:Verifying Interactions**

Pom.xml

<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>5.11.0</version>

<scope>test</scope>

</dependency>

</dependencies>

ExternalApi.java

package com.example;

public interface ExternalApi {

String getData();

}

MyService.java

package com.example;

public class MyService {

private ExternalApi externalApi;

public MyService(ExternalApi externalApi) {

this.externalApi = externalApi;

}

public String fetchData() {

return externalApi.getData();

}

}

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() {

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

MyService service = new MyService(mockApi);

service.fetchData();

verify(mockApi).getData();

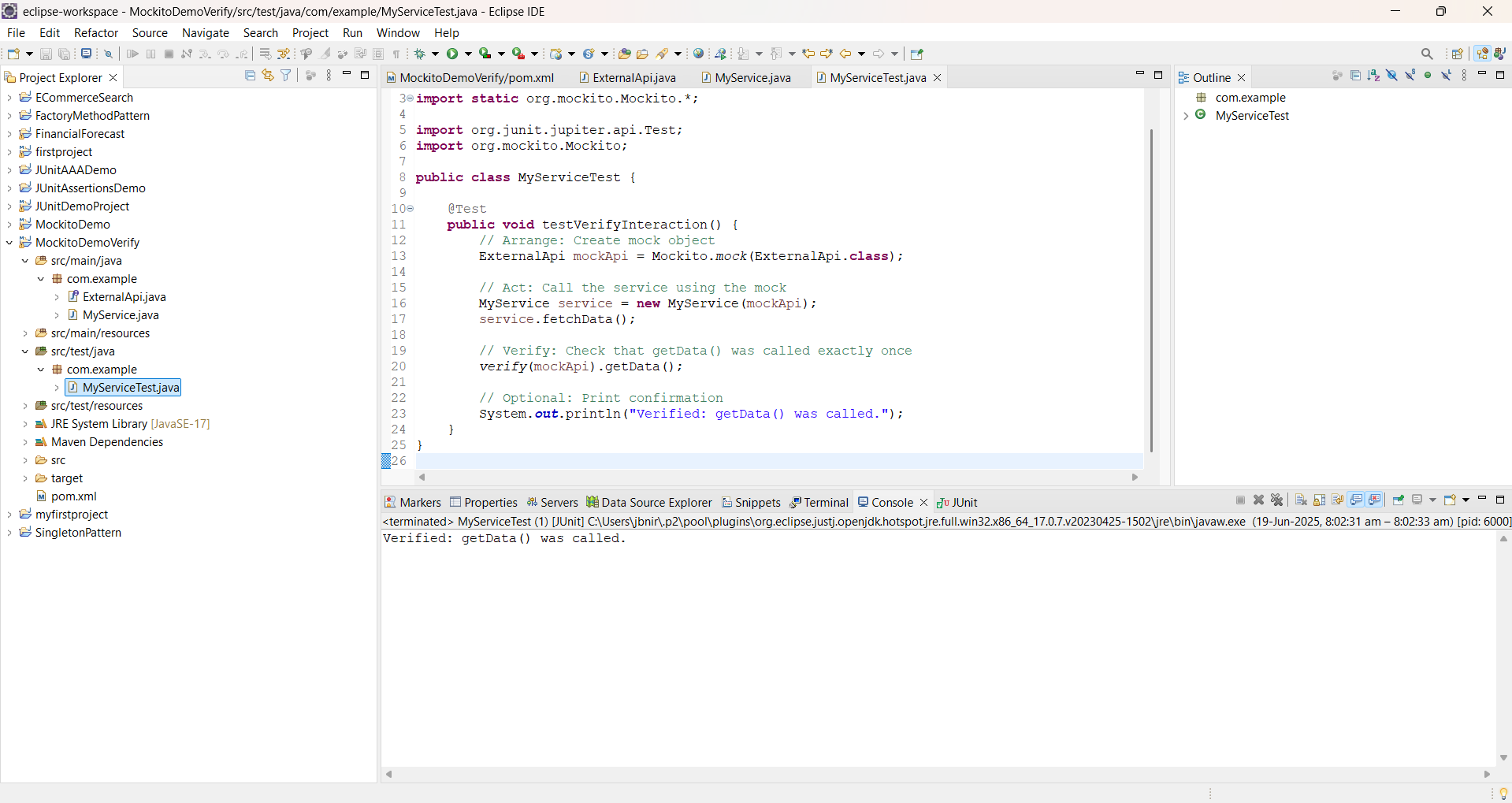
System.out.println("Verified: getData() was called.");

}

}

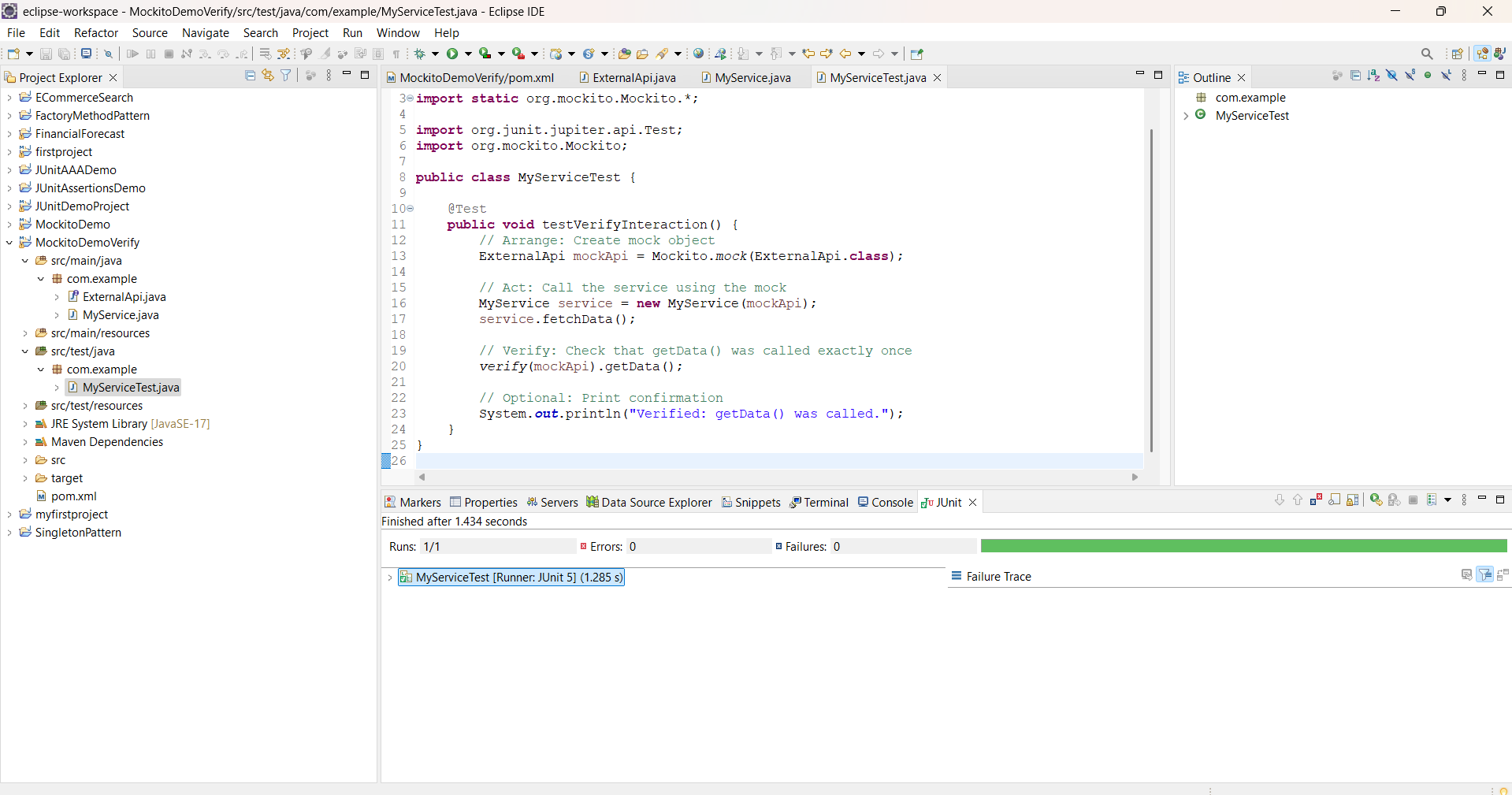
**OUTPUT**

**CONSOLE**

****



**JUNIT**

****



**LOGGING USING SLF4J**

**EXERCISE-01:** **Logging Error Messages and Warning Levels**

Pom.xml



<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>

LoggingExample.java

package com.example.logging;

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");

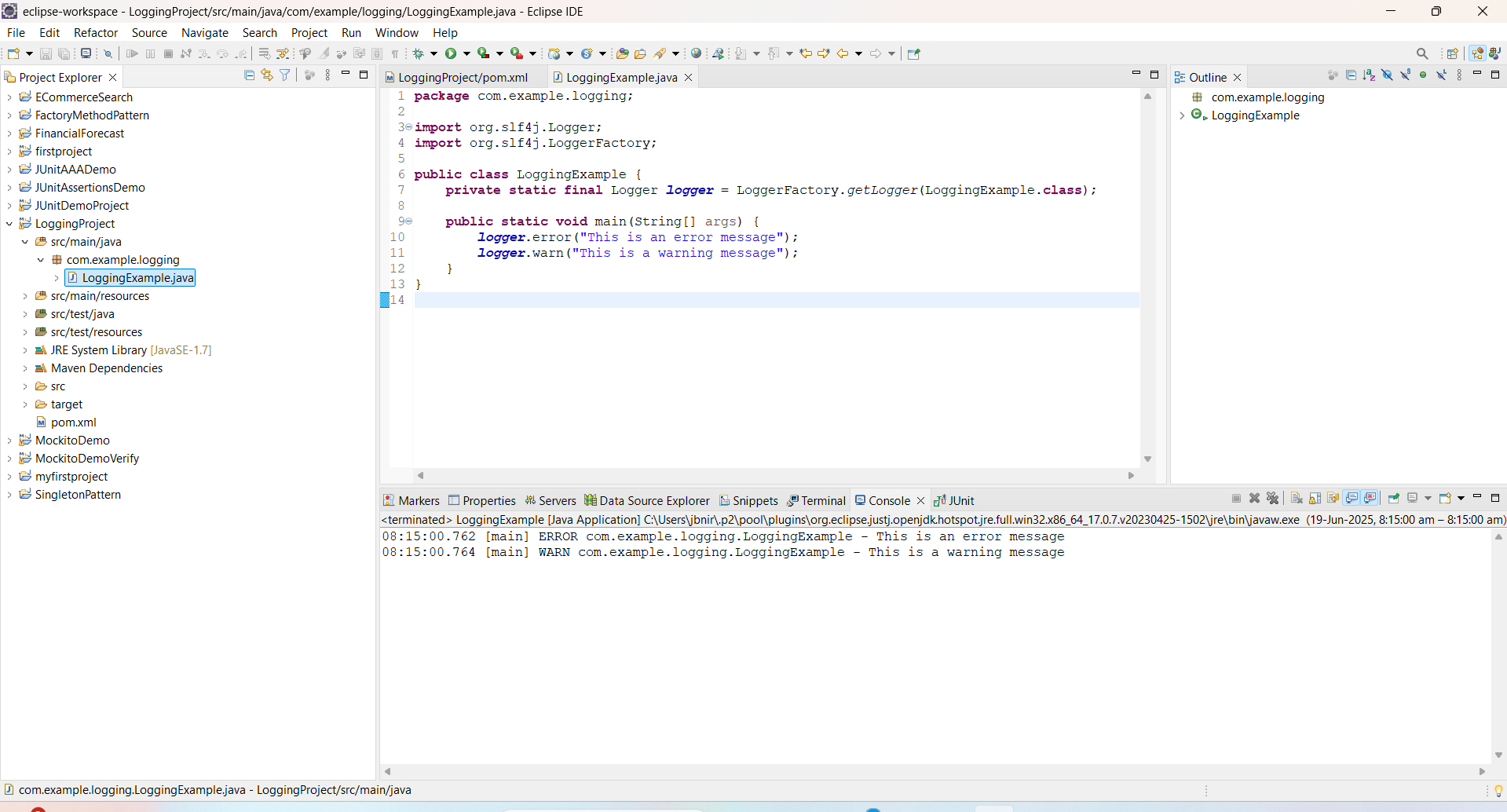
logger.warn("This is a warning message");

}

}

**OUTPUT**

**CONSOLE**

****

