**WEEK – 2**

**Module 4 : Test driven development and Logging framework**

**Exercise 1:**

**CODE:**

**Pom.xml:**

<project>

<modelVersion>4.0.0</modelVersion>

<groupId>mygroupid</groupId>

<artifactId>myartifactid</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>

**Calculator.java:**

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:**

import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class CalculatorTest {

Calculator calc = new Calculator();

@Test

public void testAddition() {

assertEquals(5, calc.add(2, 3));

}

@Test

public void testSubtraction() {

assertEquals(6, calc.subtract(10, 4));

}

}

**Main.java:**

import org.junit.runner.JUnitCore;

import org.junit.runner.Result;

import org.junit.runner.notification.Failure;

public class Main {

public static void main(String[] args) {

Result result = JUnitCore.runClasses(CalculatorTest.class);

for (Failure failure : result.getFailures()) {

System.out.println(failure.toString());

}

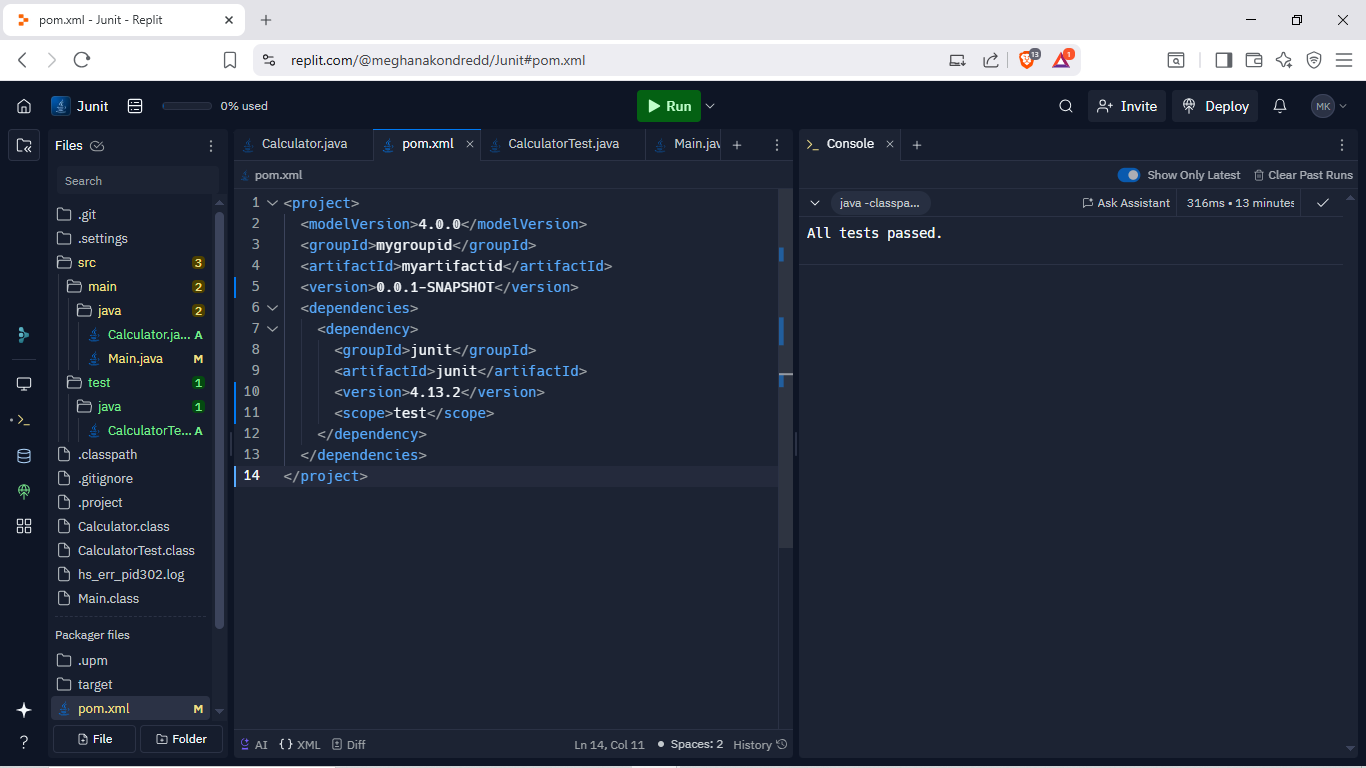
if (result.wasSuccessful()) {

System.out.println("All tests passed.");

}

}

}

**Output:**

**Exercise 3:**

**CODE:**

**AssertionsTest.java:**

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions(){

assertEquals(5, 2 + 3);

assertTrue(5 > 3);

assertFalse(5 < 3);

assertNull(null);

assertNotNull(new Object());

}

}

**Main.java:**

import org.junit.runner.JUnitCore;

import org.junit.runner.Result;

import org.junit.runner.notification.Failure;

public class Main {

public static void main(String[] args) {

Result result = JUnitCore.runClasses(

AssertionsTest.class

);

for (Failure failure : result.getFailures()) {

System.out.println(failure.toString());

}

if (result.wasSuccessful()) {

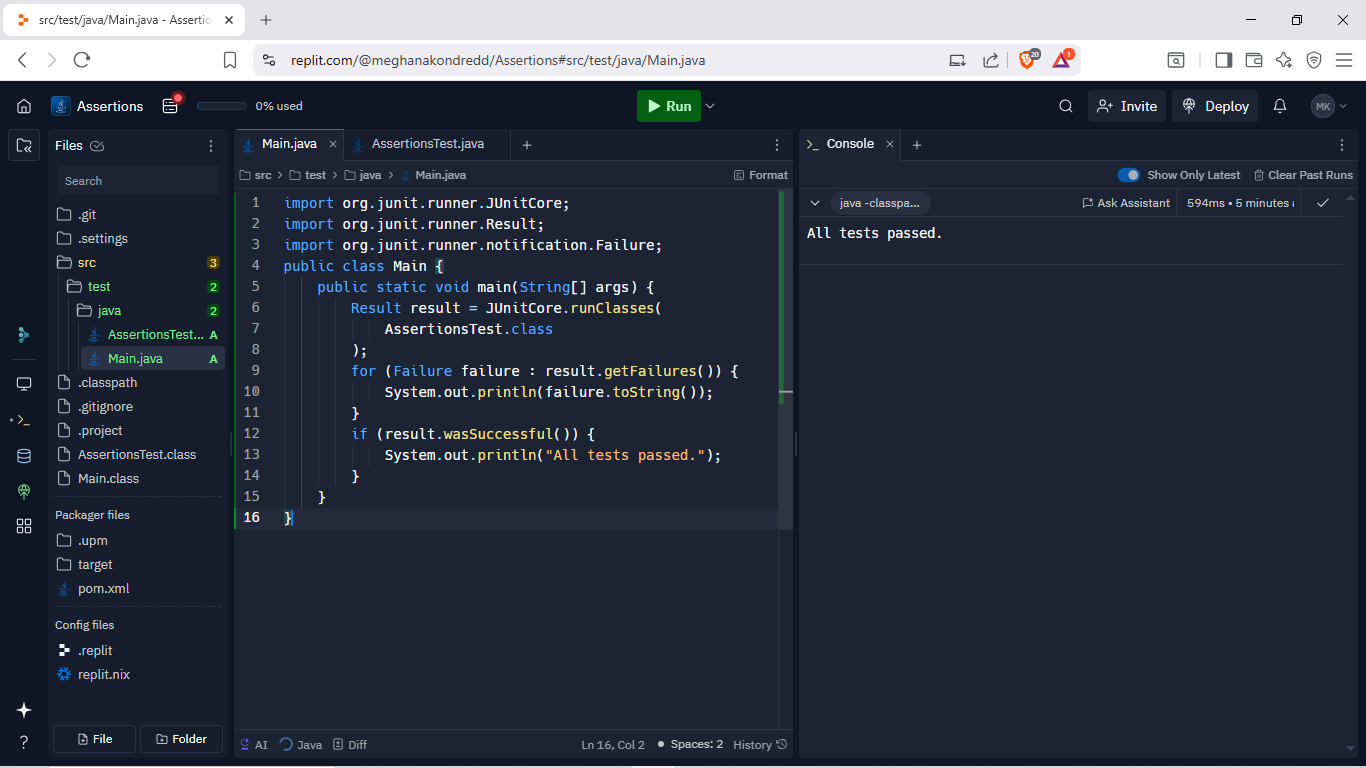
System.out.println("All tests passed.");

}

}

}

**Output:**



**Exercise 4:**

**CODE:**

**Calculator.java:**

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:**

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

private Calculator calculator;

@Before

public void setUp() {

calculator = new Calculator();

System.out.println("Setting up Calculator");

}

@After

public void tearDown() {

calculator = null;

System.out.println("Cleaning up Calculator");

}

@Test

public void testAddition() {

int a = 5, b = 3;

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

assertEquals(8, result);

}

@Test

public void testSubtraction() {

int a = 10, b = 4;

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

assertEquals(6, result);

}

}

import org.junit.runner.JUnitCore;

import org.junit.runner.Result;

import org.junit.runner.notification.Failure;

public class Main {

public static void main(String[] args) {

Result result = JUnitCore.runClasses(CalculatorTest.class);

for (Failure failure : result.getFailures()) {

System.out.println(failure.toString());

}

if (result.wasSuccessful()) {

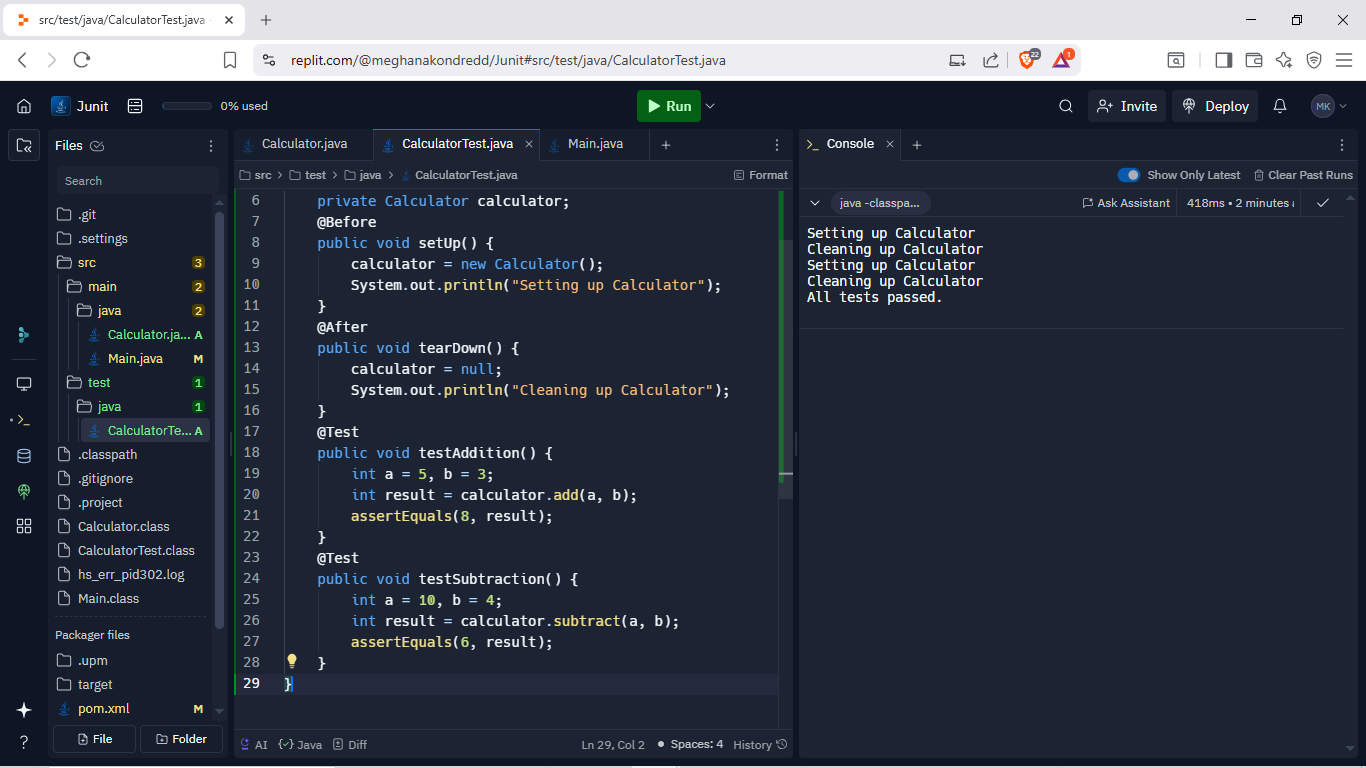
System.out.println("All tests passed.");

}

}

}

**Output:**



**MOCKITO EXERCISES:**

**Exercise 1:** Mocking and Stubbing

**CODE:**

**ExternalApi.java:**

public interface ExternalApi {

String getData();

}

**MyService.java:**

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**MyServiceTest.java:**

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);

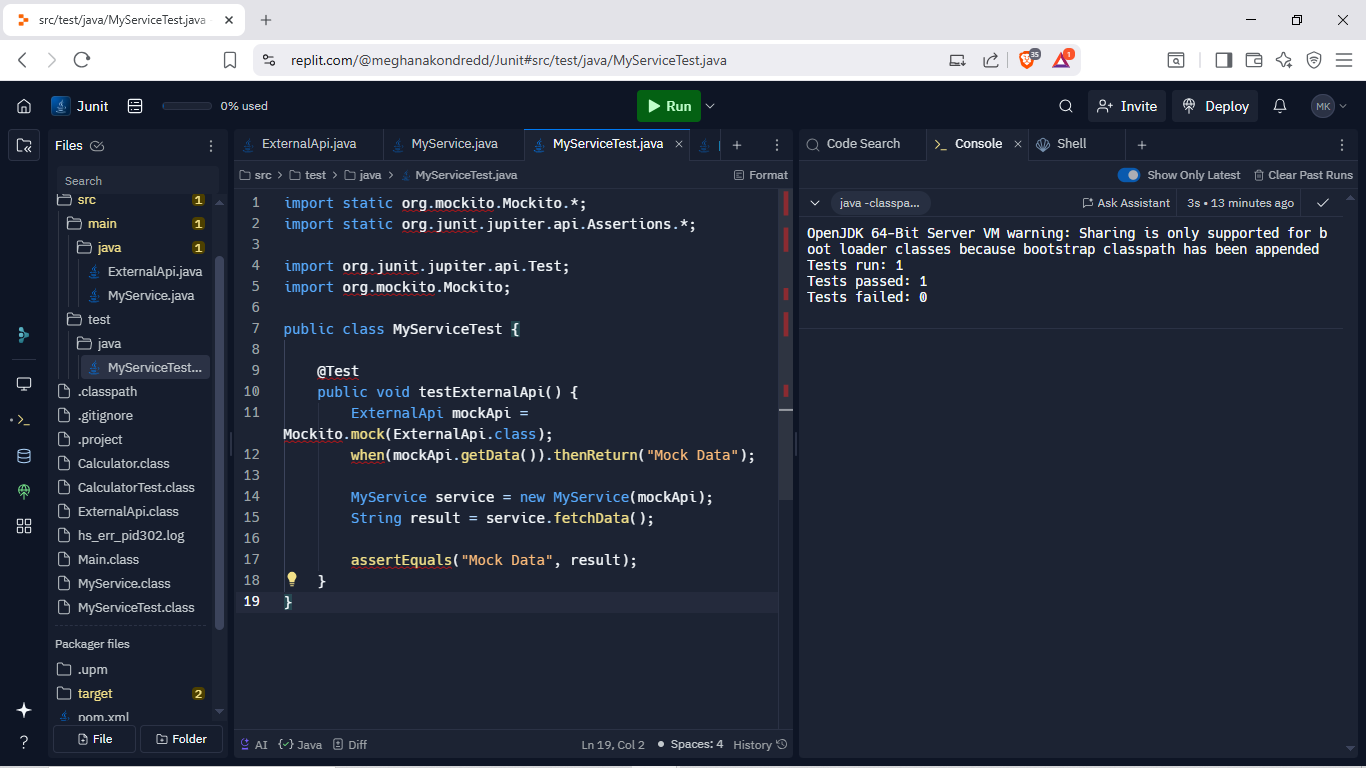
String result = service.fetchData();

assertEquals("Mock Data", result);

}

}

**Output:**



**Exercise 2:** Verifying Interactions

**CODE:**

**ExternalApi.java:**

public interface ExternalApi {

String getData();

}

**MyService.java:**

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**MyServiceTest.java:**

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);

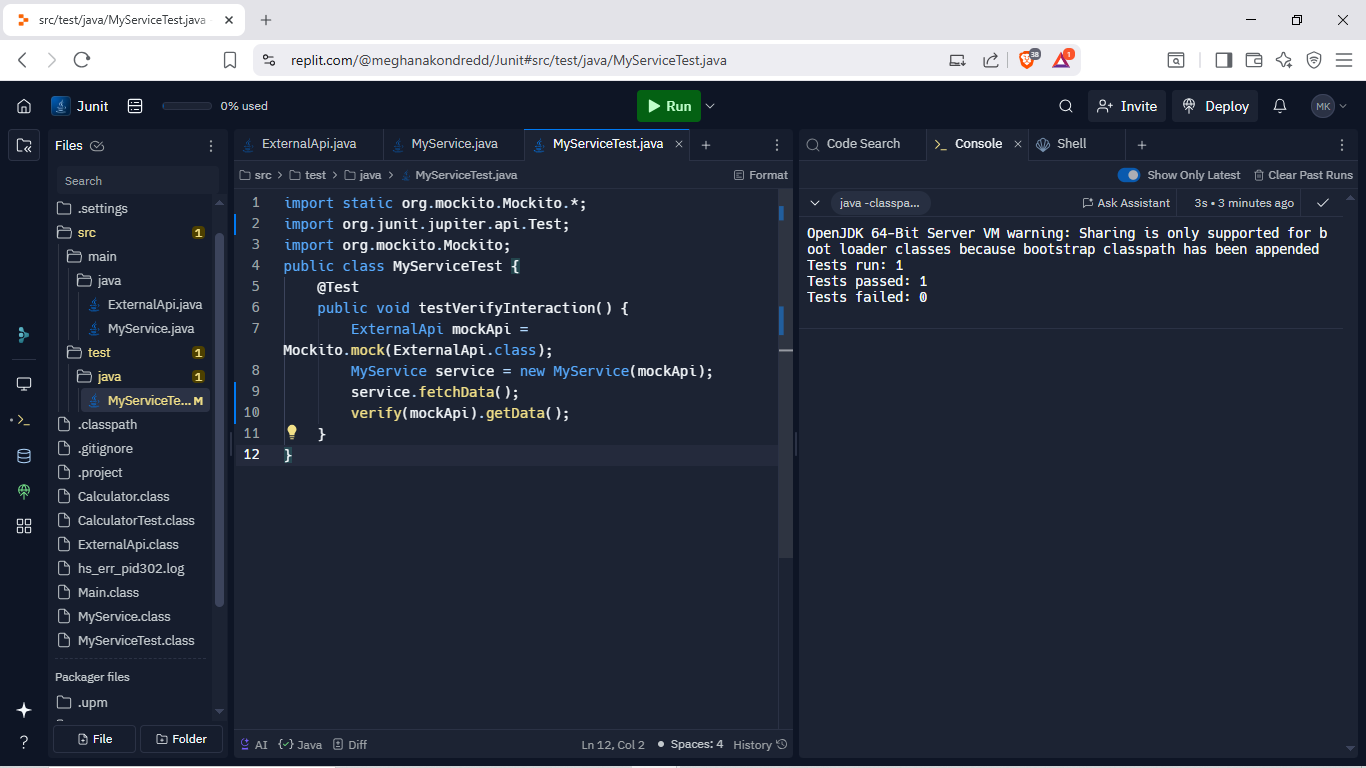
service.fetchData();

verify(mockApi).getData();

}

}

**Output:**



**SL4J Logging Exercises**

**Exercise 1:**

**CODE:**

**LoggingExample.java:**

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");

}

}

**Main.java:**

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class Main {

public static void main(String[] args) {

LoggingExample.main(args);

}

}

**OUTPUT:**

