**JUnit Testing Exercises**

**# pom.xml**

<dependencies>

<!-- https://mvnrepository.com/artifact/junit/junit -->  
<dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.13.1</version>  
 <scope>test</scope>  
</dependency>

</dependencies>

***Exercise 1: Setting Up Junit***

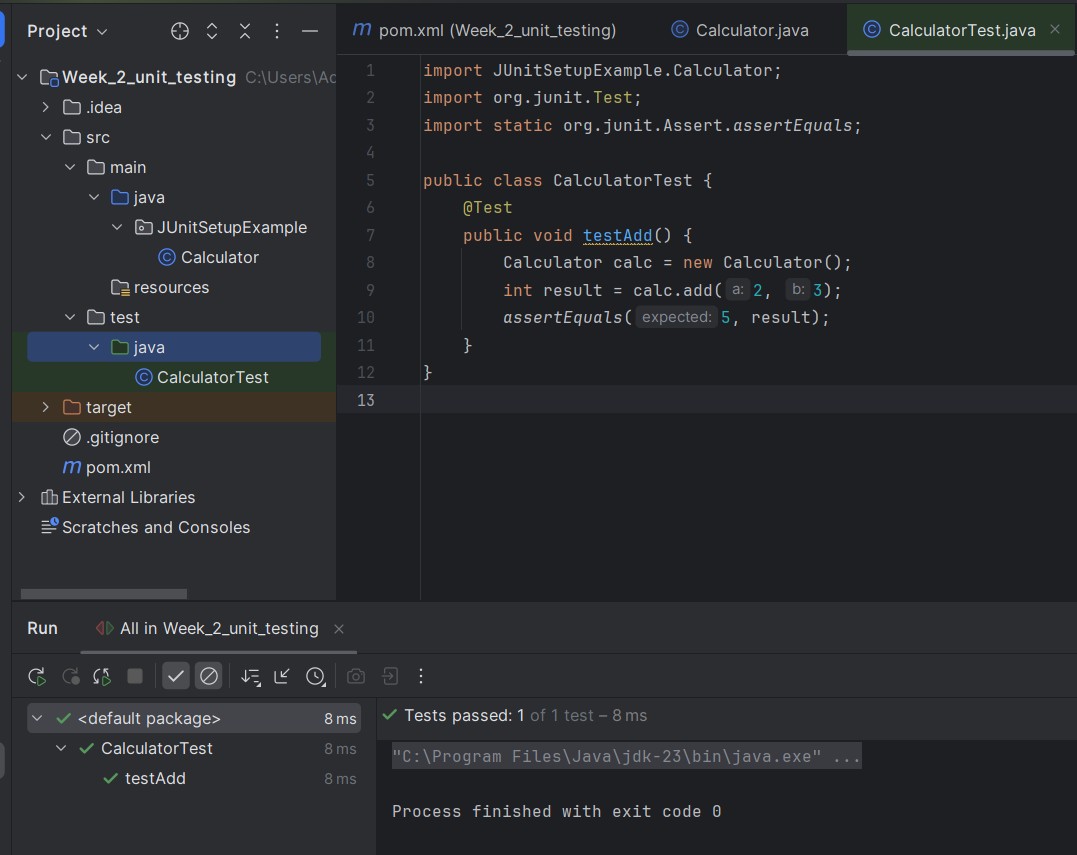
**# Calculator.java**

package JUnitSetupExample;  
  
public class Calculator {  
 public int add(int a, int b) {  
 return a + b;  
 }  
}

**# CalculatorTest.java**

import JUnitSetupExample.Calculator;  
import org.junit.Test;  
import static org.junit.Assert.*assertEquals*;  
  
public class CalculatorTest {  
 @Test  
 public void testAdd() {  
 Calculator calc = new Calculator();  
 int result = calc.add(2, 3);  
 *assertEquals*(5, result);  
 }  
}

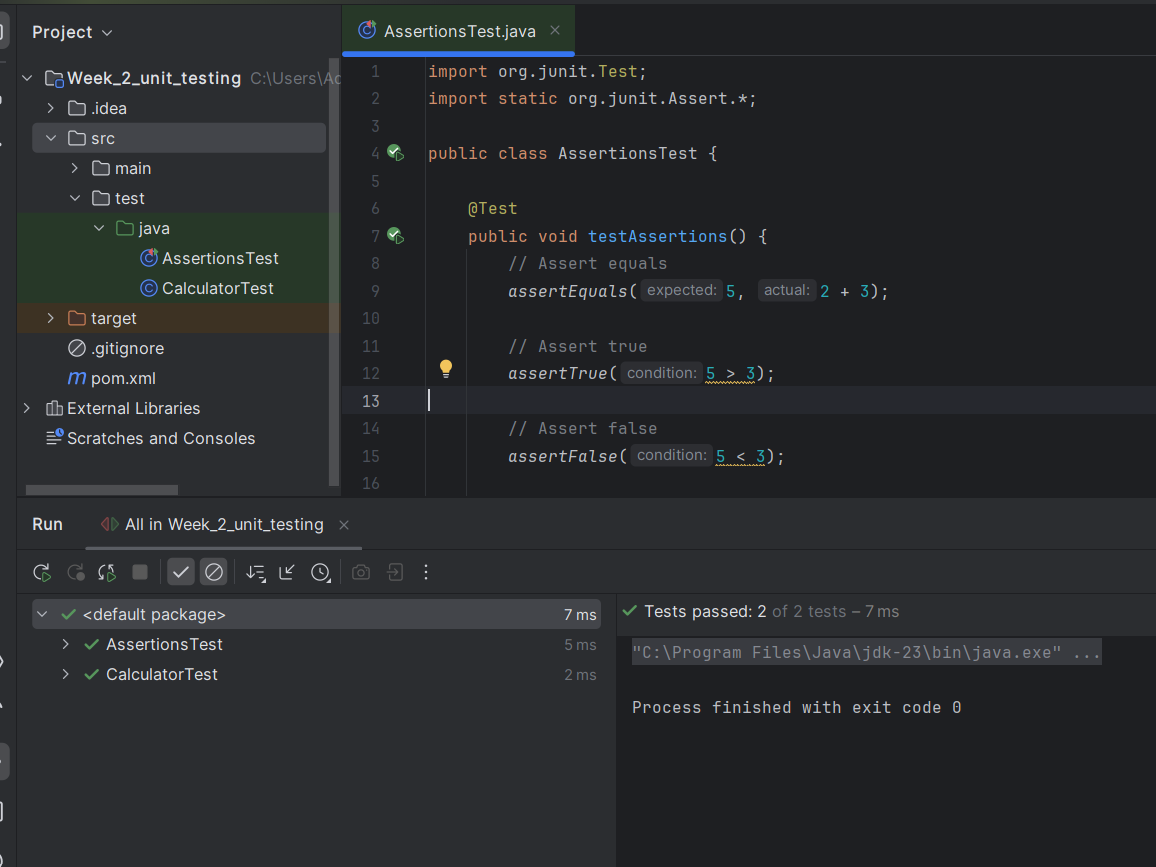
**# Output**



***Exercise 3: Assertions in Junit***

**# AssertionTest.java**

import org.junit.Test;  
import static org.junit.Assert.\*;  
  
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());  
 }  
}

**# Output**

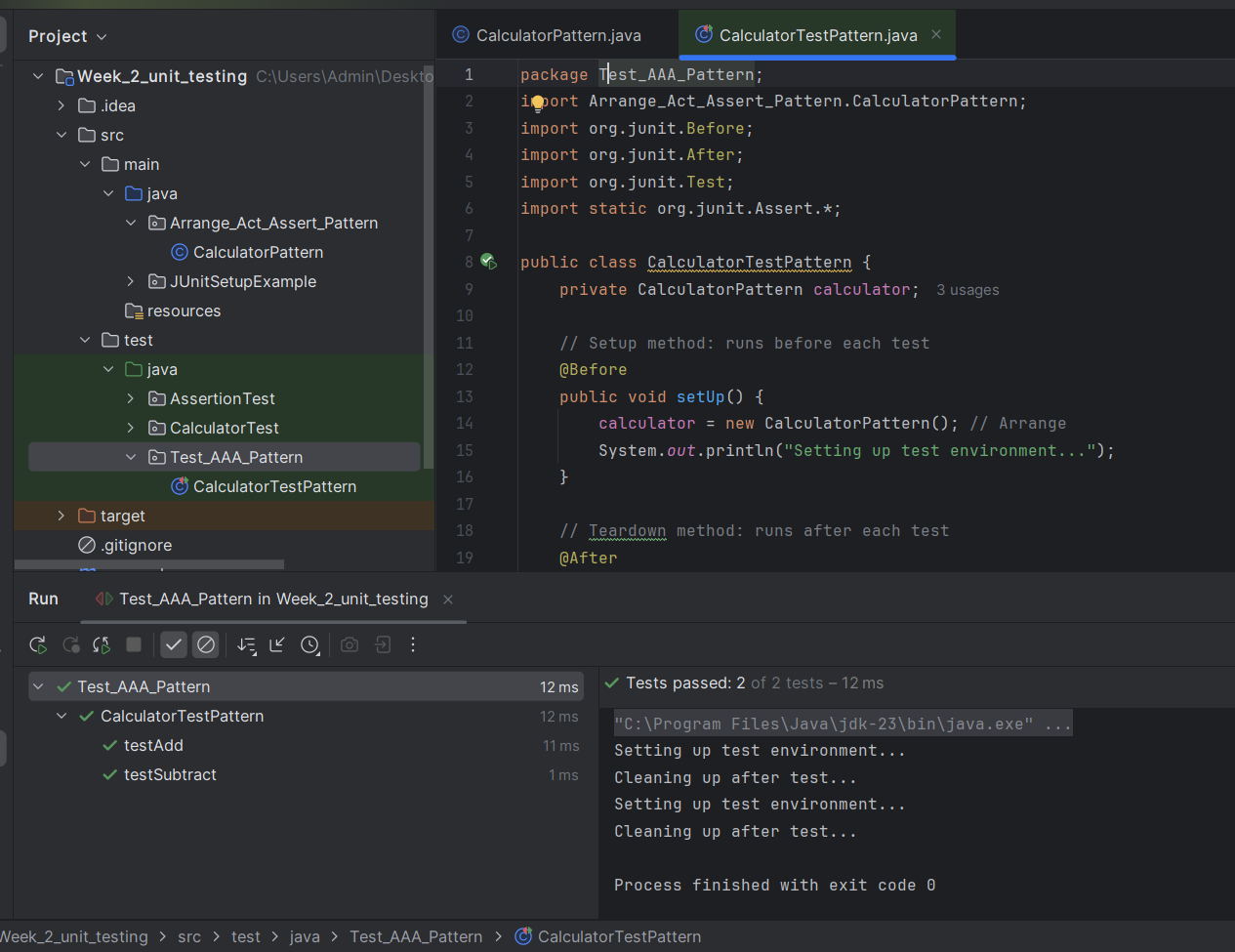
***Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit***

**# CalculatorPattern.java**

package Arrange\_Act\_Assert\_Pattern;  
  
public class CalculatorPattern {  
 public int add(int a, int b) {  
 return a + b;  
 }  
 public int subtract(int a, int b) {  
 return a - b;  
 }  
}

**# CalculatorPatternTest.java**

package Test\_AAA\_Pattern;  
import Arrange\_Act\_Assert\_Pattern.CalculatorPattern;  
import org.junit.Before;  
import org.junit.After;  
import org.junit.Test;  
import static org.junit.Assert.\*;  
  
public class CalculatorTestPattern {  
 private CalculatorPattern calculator;  
  
 // Setup method: runs before each test  
 @Before  
 public void setUp() {  
 calculator = new CalculatorPattern(); // Arrange  
 System.*out*.println("Setting up test environment...");  
 }  
  
 // Teardown method: runs after each test  
 @After  
 public void tearDown() {  
 System.*out*.println("Cleaning up after test...");  
 }  
  
 @Test  
 public void testAdd() {  
 // Act  
 int result = calculator.add(2, 3);  
  
 // Assert  
 *assertEquals*(5, result);  
 }  
  
 @Test  
 public void testSubtract() {  
 // Act  
 int result = calculator.subtract(10, 4);  
  
 // Assert  
 *assertEquals*(6, result);  
 }  
}

**# Output**

**Mockito**

**# pom.xml**

<dependencies>

<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter -->  
<dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter</artifactId>  
 <version>5.13.0</version>  
 <scope>test</scope>  
</dependency>

<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->  
<dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter-api</artifactId>  
 <version>5.13.2</version>  
 <scope>test</scope>  
</dependency>

<!-- https://mvnrepository.com/artifact/org.mockito/mockito-core -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.14.2</version> <!-- Or the latest version -->

<scope>test</scope>

</dependency>

</dependencies>

***Exercise 1: Mocking and Stubbing***

**# ExternalApi.java (interface)**

package Mocking\_nd\_Stubbing;  
  
public interface ExternalApi {  
 String getData();  
}

**# MyService.java**

package Mocking\_nd\_Stubbing;  
  
public class MyService {  
 private final ExternalApi externalApi;  
  
 public MyService(ExternalApi externalApi) {  
 this.externalApi = externalApi;  
 }  
  
 public String fetchData() {  
 return externalApi.getData();  
 }  
}

**# MyServiceTest.java**

import Mocking\_nd\_Stubbing.\*;

import org.junit.jupiter.api.Test;

import static org.mockito.Mockito.\*;

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

public class MyServiceTest {

@Test

public void testMocking() {

ExternalApi api = mock(ExternalApi.class);

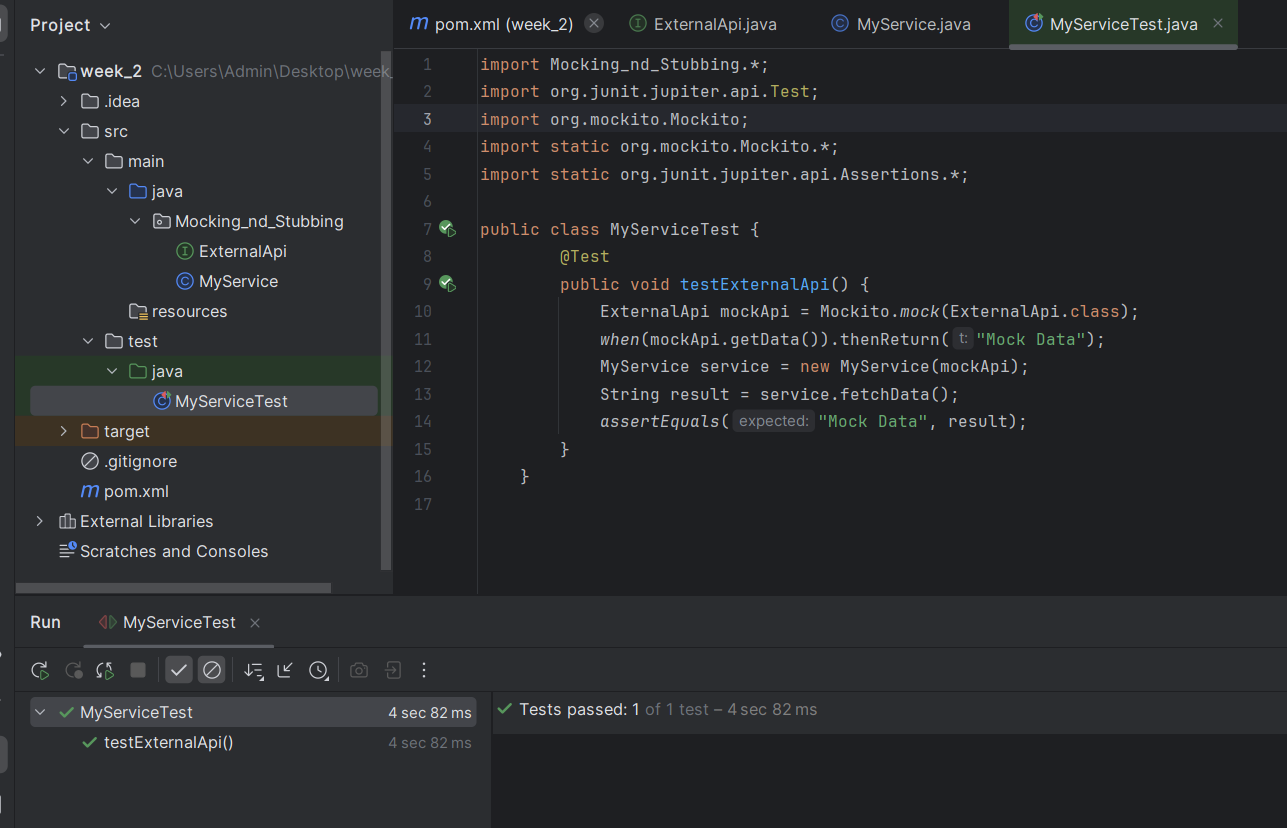
when(api.getData()).thenReturn("Hello");

MyService service = new MyService(api);

assertEquals("Hello", service.fetchData());

}

}

**# Output**

***Exercise 2: Verifying Interactions***

**# ExternalApi.java (interface)**

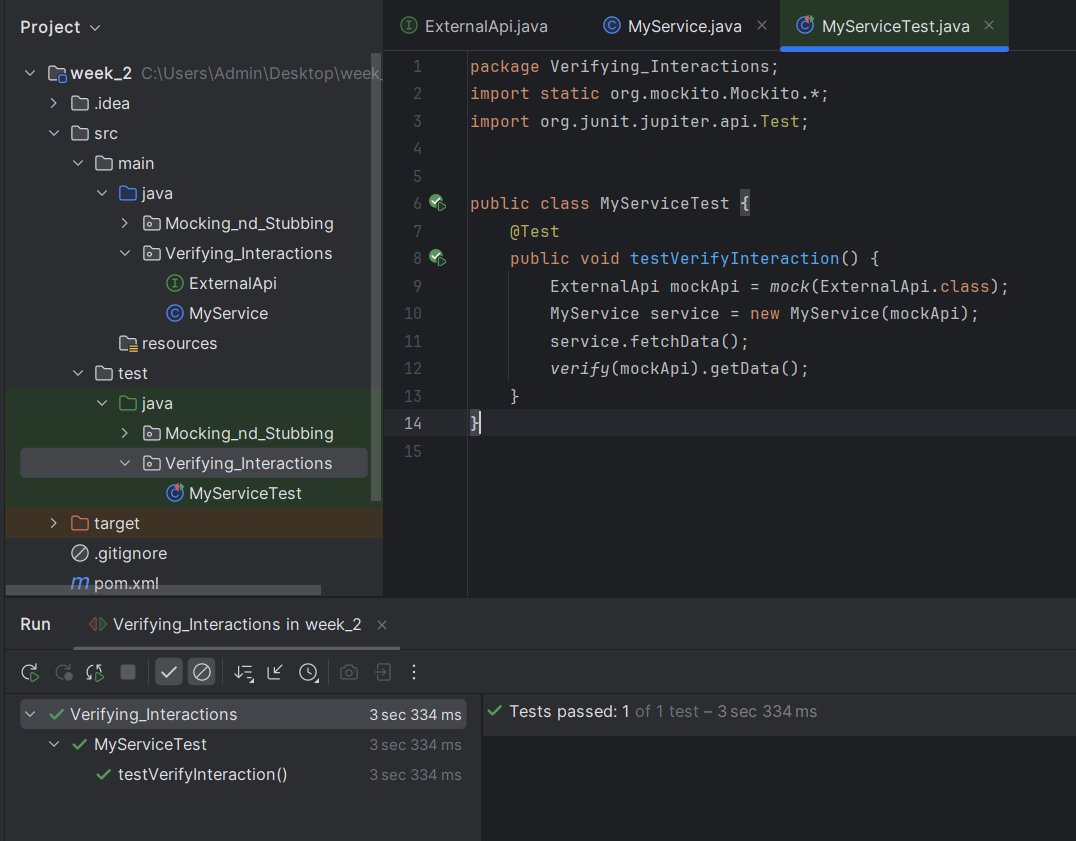
package Verifying\_Interactions;  
  
public interface ExternalApi {  
 String getData();  
}

**# MyService.java**

package Verifying\_Interactions;  
  
public class MyService {  
 private final ExternalApi externalApi;  
  
 public MyService(ExternalApi externalApi) {  
 this.externalApi = externalApi;  
 }  
  
 public String fetchData() {  
 return externalApi.getData();  
 }  
}

**# MyServiceTest.java**

package Verifying\_Interactions;  
import static org.mockito.Mockito.\*;  
import org.junit.jupiter.api.Test;  
  
  
public class MyServiceTest {  
 @Test  
 public void testVerifyInteraction() {  
 ExternalApi mockApi = *mock*(ExternalApi.class);  
 MyService service = new MyService(mockApi);  
 service.fetchData();  
 *verify*(mockApi).getData();  
 }  
}

**# Output**

**Logging using SLF4J**

<dependencies>

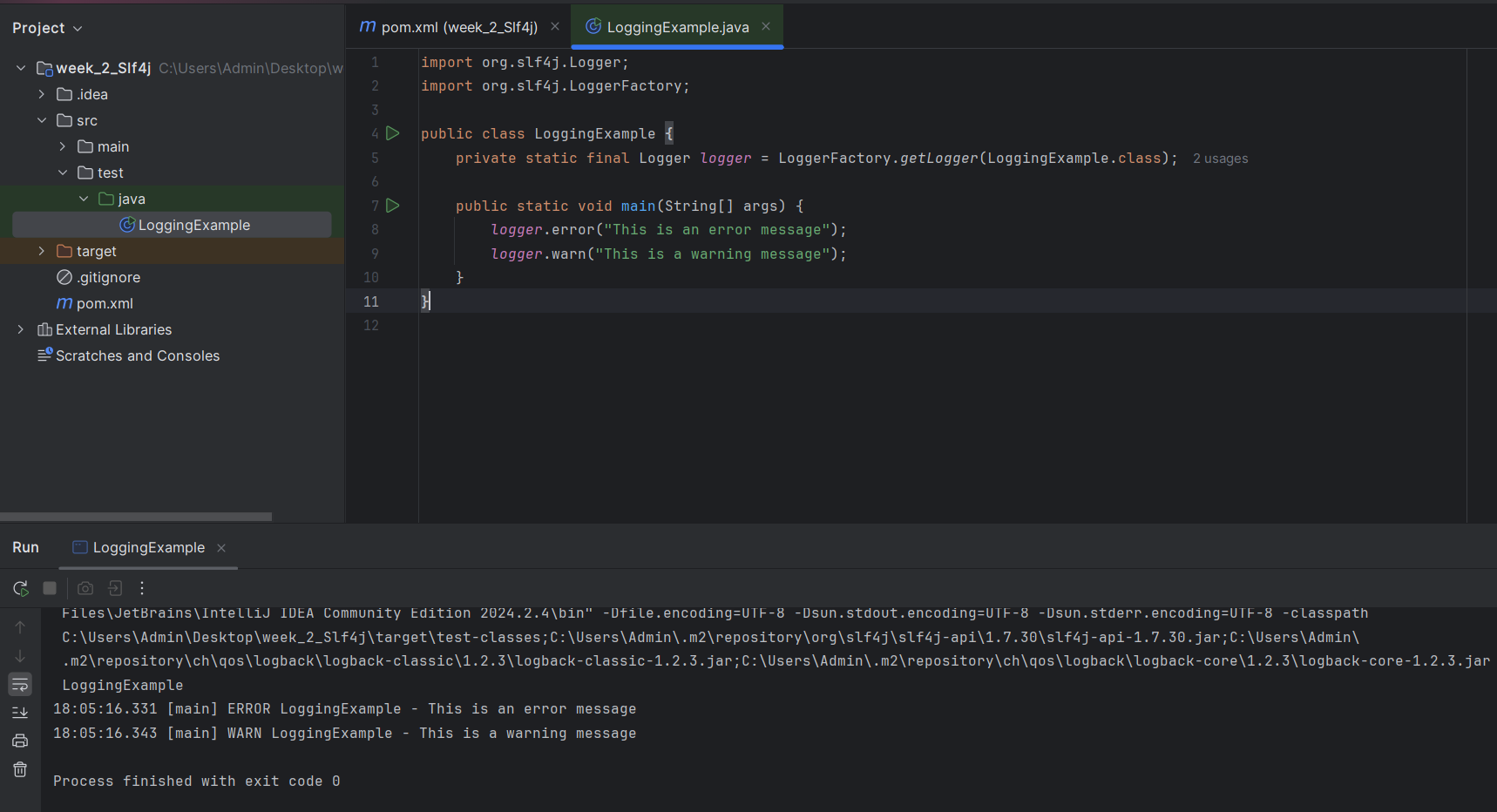
<dependency>  
 <groupId>org.slf4j</groupId>  
 <artifactId>slf4j-api</artifactId>  
 <version>1.7.30</version>  
 </dependency>  
  
 <dependency>  
 <groupId>ch.qos.logback</groupId>  
 <artifactId>logback-classic</artifactId>  
 <version>1.2.3</version>  
 </dependency>  
  
</dependencies>

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

**# LoggerExample.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");  
 }  
}

**# Output**

****