**Exercise 1: Mocking and Stubbing**

**The Interface (ExternalApi)**

// ExternalApi.java

public interface ExternalApi {

String getData(); // method to fetch data from an external source

}

**The Service Class (MyService)**

// MyService.java

public class MyService {

private ExternalApi externalApi;

// Constructor injection

public MyService(ExternalApi externalApi) {

this.externalApi = externalApi;

}

public String fetchData() {

// Calls the external API

return externalApi.getData();

}

}

**Add Mockito and JUnit Dependencies**

**pom.xml (for Maven)**

<dependencies>

<!-- JUnit 5 -->

<dependency>

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

<artifactId>junit-jupiter</artifactId>

<version>5.9.3</version>

<scope>test</scope>

</dependency>

<!-- Mockito -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

</dependencies>

**Test Class Using Mockito**

// 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() {

// Step 1: Create a mock object of ExternalApi

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

// Step 2: Stub the getData() method to return a predefined value

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

// Step 3: Inject the mock into MyService

MyService service = new MyService(mockApi);

// Step 4: Call the method and assert the result

String result = service.fetchData();

assertEquals("Mock Data", result);

}

}

**Output:**

