**Name: Amirtha A  
superset id : 6394120**

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

You need to test a service that depends on an external API. Use Mockito to mock the external API and stub its methods.

**Code:**

**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 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("Fetched data: " + result);

*assertEquals*("Mock Data", result);

}

}

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.

**Exercise 2: Verifying Interactions Scenario:**

You need to ensure that a method is called with specific arguments.

**MyServiceTest1.java:**  
  
package com.example;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest1 {

@Test

public void testVerifyInteraction() {

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

MyService service = new MyService(mockApi);

service.fetchData();

verify(mockApi).getData();

System.out.println("Interaction verified!");

}

}

**Output:**

A screenshot of a computer

AI-generated content may be incorrect.