**WEEK-2:**

**Mockito Hands-On Exercises**

**Exercise 1:** Mocking and Stubbing

Program:

**ExternalApi.java**

public interface ExternalApi {

String getData();

}

**MyServices.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.junit.jupiter.api.Assertions.assertEquals;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testExternalApi() {

// Step 1: Mock the ExternalApi

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

// Step 2: Stub the getData method

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

// Step 3: Inject the mock into the service

MyService service = new MyService(mockApi);

// Step 4: Call the method and assert

String result = service.fetchData();

assertEquals("Mock Data", result);

}

}

OUTPUT:

✔ testExternalApi() [MyServiceTest]

Tests run: 1, Failures: 0

**Exercise 2: Verifying Interactions**

**Program:**

import static org.mockito.Mockito.\*;

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

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

// Define the external API interface

interface ExternalApi {

String getData();

}

// Define the service class that uses ExternalApi

class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData(); // this should be verified

}

}

// Unit test using Mockito

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

// Step 1: Create a mock of ExternalApi

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

// Step 2: Create service with the mock

MyService service = new MyService(mockApi);

// Step 3: Call the method (Act)

service.fetchData();

// Step 4: Verify the interaction (Assert)

verify(mockApi).getData(); // Verifies that getData() was called

}

**Output:**

}