# Exercise 2: Verifying Interactions

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

**ExternalApi.java:**

package com.example;

public interface ExternalApi {

String getData();

String processData(String input);

}

**MyService.java:**

private final ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

public String processUserData(String userId) {

return api.processData(userId);

}

}

**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(); }

@Test

public void testVerifyWithArguments() {

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

when(mockApi.processData(anyString())).thenReturn("Processed");

MyService service = new MyService(mockApi);

service.processUserData("test123");

// Verify processData was called with exactly "test123"

verify(mockApi).processData(eq("test123"));

}

// Verification of call count

@Test

public void testVerifyCallCount() {

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

MyService service = new MyService(mockApi);

service.fetchData();

service.fetchData();

// Verify getData was called exactly 2 times

verify(mockApi, times(2)).getData();

}

// Verification with argument matchers

@Test

public void testVerifyWithArgumentMatchers() {

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

MyService service = new MyService(mockApi);

service.processUserData("user\_42");

// Verify processData was called with any string starting with "user\_"

verify(mockApi).processData(startsWith("user\_"));

}

}