Name:- Vidya CS Track: JAVA FSE

# **Mockito Hands-On Exercises**

## **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.

#### Outcome:

#### Step1: Create a mock object for the external API

```
eciipisezavarrograms - iviockitopemo/sic/main/java/com/exampi/externalApi.java - eciipse ibe
File Edit Source Refactor Navigate Search Project Run Window Help
E S 7 8 □ □ MockitoDemo/pom.xml
                                                         ☑ ExternalApi.java X
☑ MyService.java
☑ MyServiceTest.java
Project Explorer ×
                                       package com.exampl;

✓ 

MockitoDemo

  ✓ # src/main/iava
    4 public interface ExternalApi {
                                            String getData();
     > 🗗 ExternalApi.java
      > MyService.java
  > 🕮 src/main/resources
  >  MyServiceTest.java
  >  src/test/resources
  > Mark JRE System Library [JavaSE-1.8]
  > Maven Dependencies
```

## Step2: Stub the methods to return predefined values

```
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer X
                MockitoDemo/pom.xml
                                                                 MvService.iava
                                                    ExternalApi.java
                                   1 package com.exampl;

∨ ₩ MockitoDemo

 public class MyService {

✓ 

⊕ com.exampl

                                   4
                                        private ExternalApi api;
    > 🗗 ExternalApi.java
                                   5
                                        public MyService(ExternalApi api) {
                                   6⊖
     7
                                           this.api = api;
 > # src/main/resources
                                   8
 10⊖
                                        public String fetchData() {
   11
                                           return api.getData();
     12
                                  13 }
 >  src/test/resources
 > A JRE System Library [JavaSE-1.8]
 > Maven Dependencies
```

## Step3: Write a test case that uses the mock object

```
☑ ExternalApi.java
☑ MyService.java
☑ MyServiceTest.java
MockitoDemo/pom.xml
 1 package com.exampl;
 3 import org.junit.jupiter.api.Test;
 import static org.junit.jupiter.api.Assertions.*;
import static org.mockito.Mockito.*;
    public class MyServiceTest {
 8
 9
         @Test
         public void testExternalApi() {
    // Step 1: Create a mock of ExternalApi
10⊝
11
             ExternalApi mockApi = mock(ExternalApi.class);
12
13
14
             // Step 2: Stub the getData() method to return "Mock Data"
             when(mockApi.getData()).thenReturn("Mock Data");
15
16
17
              // Step 3: Inject mock into MyService
18
             MyService service = new MyService(mockApi);
20
              // Step 4: Call the method and assert the result
21
             String result = service.fetchData();
             assertEquals("Mock Data", result);
22
23
         }
24
    }
```

#### **Output :- Test pass**

```
Finished after 2.065 seconds
                                                           3⊖ import org.junit.jupiter.api.Test;
                                                              import static org.junit.jupiter.api.Assertions.*;
import static org.mockito.Mockito.*;
 Runs: 1/1
                                   ■ Failures: 0
                                                             public class MyServiceTest {
                                                          8
 > MyServiceTest [Runner: JUnit 5] (1.924 s)
                                                                   public void testExternalApi() {
    // Step 1: Create a mock of ExternalApi
                                                         10⊝
                                                         11
                                                                        ExternalApi mockApi = mock(ExternalApi.class);
                                                         13
                                                                        // Step 2: Stub the getData() method to return "Mock Data"
when(mockApi.getData()).thenReturn("Mock Data");
                                                         14
                                                         16
                                                         17
                                                                        // Step 3: Inject mock into MyService
                                                         18
19
                                                                        MyService service = new MyService(mockApi);
                                                         20
                                                                         // Step 4: Call the method and assert the result
                                                         21
                                                                        String result = service.fetchData();
                                                                        assertEquals("Mock Data", result);
                                                         23
```

# **Exercise 2: Verifying Interactions**

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

1.Create a mock object.

Code :- ExternalApi mockApi = Mockito.mock(ExternalApi.class);

2.Call the method with specific arguments.

Code:- MyService service = new MyService(mockApi); service.fetchData();

3. Verify the interaction.

Code:- verify(mockApi).getData();

## Output:- Test successful

```
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer  

▼

**JUnit ×*
                                        MockitoDemo/...
                                                          ☑ ExternalApi.j...
☑ MyService.java
☑ MyServiceTest...

☑ MyServiceTest... ×
                                          1 package com.exampl;
          2⊖ import static org.mockito.Mockito.*;
inished after 1.363 seconds
                                          3 import org.junit.jupiter.api.Test;
                                          4 import org.mockito.Mockito;
Runs: 1/1

■ Errors: 0

                         ■ Failures: 0
                                          6 public class MyServiceTestVerifyInteraction{
> MyServiceTestVerifyInteraction [Runner: JUnit 5] (1.173
                                                public void testVerifyInteraction() {
                                         10
                                                    // Step 1: Create a mock object
                                                    ExternalApi mockApi = Mockito.mock(ExternalApi.class);
                                         12
13
14
15
16
17
18
19
                                                    // Step 2: Inject the mock into MyService
                                                    MyService service = new MyService(mockApi);
                                                    // Step 2: Call the method
                                                    service.fetchData();
                                                    // Step 3: Verify that getData() was called on mockApi
                                                    verify(mockApi).getData();
                                         22 }
```