**Week 2 Mockito Handson**

**Name- Adi Jain**

**Superset ID- 6390400**

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.

**Steps:**

1. Create a mock object for the external API.

2. Stub the methods to return predefined values.

3. Write a test case that uses the mock object.

**Code**

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <groupId>org.example</groupId>  
 <artifactId>mockitodemo</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <dependencies>  
 <!-- JUnit 5 -->  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter</artifactId>  
 <version>5.10.2</version>  
 <scope>test</scope>  
 </dependency>  
  
 <!-- Mockito -->  
 <dependency>  
 <groupId>org.mockito</groupId>  
 <artifactId>mockito-core</artifactId>  
 <version>5.12.0</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
  
 <properties>  
 <maven.compiler.source>21</maven.compiler.source>  
 <maven.compiler.target>21</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
  
</project>

**Pom.xml**

public interface externalapi {  
 String getData();  
}

**externalapi.java**

package org.example;  
  
public class Main {  
 public static void main(String[] args) {  
 System.*out*.printf("Hello and welcome!");  
 for (int i = 1; i <= 5; i++) {  
   
 System.*out*.println("i = " + i);  
 }  
 }  
}

**Main.java**

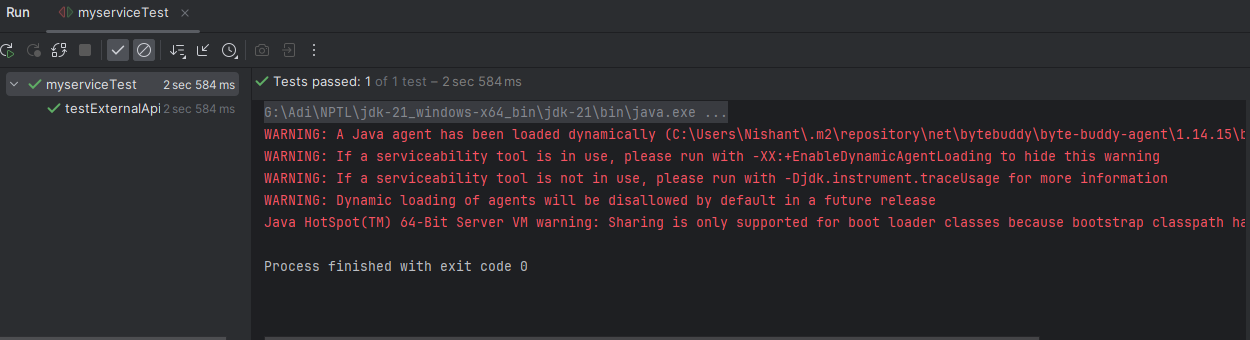
public class myservice {  
 private externalapi api;  
  
 public myservice(externalapi api) {  
 this.api = api;  
 }  
  
 public String fetchData() {  
 return api.getData();  
 }  
}

**myservice.java**

import org.junit.jupiter.api.Test;  
import org.mockito.Mockito;  
  
import static org.mockito.Mockito.\*;  
import static org.junit.jupiter.api.Assertions.\*;  
  
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();  
 *assertEquals*("Mock Data", result);  
 }  
}

**myserviceTest.java**

**Output**



Exercise 2: Verifying Interactions

**Scenario:**

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

**Steps:**

1. Create a mock object.

2. Call the method with specific arguments.

3. Verify the interaction.

**Code**

import org.junit.jupiter.api.Test;  
import org.mockito.Mockito;  
  
import static org.mockito.Mockito.\*;  
import static org.junit.jupiter.api.Assertions.\*;  
  
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();  
  
 *assertEquals*("Mock Data", result);  
 }  
  
 @Test  
 public void testVerifyInteraction() {  
   
 externalapi mockApi = Mockito.*mock*(externalapi.class);  
  
 myservice service = new myservice(mockApi);  
  
 service.fetchData();  
  
 *verify*(mockApi).getData();  
 }  
}

**myserviceTest.java**

**Output**

