**Exercise1: Mocking and Stubbing with Mockito**

**Objective:** To test a class that depends on an external API by using Mockito to mock the dependency and stub method behavior.

Required Maven Dependencies:

Add the below dependencies to the existing pom.xml

|  |
| --- |
| <!-- JUnit Jupiter -->  <dependency>  <groupId>org.junit.jupiter</groupId>  <artifactId>junit-jupiter</artifactId>  <version>5.10.0</version>  <scope>test</scope>  </dependency>  <!-- Mockito -->  <dependency>  <groupId>org.mockito</groupId>  <artifactId>mockito-core</artifactId>  <version>5.11.0</version>  <scope>test</scope>  </dependency> |

Mocking: creating fake objects that mimics real objects behavior.

Stubbing: Pre-defining return values for specific method calls on a mock.

Construction of Source Code:

1. External API Interface:

ExternalApi.java

|  |
| --- |
| package harshita;  public interface ExternalApi {  String getData();  } |

1. Service class to be tested:

MyService.java:

|  |
| --- |
| package harshita;  public class MyService {  private ExternalApi externalApi;  public MyService (ExternalApi externalApi) {  this.externalApi = externalApi;  }  public String fetchData () {  return externalApi.getData();  }  } |

1. Unit Test with Mock and Stub:

MyServiceTest.java:

|  |
| --- |
| package harshita;  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() {  // Step 1: Create a mock of ExternalApi  ExternalApi mockApi = Mockito.mock (ExternalApi.class);  // Step 2: Stub the method to return a predefined value  When (mockApi.getData()).thenReturn ("Mock Data");  // Step 3: Inject the mock into the service  MyService service = new MyService(mockApi);  // Step 4: Act and Assert  String result = service.fetchData();  assertEquals("Mock Data", result);  }  } |

Expected output:

