**WEEK – 2 TDD USING JUNIT5 AND MOCKITO**

**EXERCISE 1:** Mocking and Stubbing

**DEPENDENCY:**

<dependencies>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.11.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

This dependency setup enables **modern unit testing with JUnit 5 and Mockito**, while also supporting legacy **JUnit 4** test cases — giving you full flexibility for writing clean, isolated, and maintainable tests.

**EXTERNALAPI.JAVA**

package org.example;

public interface ExternalApi {

String getData();

}

**MyService.java**

package org.example;public class MyService { private final ExternalApi externalApi; public MyService(ExternalApi externalApi) { this.externalApi = externalApi; } public String fetchData() { return externalApi.getData(); }}

This class MyService depends on ExternalApi and uses **constructor injection** to receive it.  
The fetchData() method simply delegates the call to externalApi.getData() and returns the result.

**MYSERVICETEST.JAVA**

package org.example;import org.junit.jupiter.api.Test;import static org.junit.jupiter.api.Assertions.assertEquals;import static org.mockito.Mockito.\*;public class MyServiceTest { @Test public void testExternalApi() { ExternalApi mockApi = mock(ExternalApi.class); when(mockApi.getData()).thenReturn("Mock Data"); MyService service = new MyService(mockApi); String result = service.fetchData(); assertEquals("Mock Data", result); }}

This test class verifies that MyService correctly calls ExternalApi and returns the expected data.  
It uses **Mockito** to mock the API and **JUnit** to assert the result from fetchData().

**OUTPUT:**

BUILD SUCCESS

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0