**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.

**ExternalApi.java**

package com.example;

public interface ExternalApi {

String getData();

}

**MyService.java**

package com.example;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**MyServiceTest.java**

package com.example;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class MyServiceTest {

@Test

public void testExternalApi() {

ExternalApi mockApi = mock(ExternalApi.class);

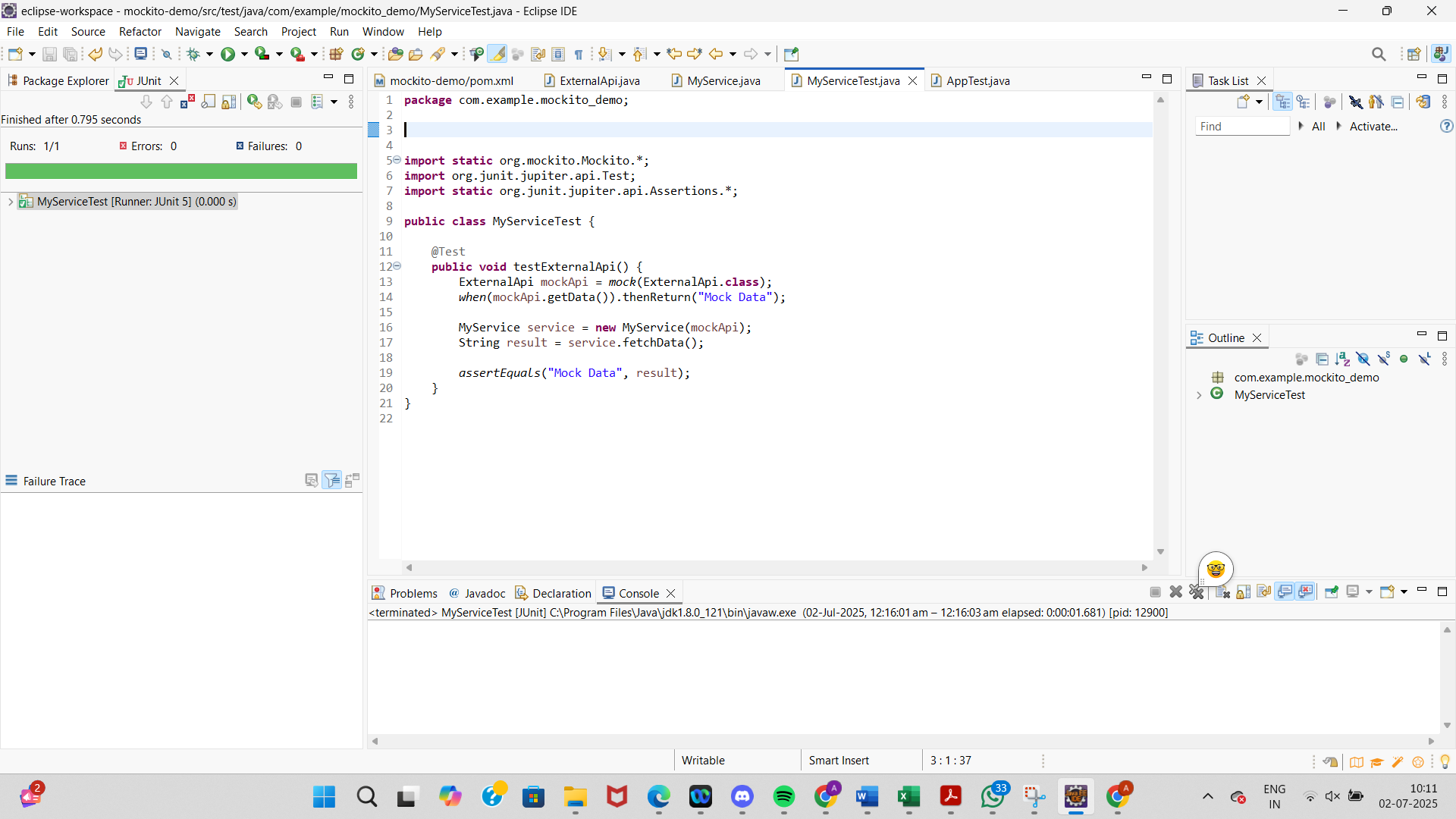
when(mockApi.getData()).thenReturn("Mock Data");

MyService service = new MyService(mockApi);

String result = service.fetchData();

assertEquals("Mock Data", result);

}



**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();

}

**MyService.java**

package com.example;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**VerifyInteractionTest.java**

package com.example.mockito\_demo;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

public class VerifyInteractionTest {

@Test

public void testVerifyInteraction() {

ExternalApi mockApi = mock(ExternalApi.class);

when(mockApi.getData()).thenReturn("Mocked Response");

MyService service = new MyService(mockApi);

service.fetchData();

verify(mockApi).getData();

}

}

