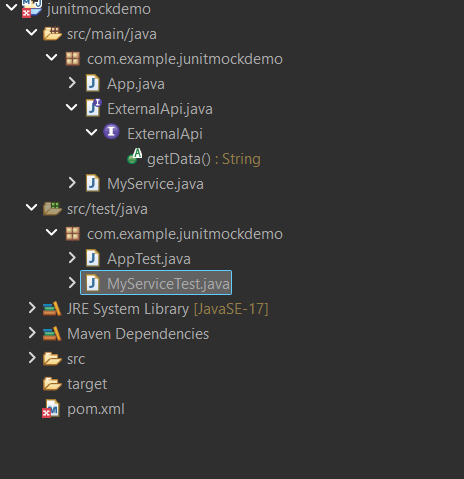
**Exercise 1: Mocking and Stubbing**

**Scenario: You need to test a service that depends on an external API. Use Mockito to mock the external API and stub its methods.**

**FOLDER STRUCTURE**

****

**ExternalApi.java**

package com.example.junitmockdemo;

public interface ExternalApi {

String getData();

}

**MyService.java**

package com.example.junitmockdemo;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**MyServiceTest**.**java**

package com.example.junitmockdemo;

import static org.mockito.Mockito.\*;

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

import org.junit.jupiter.api.Test;

public class MyServiceTest {

*@Test*

public void testExternalApi() {

ExternalApi mockApi = *mock*(ExternalApi.class);

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

MyService service = new MyService(mockApi);

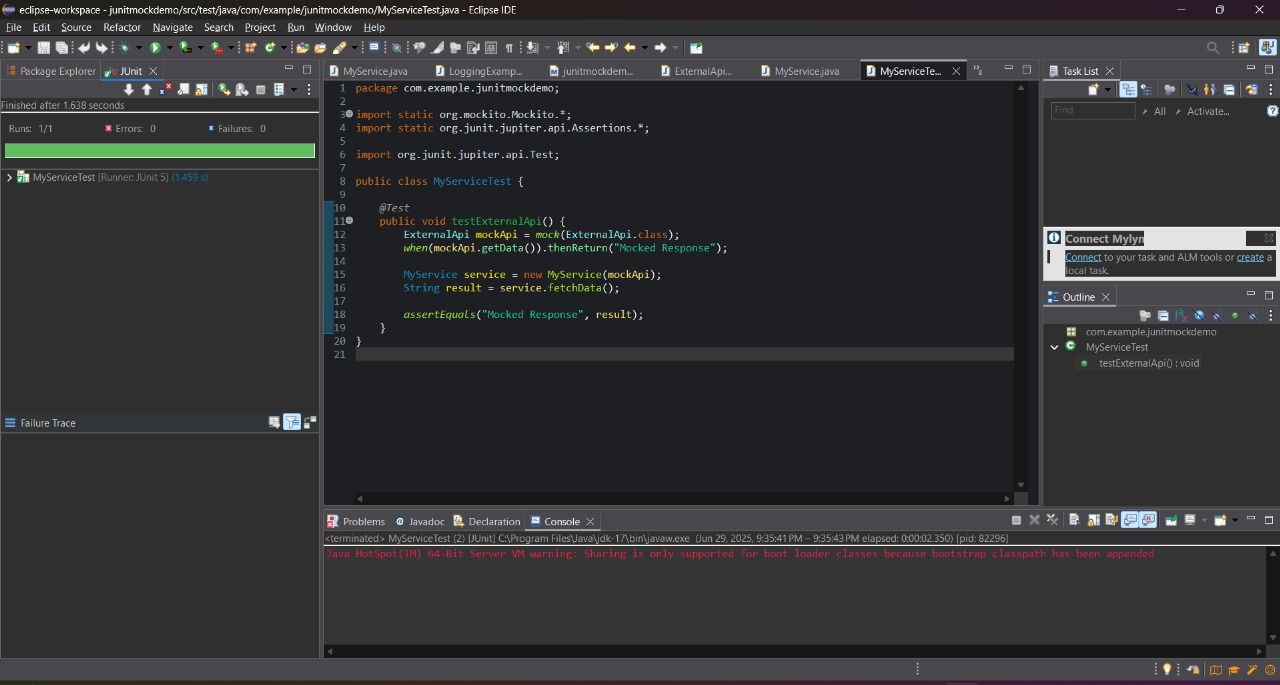
String result = service.fetchData();

*assertEquals*("Mocked Response", result);

}

}

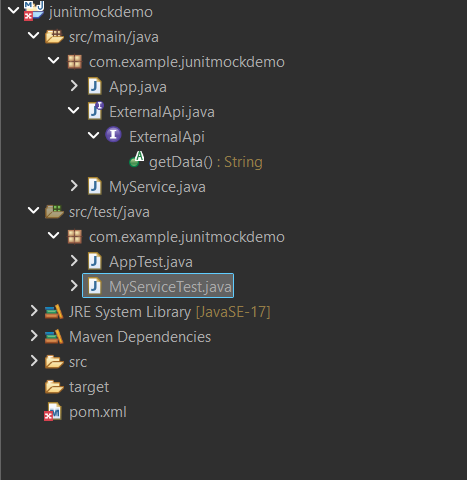
**OUTPUT**



**Exercise 2: Verifying Interactions**

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

**FOLDER STRUCTURE**

****

**MyServiceTest.java**

package com.example.junitmockdemo;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

public class MyServiceTest {

*@Test*

public void testVerifyInteraction() {

ExternalApi mockApi = *mock*(ExternalApi.class);

MyService service = new MyService(mockApi);

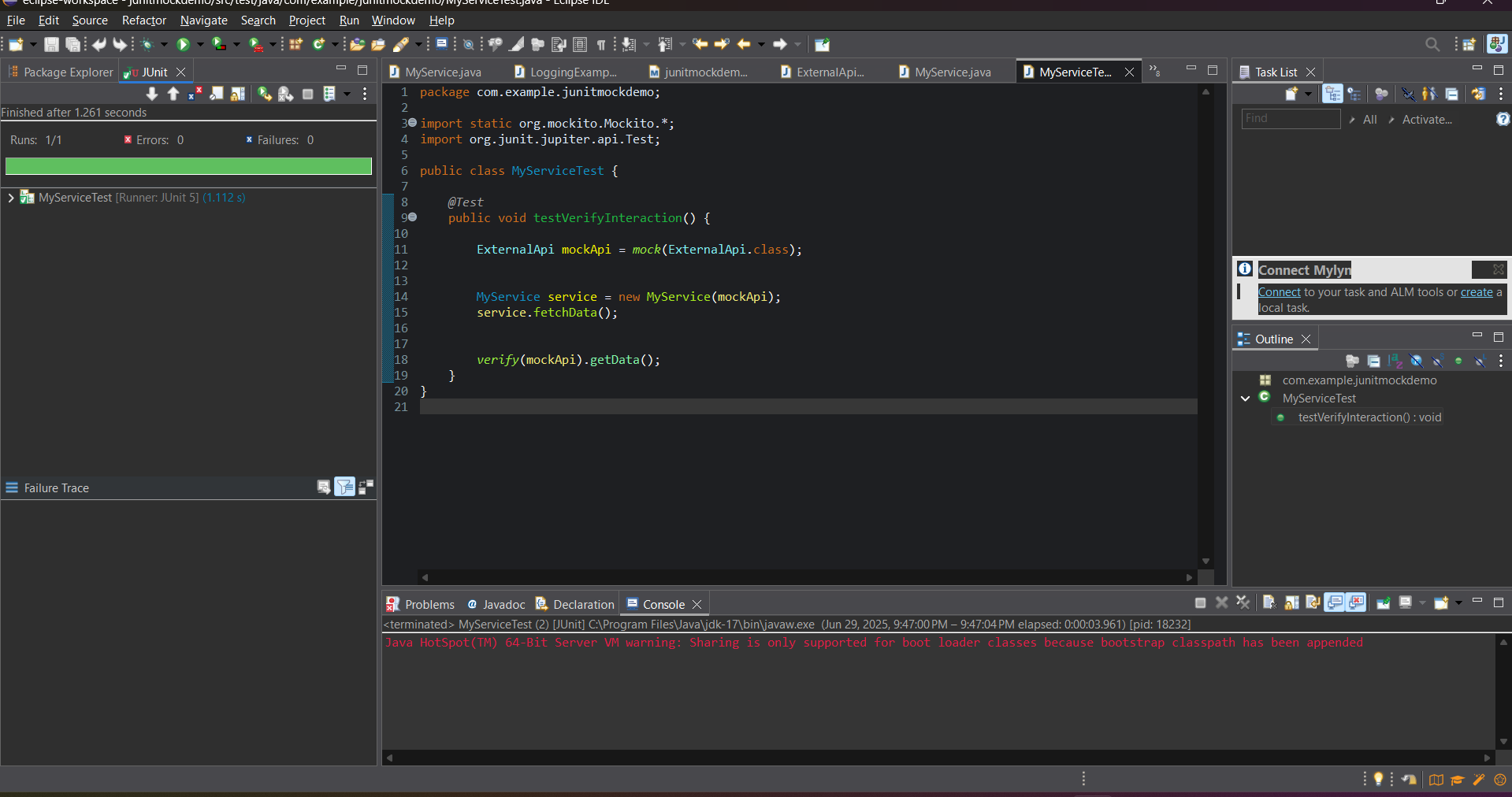
service.fetchData();

*verify*(mockApi).getData();

}

}

**OUTPUT**

****