**Exercise 1: Mocking and Stubbing**

**Directory:** c:\Users\shreya ghosh\Downloads\deepskilling\

├── pom.xml

└── src\

├── main\

│ └── java\

│ ├── Service.java

│ └── ExternalApi.java

└── test\

└── java\

└── ServiceTest.java

Code for Service.java:

public class Service {

    ExternalApi externalApi;

    public Service(ExternalApi externalApi) {

        this.externalApi = externalApi;

    }

    public String fetchData() {

        return  externalApi.getData();

    }

}

**Code for ExternalApi.java:**

public class ExternalApi {

    public String getData() {

        // Simulate fetching data from an external API

        return "RealData";

    }

}

**Code for ServiceTest.java:**

import org.junit.jupiter.api.Assertions;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class ServiceTest {

    ExternalApi externalApi=Mockito.mock(ExternalApi.class);

    Service service = new Service(externalApi);

    @Test

    public void testFetchData() {

        // Arrange

        Mockito.when(externalApi.getData()).thenReturn("MockedData");

        // Act

        String result = service.fetchData();

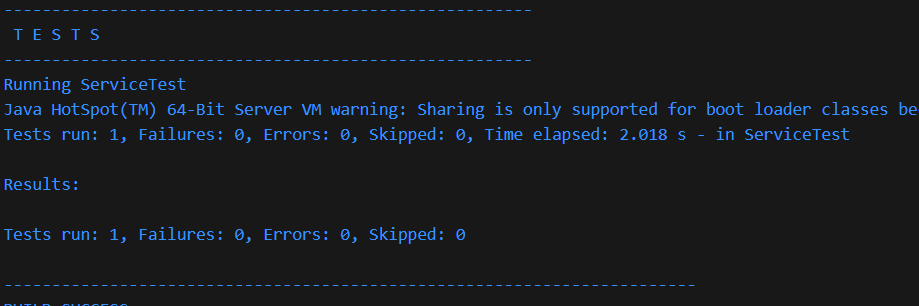
        // Assert

        Assertions.assertEquals("MockedData", result);

    }

}

Output:



**Exercise 2: Verifying Interactions**

Directory: c:\Users\shreya ghosh\Downloads\deepskilling\

├── pom.xml

└── src\

├── main\

│ └── java\

│ ├── Service.java

│ └── ExternalApi.java

└── test\

└── java\

└── ServiceTest.java

**Code for ServiceTest.java:**

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

import static org.mockito.Mockito.verify;

public class ServiceTest {

    ExternalApi externalApi=Mockito.mock(ExternalApi.class);

    Service service = new Service(externalApi);

    @Test

    public void testFetchData() {

        // Arrange

        Mockito.when(externalApi.getData()).thenReturn("MockedData");

        // Act

        String result = service.fetchData();

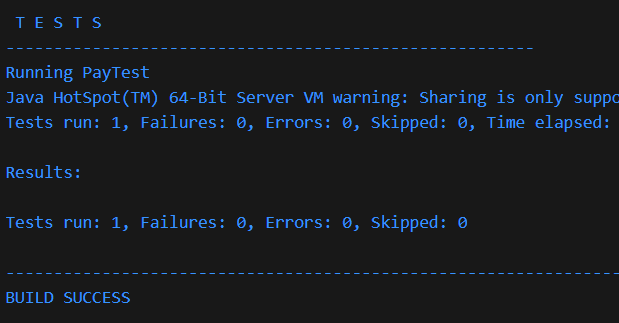
       //Verify

        verify(externalApi).getData();

    }

}

Output:



**Exercise 3: Argument Matching**

Directory: c:\Users\shreya ghosh\Downloads\deepskilling\

├── pom.xml

└── src\

├── main\

│ └── java\

│ ├── Logger.java

│ └── PaymentService.java

└── test\

└── java\

└── PayTest.java

Code for Logger.java:

public class Logger {

    public void log(String message) {

        System.out.println("Log: " + message);

    }

}

**Code for PaymnetService.java:**

public class PaymentService {

    Logger logger;

    public PaymentService(Logger logger) {

        this.logger = logger;

    }

    public void pay(String useId,String amount) {

        logger.log("Payment processed for user: " + useId + " of amount: " + amount);

    }

}

**Code for PayTest.java:**

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

import static org.mockito.ArgumentMatchers.contains;

import static org.mockito.Mockito.verify;

public class PayTest {

    Logger logger=Mockito.mock(Logger.class);

    PaymentService paymentService = new PaymentService(logger);

   @Test

    public void testPay() {

        String userId = "user123";

        String amount = "100";

        paymentService.pay(userId, amount);

        verify(logger).log("Payment processed for user: " + userId + " of amount: " + amount);

    }

}

**Output:**

