Cognizant Java FSE – (Deep Skilling)

(WEEK-2)

**MODULE 1:** PL/SQL programming

**MODULE 2:** TDD using JUnit5 and Mockito

**MODULE 3:** SLF4J logging framework

**Submitted by**  
**Name:** SOLLETI VENKATA KUSUMA  
**Roll No:** 111522102145  
**Email:** [22102145@rmd.ac.in](mailto:22102145@rmd.ac.in)

**College:** RMD ENGINEERING COLLEGE  
**Batch:** Java FSE – 2026

**MODULE 1: PL/SQL programming**

**EXERCISE 1**: Control Structures

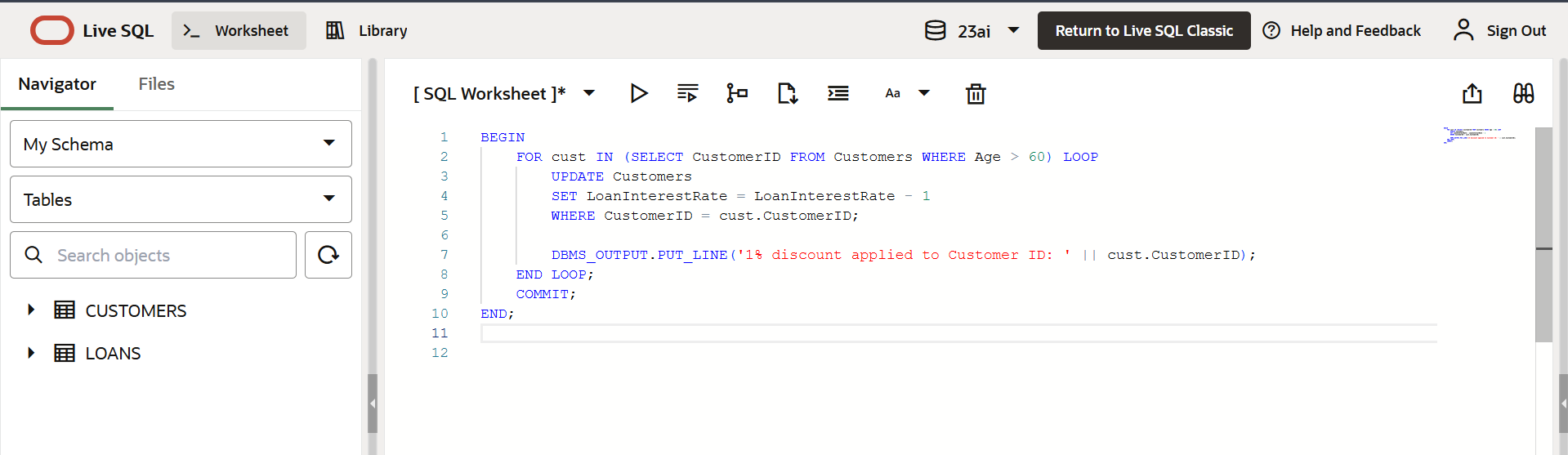
**Scenario 1:** The bank wants to apply a discount to loan interest rates for customers above 60 years old.

**Question:** Write a PL/SQL block that loops through all customers, checks their age, and if they are above 60, apply a 1% discount to their current loan interest rates.

A screenshot of a computer

AI-generated content may be incorrect.

1. Firstly, we have to create the CUSTOMERS and LOANS table and insert some data into the tables.
2. Here I have used Oracle Live SQL to run this entire task.
3. Once we have done with creating and inserting into the tables we run the scripts for the scenarios given.
4. In the scenario 1 we have to run the script that apply 1% discount on their current loan rates for the people of age group above 60.



OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.

**Scenario 2:** A customer can be promoted to VIP status based on their balance.

**Question:** Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over $10,000.

* In this scenario, we have to write the script to set the customers with balance over $10,000 as VIP’S.
* Here we iterate through all the customers and check the balance of customers and set IsVIP flag to true for the customers whose balance is over $10,000.

A screenshot of a computer

AI-generated content may be incorrect.

OUTPUT:

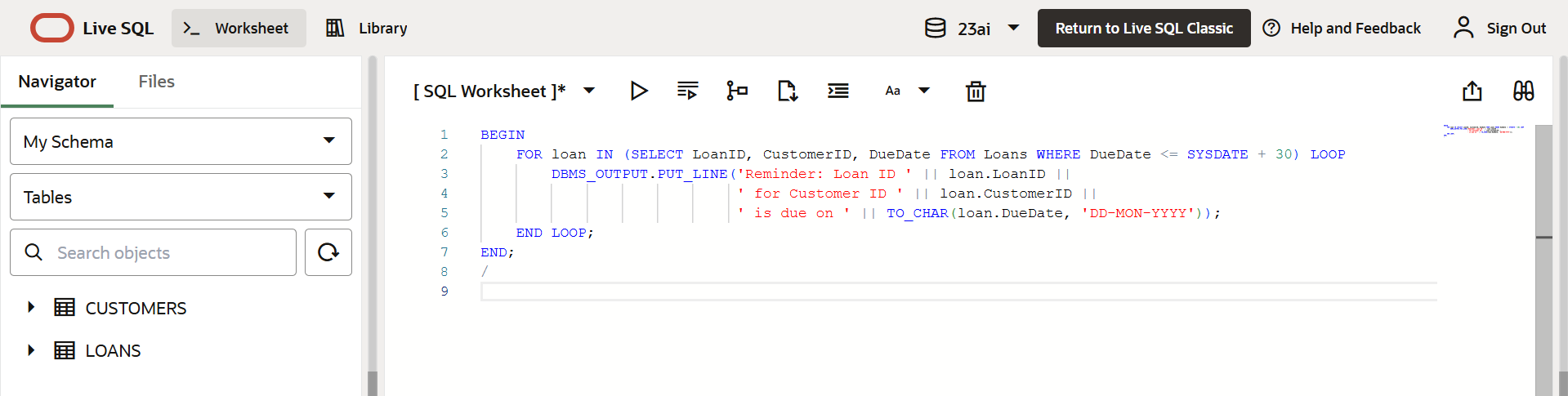
A screenshot of a computer

AI-generated content may be incorrect.

**Scenario 3:** The bank wants to send reminders to customers whose loans are due within the next 30 days.

**Question:** Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.

* In this scenario, we have to write the script that fetches all loans due in the next 30 days and prints a remainder message for each customer.



OUTPUT:

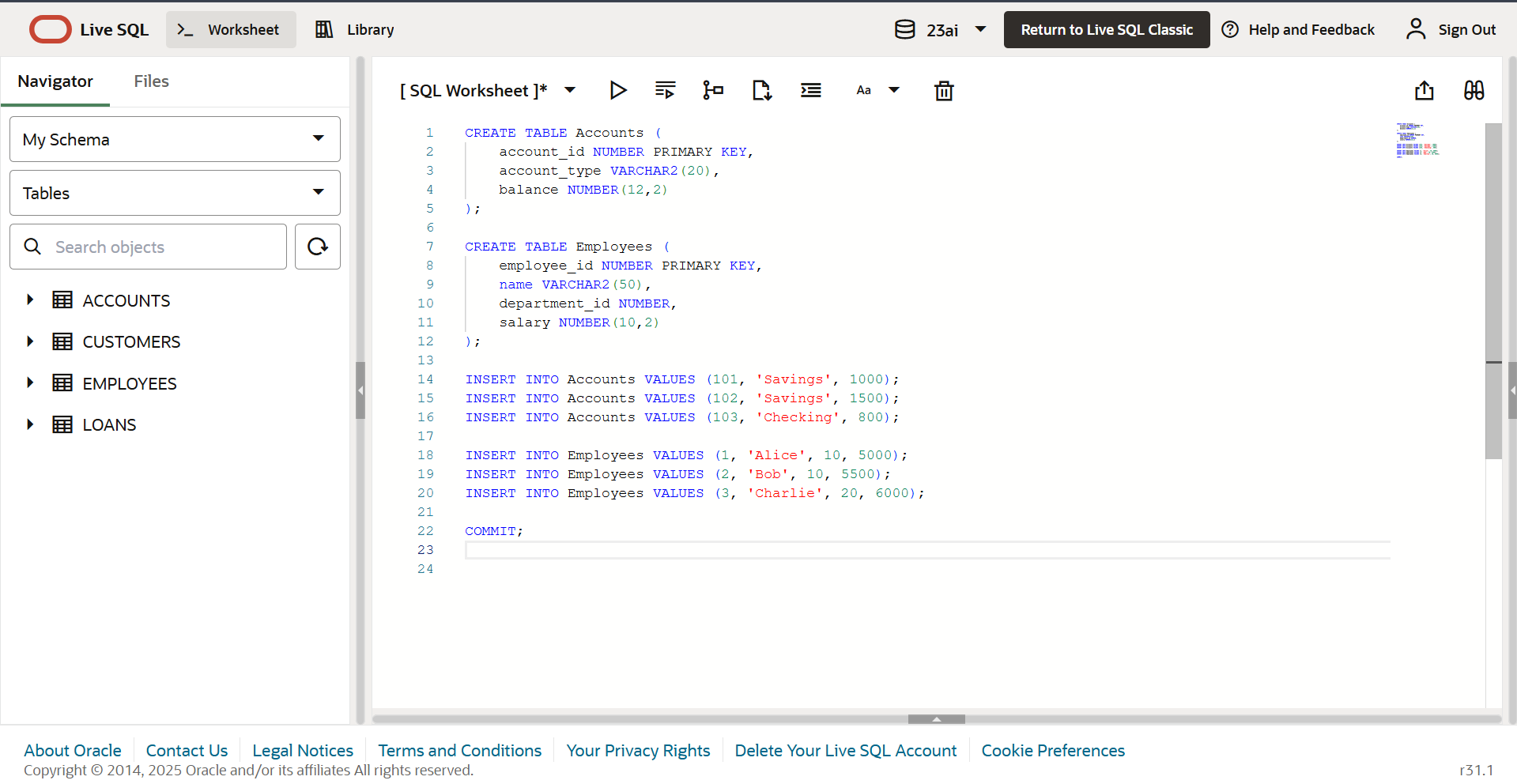
A screenshot of a computer

AI-generated content may be incorrect.

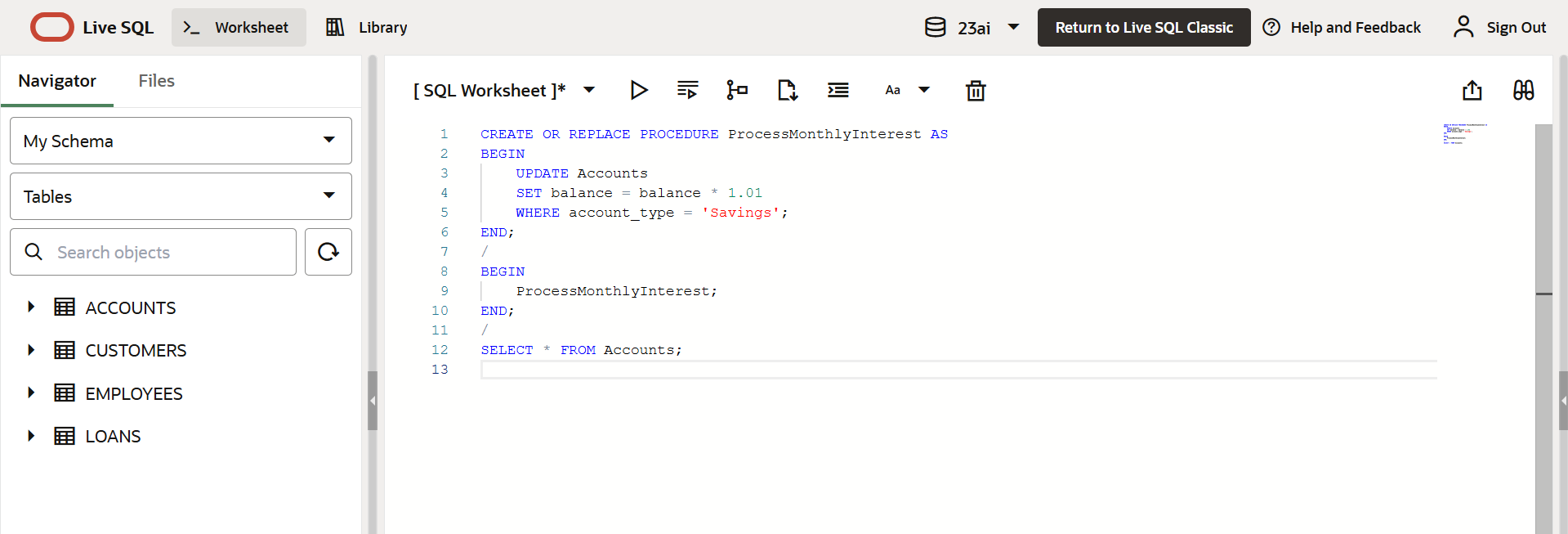
**Exercise 3**: Stored Procedures

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

**Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.



* Here we have created Accounts and Employees table and inserted few data into both the tables.
* Now we have to create the procedures and test and run them to see the output.



OUTPUT:

A white background with black and white clouds

AI-generated content may be incorrect.

**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

**Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.



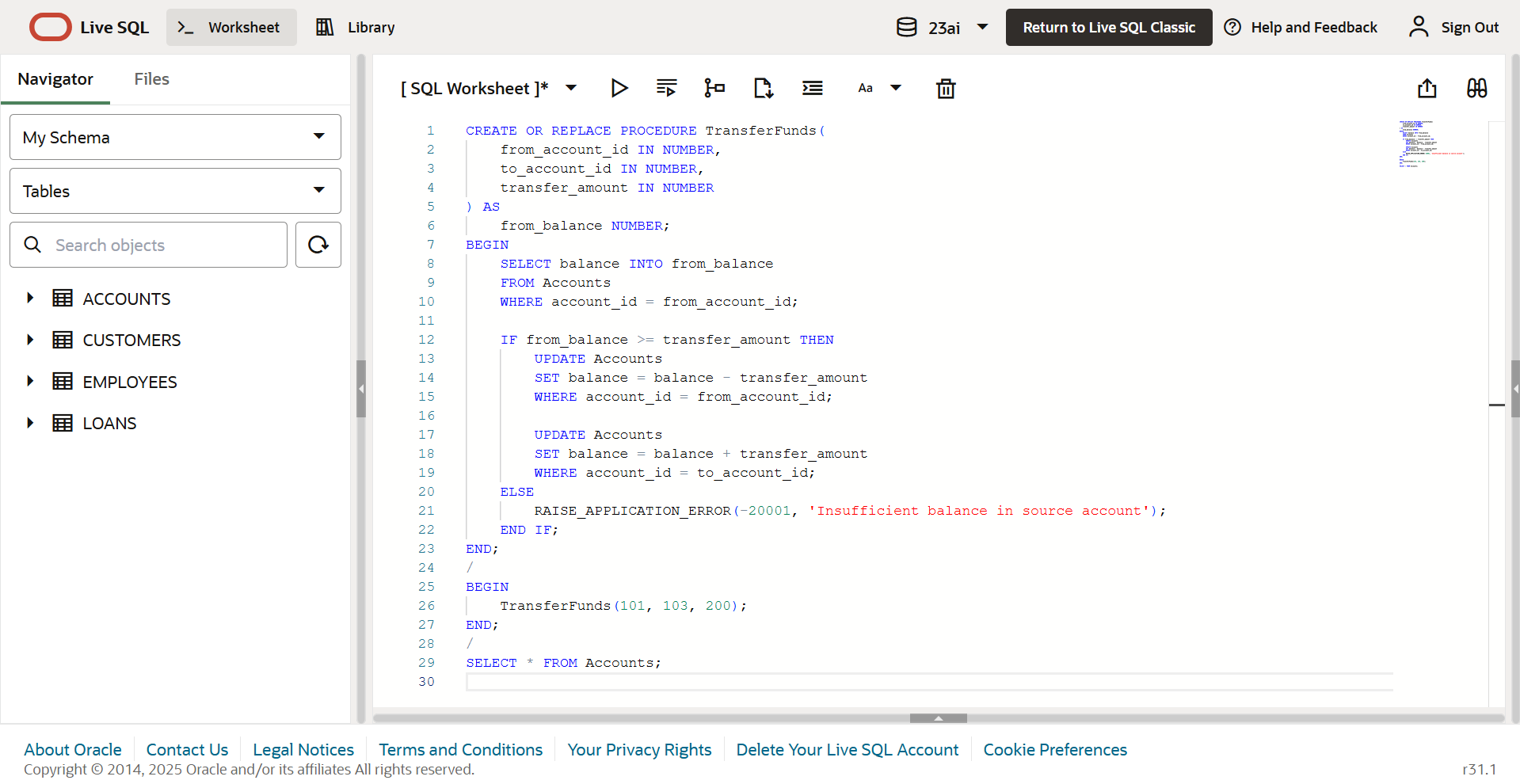
OUTPUT:

A white background with black and white text

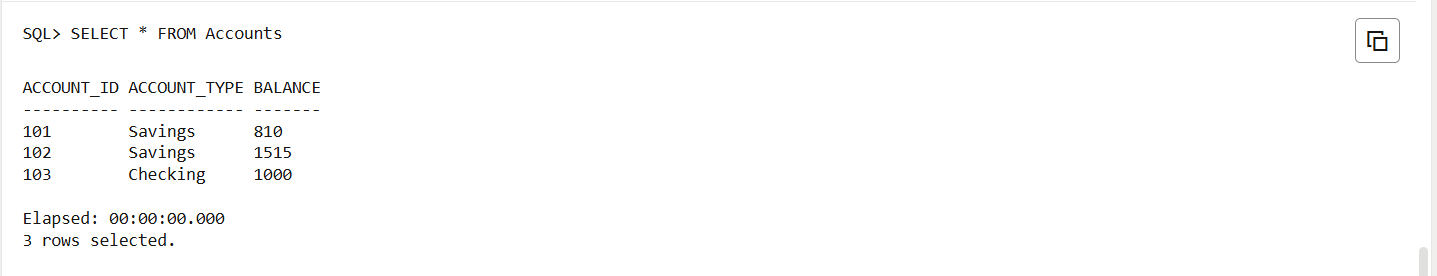
AI-generated content may be incorrect.

**Scenario 3:** Customers should be able to transfer funds between their accounts.

**Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.



OUTPUT:



**MODULE 2: TDD using JUnit5 and Mockito**

**Exercise 1**: Setting Up JUnit

**Scenario:**

You need to set up JUnit in your Java project to start writing unit tests.

Steps:

1. Create a new Java project in your IDE (e.g., IntelliJ IDEA, Eclipse).

2. Add JUnit dependency to your project. If you are using Maven, add the following to your

pom.xml:

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

3. Create a new test class in your project.

**Step 1: Create Maven Project in IntelliJ**

* Open IntelliJ IDEA.
* Select **New Project > Maven**.
* Set the **project name** to: FactorialCalculator.
* Choose a valid **JDK version**.
* Click **Create**.

**Step 2: Update pom.xml with JUnit Dependency**

**A screenshot of a computer program

AI-generated content may be incorrect.**

Then click the **Reload Maven** icon.

**Step 3: Create the Factorial Class**

* Go to src/main/java/com/example/
* create a file named Factorial.java

A screenshot of a computer program

AI-generated content may be incorrect.

**Step 4: Create the Test Class for Factorial**

* Create folder: src/test/java/com/example/
* Inside it, create FactorialTest.java

A screenshot of a computer program

AI-generated content may be incorrect.

Run the FactorialTest.java class to see the output

OUTPUT:

A screen shot of a computer

AI-generated content may be incorrect.

**Exercise 3:** Assertions in JUnit

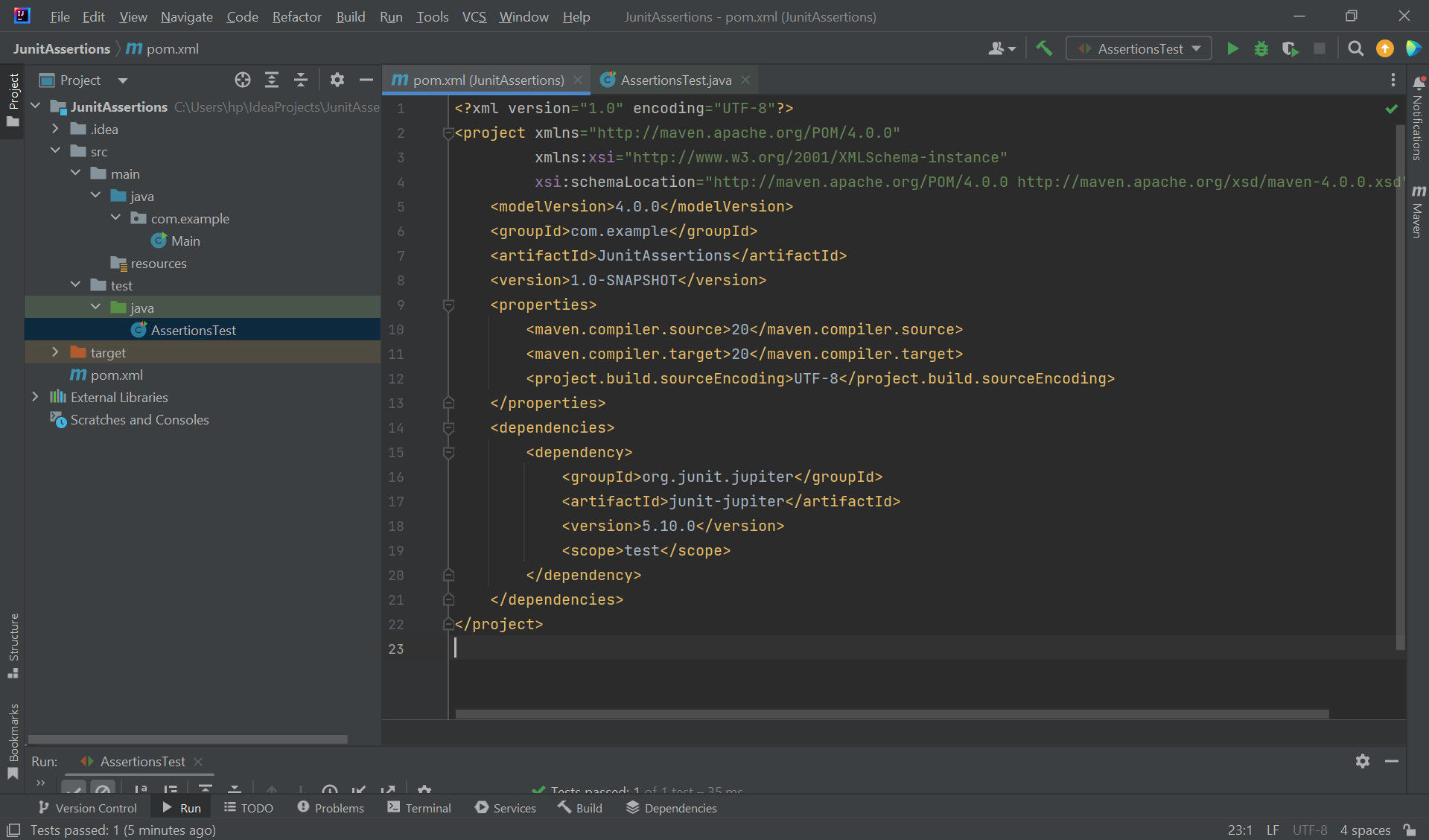
**Scenario:** You need to use different assertions in JUnit to validate your test results.

1. Write tests using various JUnit assertions.

**Step-1. Create a Maven Project in IntelliJ**

1. Open IntelliJ IDEA.
2. Click on New Project.
3. Choose Maven on the left panel.
4. Make sure JDK is selected (Java 11+ recommended).
5. Click Next → Give GroupId (com.example) and ArtifactId ( junit-assertions) → Click Finish.

**Step 2: Update pom.xml with JUnit Dependency**

****

Then click the **Reload Maven** icon.

**3. Create the Test Class**

* Right-click on src/test/java → New → Java Class.
* Name it: AssertionsTest.

A screenshot of a computer

AI-generated content may be incorrect.

Run the AssertionsTest.java class to see the output

OUTPUT:

A screen shot of a computer

AI-generated content may be incorrect.

**Exercise 4:** Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit

**Scenario:**

You need to organize your tests using the Arrange-Act-Assert (AAA) pattern and use setup and teardown methods.

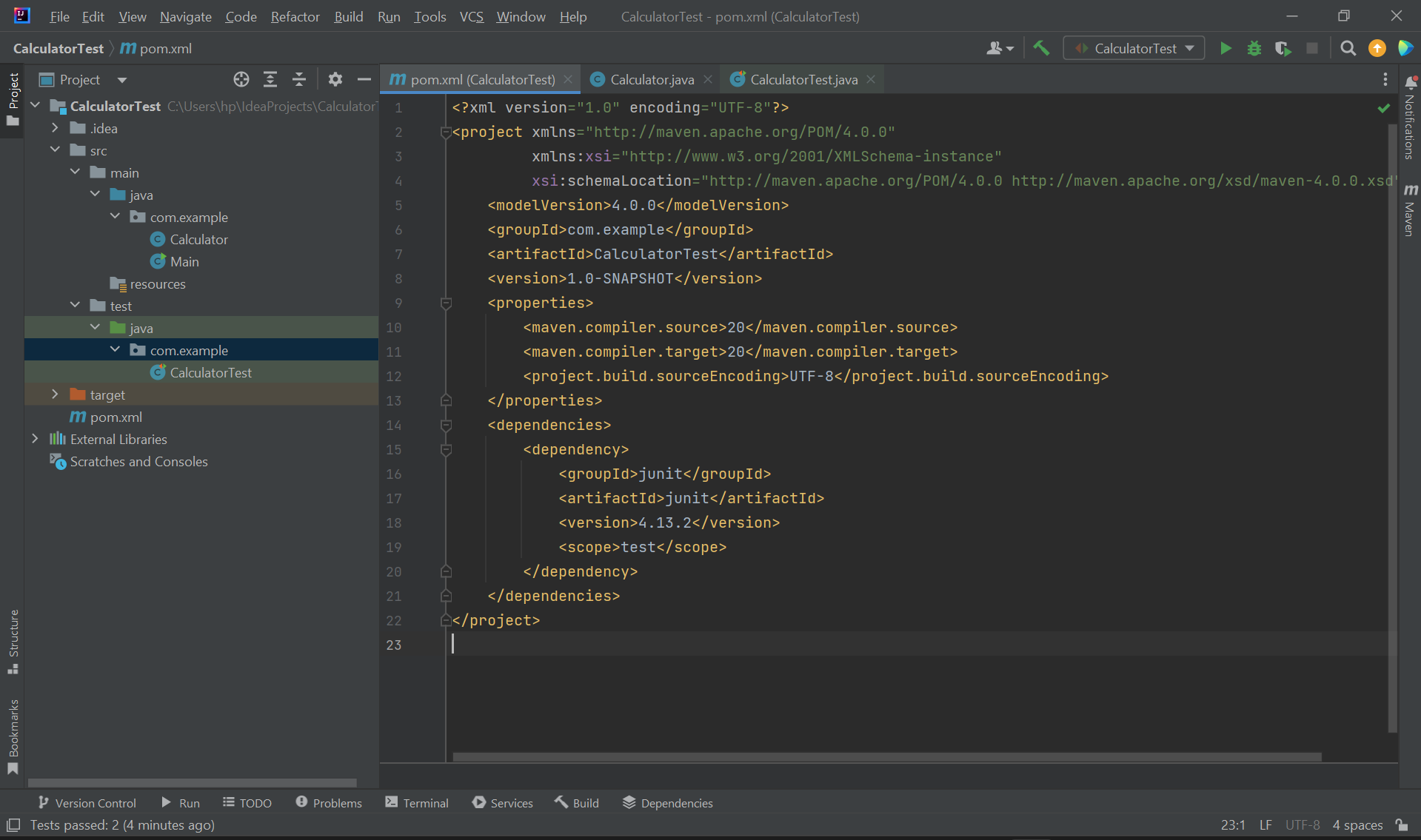
Steps:

1. Write tests using the AAA pattern.
2. .Use @Before and @After annotations for setup and teardown methods.

**Step 1: Create a New Maven Project**

1. Open IntelliJ IDEA.
2. Click on New Project named CalculatorTest.
3. On the left panel, select Maven.
4. GroupId: com.example
5. ArtifactId: CalculatorTest
6. Click Next, then Finish.

**Step 2: Add JUnit to pom.xml**

****

**Step 3:** **Create the Calculator Class**

* src/main/java/com/example/Calculator.java

**A screenshot of a computer

AI-generated content may be incorrect.**

**Step 4:** **Create the CalculatorTest Class**

* src/test/java/com/example/CalculatorTest.java

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

Run the CalculatorTest.java class to see the output

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.

**MOCKITO EXCERCISES**

**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.

Steps:

1. Create a mock object for the external API.

2. Stub the methods to return predefined values.

3. Write a test case that uses the mock object.

**Step 1: Create a Maven Project**

1. Open IntelliJ IDEA.
2. Click New Project(MockitoTest) -Select Maven.
3. GroupId: com.example,ArtifactId: MockitoTest
4. Click Finish.

**Step 2: Add Junit and Mockito to pom.xml**

A screenshot of a computer program

AI-generated content may be incorrect.

**Step 3:Create an interface ExternalApi.java and a class MyService.java**

**ExternalApi.java**

src/main/java/com/example/ExternalApi.java

**A screenshot of a computer

AI-generated content may be incorrect.**

**MyService.java**

src/main/java/com/example/MyService.java

**A screenshot of a computer

AI-generated content may be incorrect.**

**Step 4:Create a Test class named MyServiceTest.java**

src/test/java/com/example/MyServiceTest.java

**A screenshot of a computer program

AI-generated content may be incorrect.**

Run the MyServiceTest.java class to see the output.

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.

**Exercise 2:** Verifying Interactions

**Scenario:**

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

Steps:

1. Create a mock object.

2. Call the method with specific arguments.

3. Verify the interaction.

**Step 1. Open IntelliJ and Create a Maven Project**

1. Click File → New → Project(VerifyingInteractions)
2. Select Maven.
3. GroupId: com.example and ArtifactId: VerifyingInteractions
4. Click Finish.

**Step 2: Add Junit and Mockito to pom.xml**

A screenshot of a computer program

AI-generated content may be incorrect.

**Step 3:Create an interface ExternalApi.java and a class MyService.java**

**ExternalApi.java**

src/main/java/com/example/ExternalApi.java

A screenshot of a computer

AI-generated content may be incorrect.

**MyService.java**

src/main/java/com/example/MyService.java

A screenshot of a computer

AI-generated content may be incorrect.

**Step 4:Create a Test class named MyServiceTest.java**

src/test/java/com/example/MyServiceTest.java

A screenshot of a computer

AI-generated content may be incorrect.

Run the MyServiceTest.java class to see the output

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.

**MODULE 3:** SLF4J logging framework

**Exercise 1**: Logging Error Messages and Warning Levels

**Task:**

Write a Java application that demonstrates logging error messages and warning levels using SLF4J.

**Step 1. Open IntelliJ and Create a Maven Project**

1. Click File → New → Project(LoggingDemo)
2. Select Maven.
3. GroupId: com.example and ArtifactId: LoggingDemo
4. Click Finish.

**Step 2: Add SLF4J and Logback dependencies to pom.xml**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Step 3:Create a java class Logging**

**A screenshot of a computer

AI-generated content may be incorrect.**

Run the Logging.java class to see the output

OUTPUT:

A screen shot of a computer

AI-generated content may be incorrect.