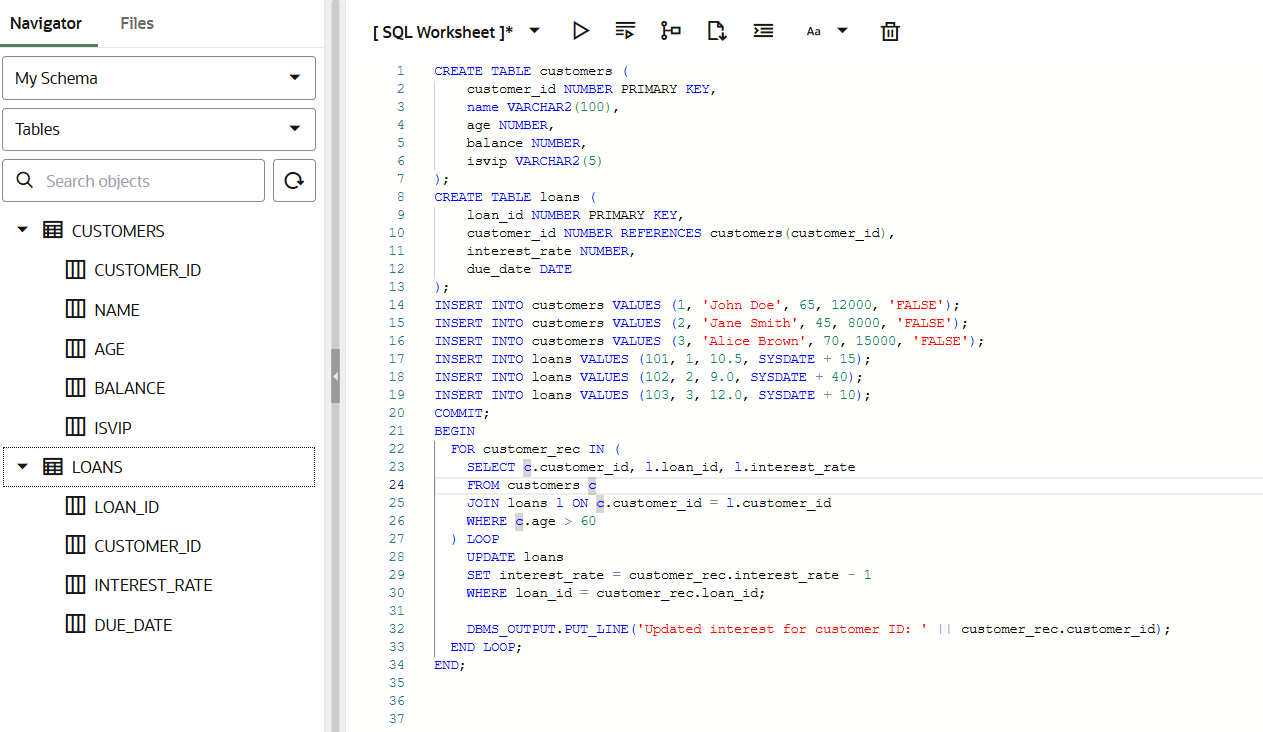
**WEEK 2 – HANDS ON**

**PL/SQL PROGRAMMING**

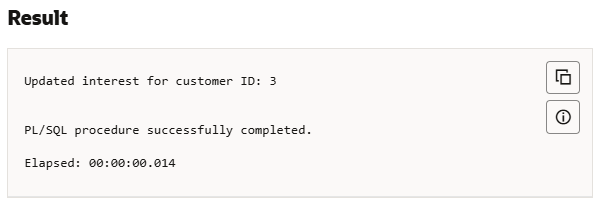
1. **EXERCISE 1: Control Structures**

**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.

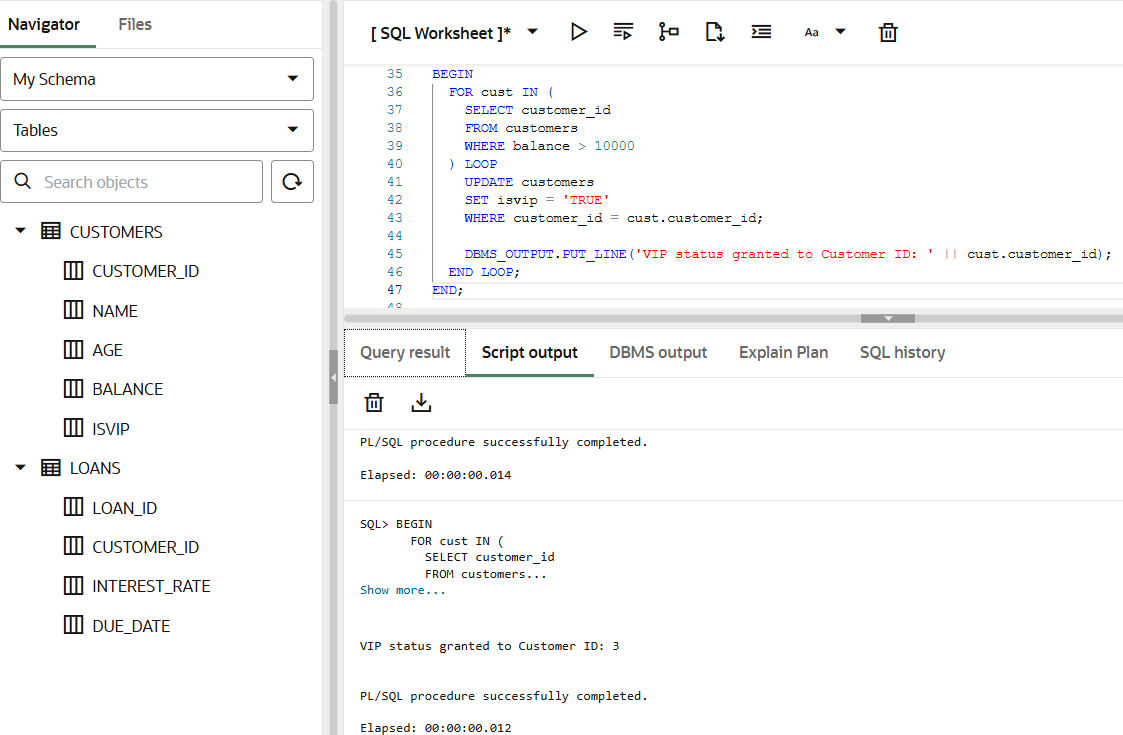
****

**OUTPUT:**

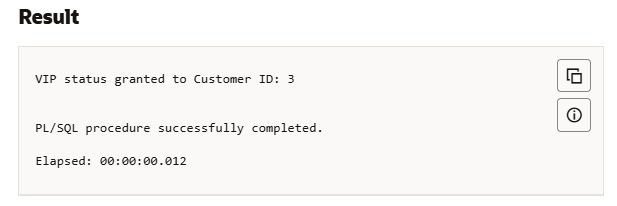
****

**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.

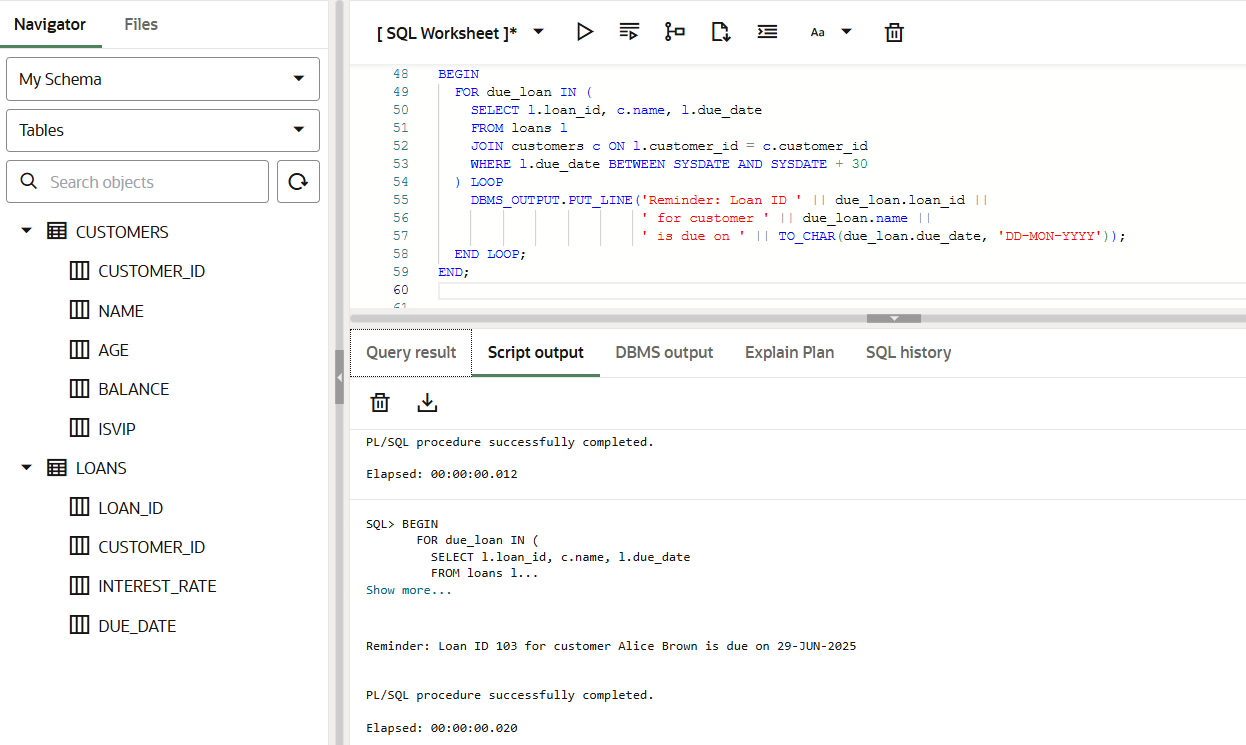
****

**OUTPUT:**

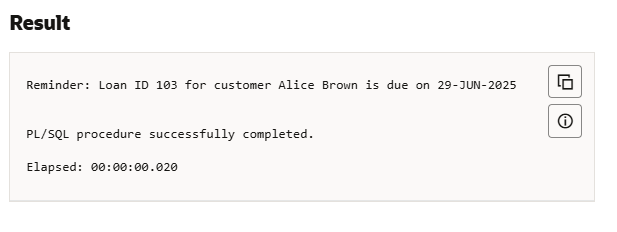
****

**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.



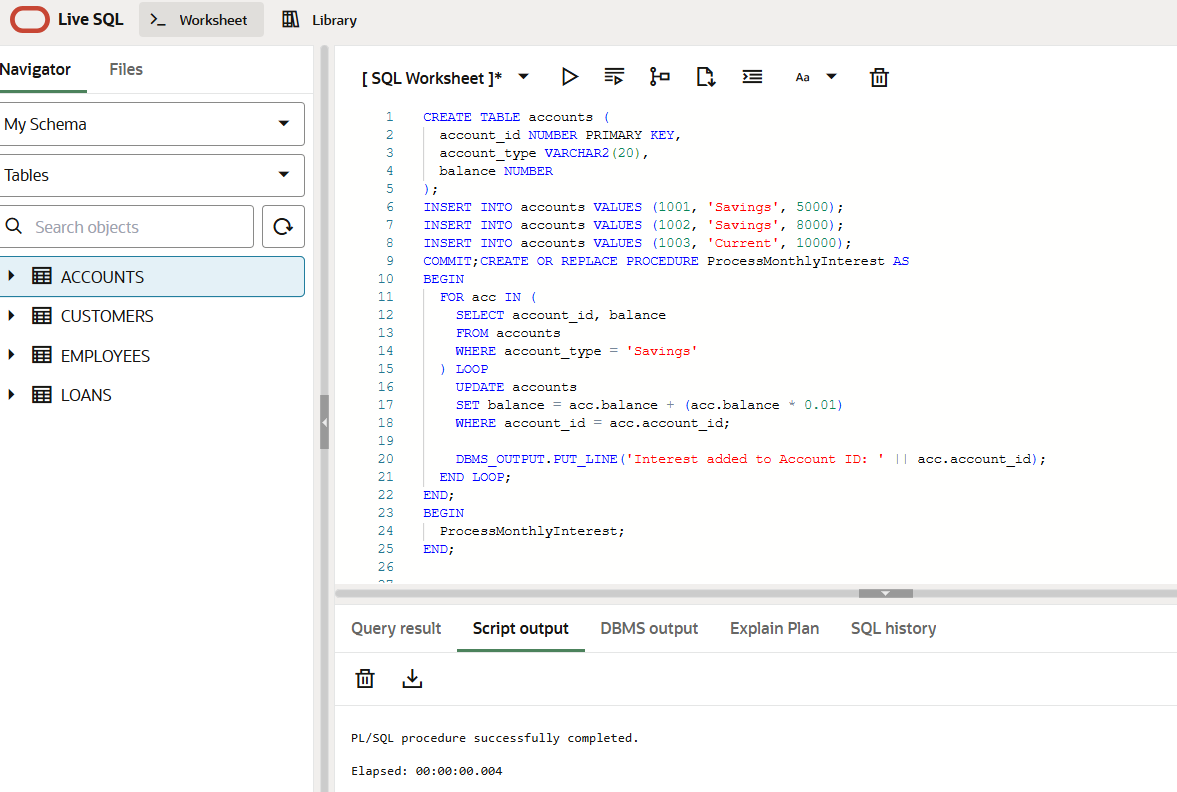
**OUTPUT:**

****

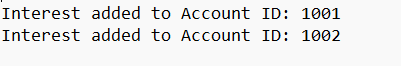
1. **EXERCISE 3: Stored Procedures**

**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.

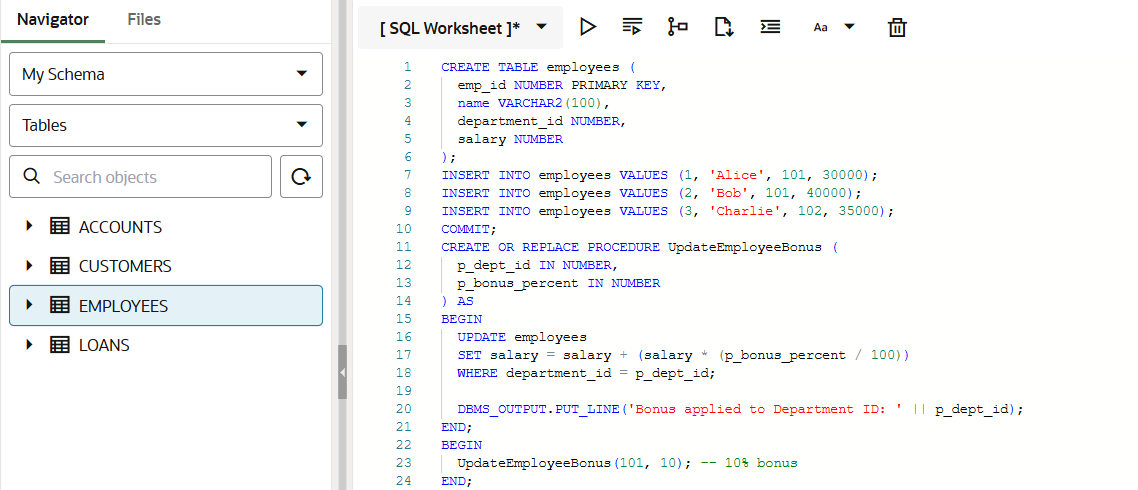


**OUTPUT:**

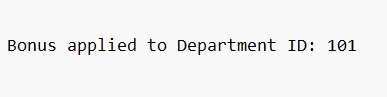
****

**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.

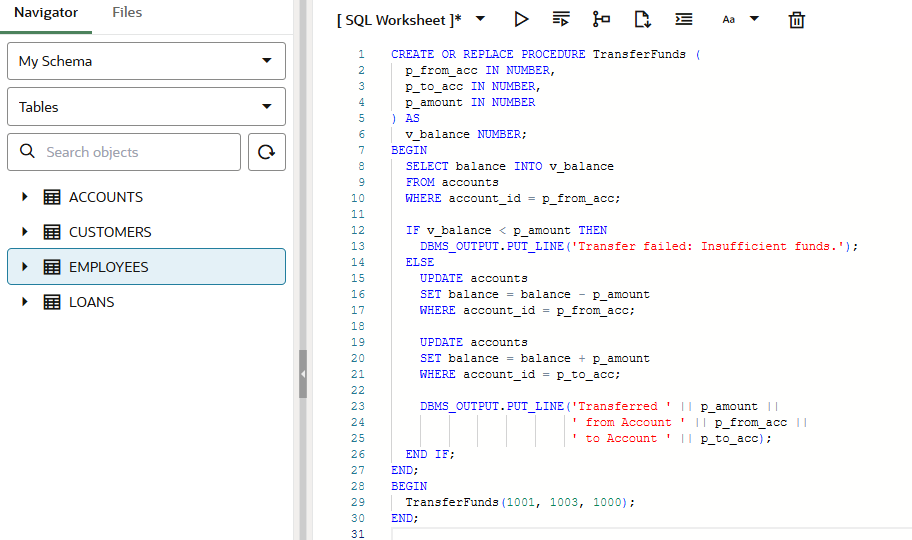
****

**OUTPUT:**

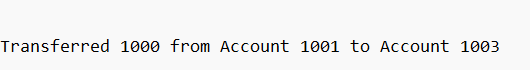
****

**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.

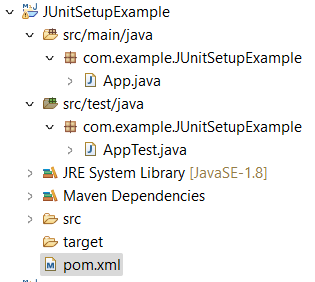


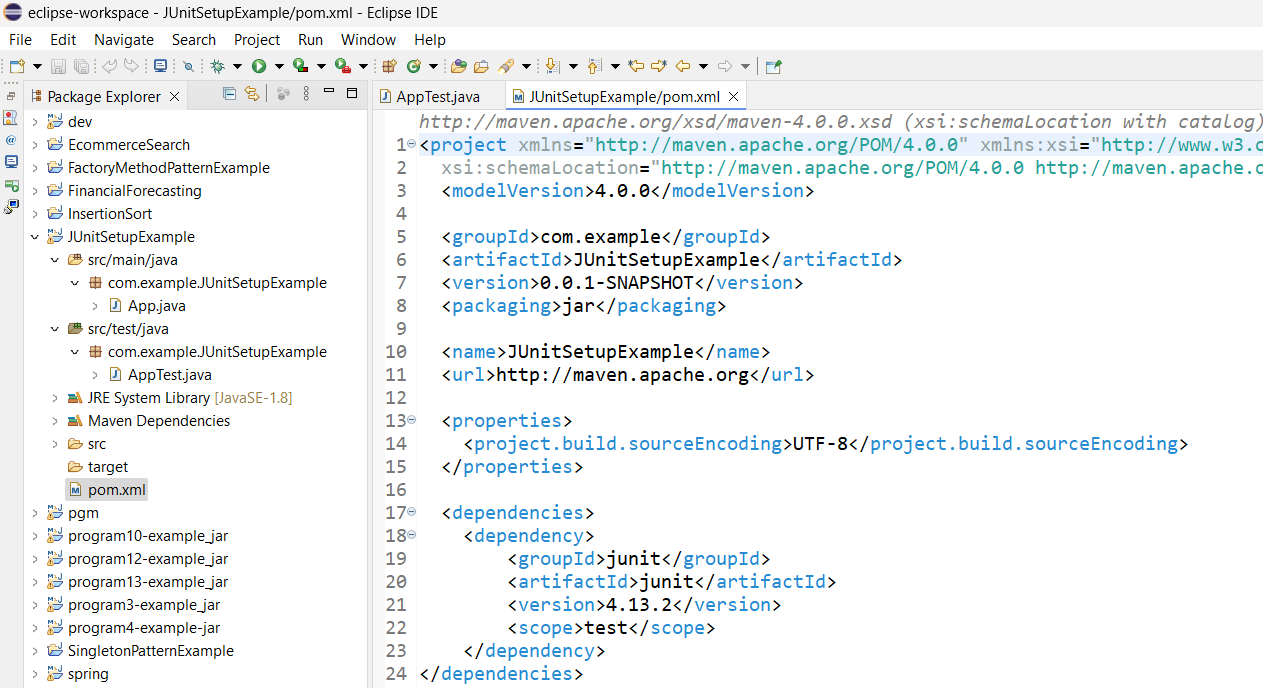
**OUTPUT:**

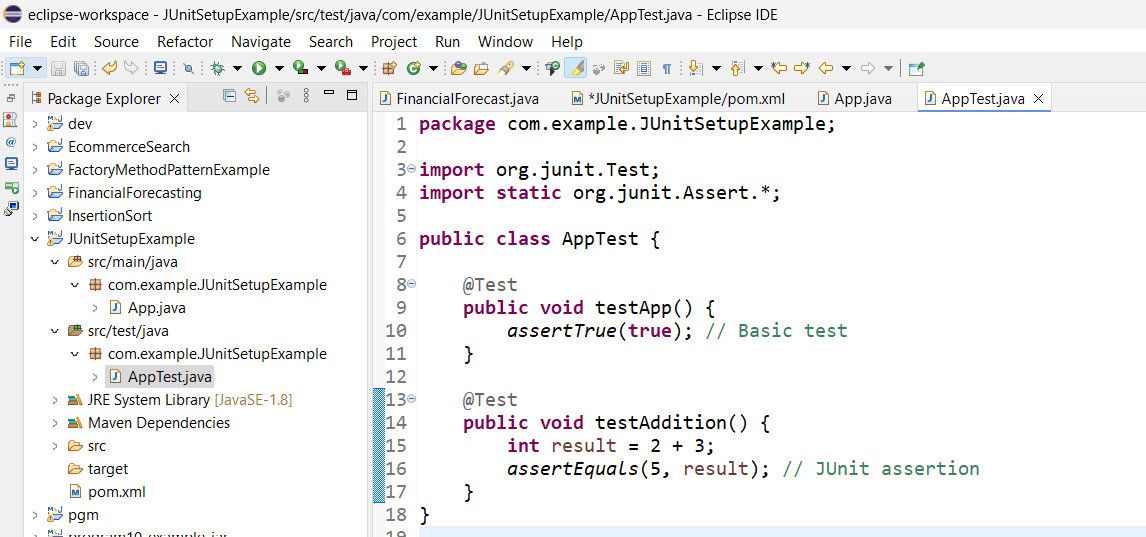
****

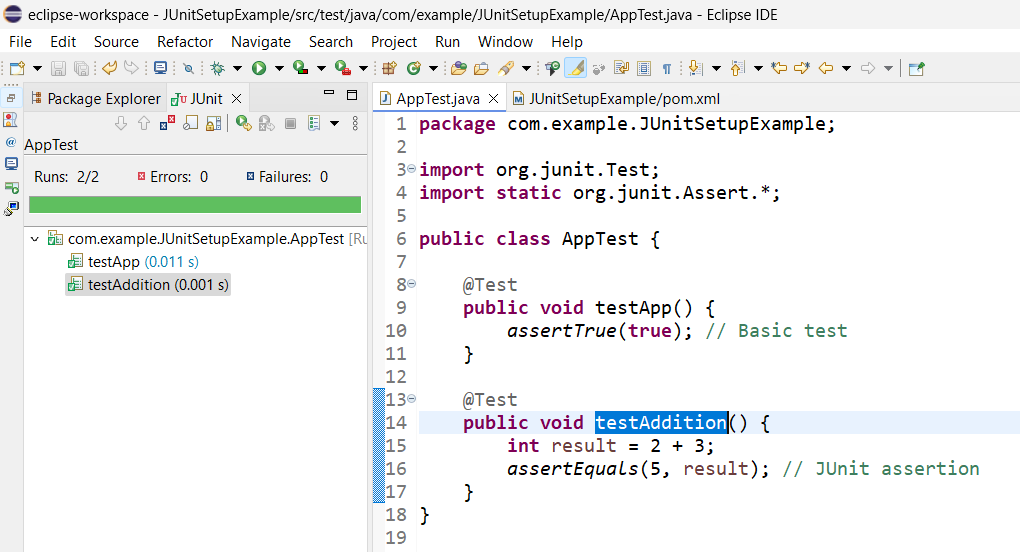
**TDD using JUnit5 and Mockito**

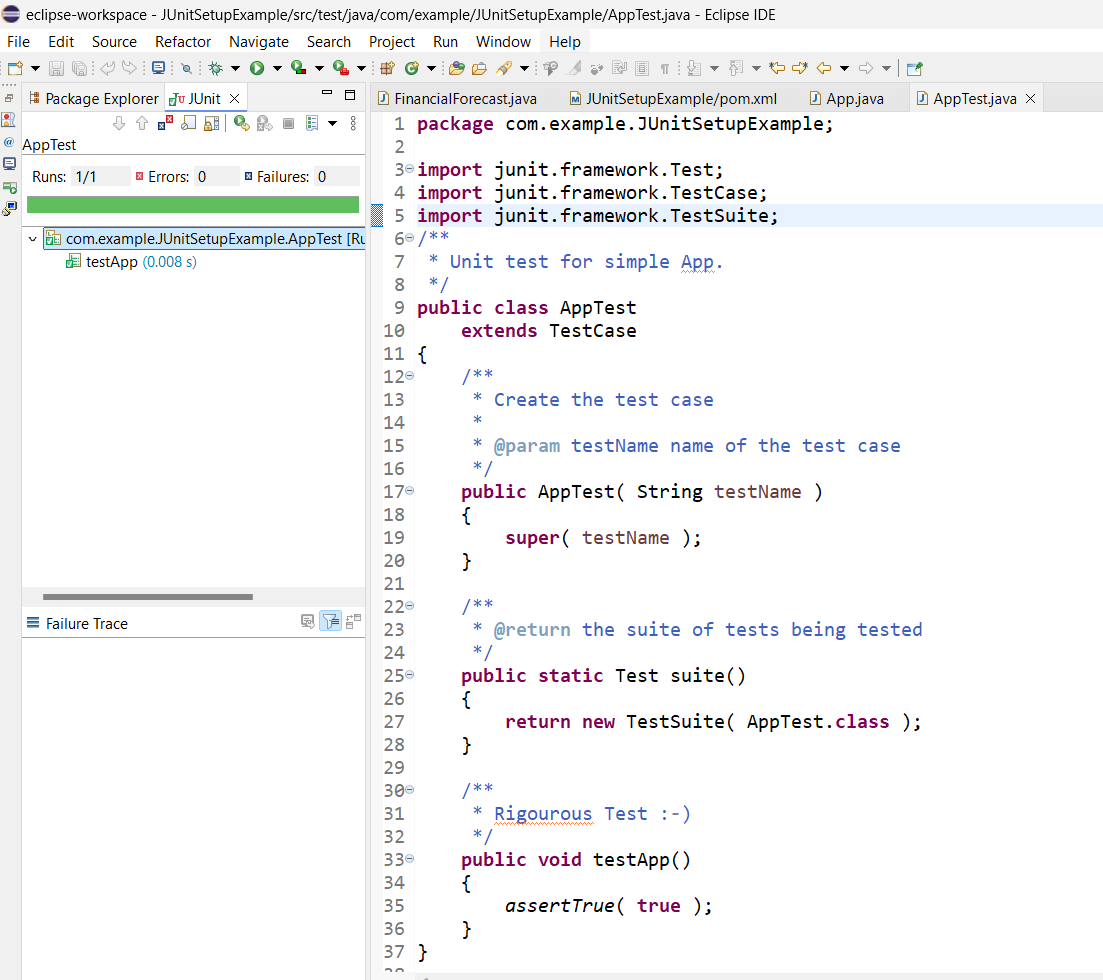
1. **Exercise 1: Setting Up Junit**

****

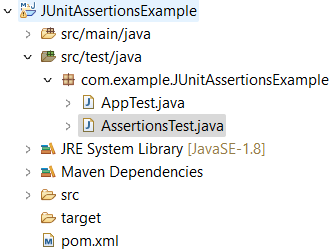
****

****

**OUTPUT:**

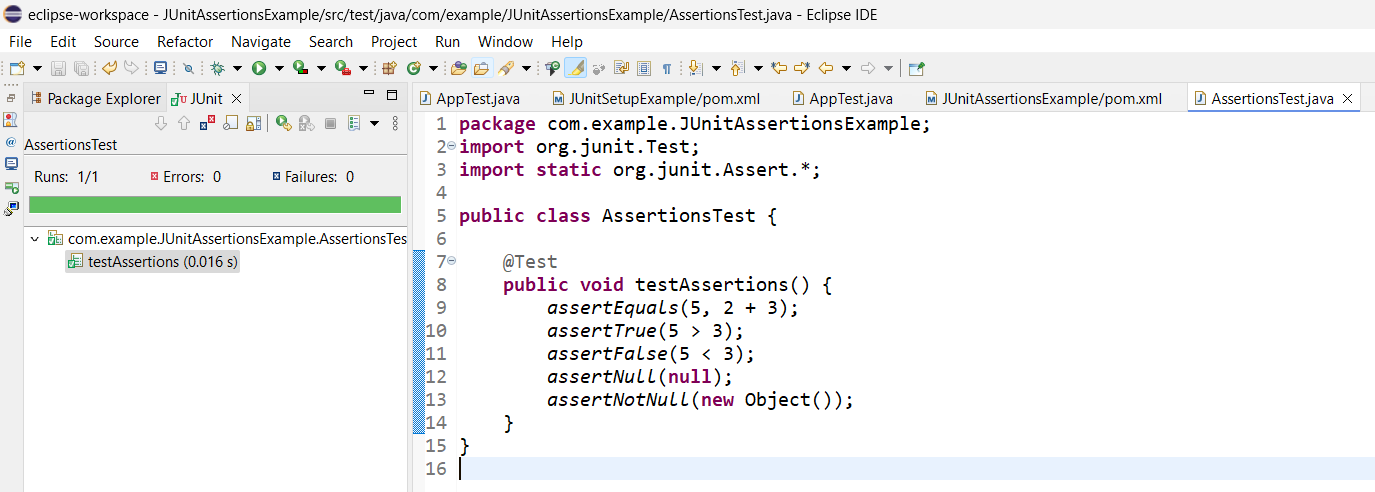
****

1. **Exercise 3: Assertions in Junit**

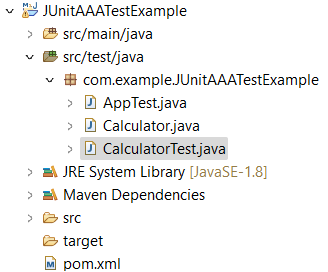
****

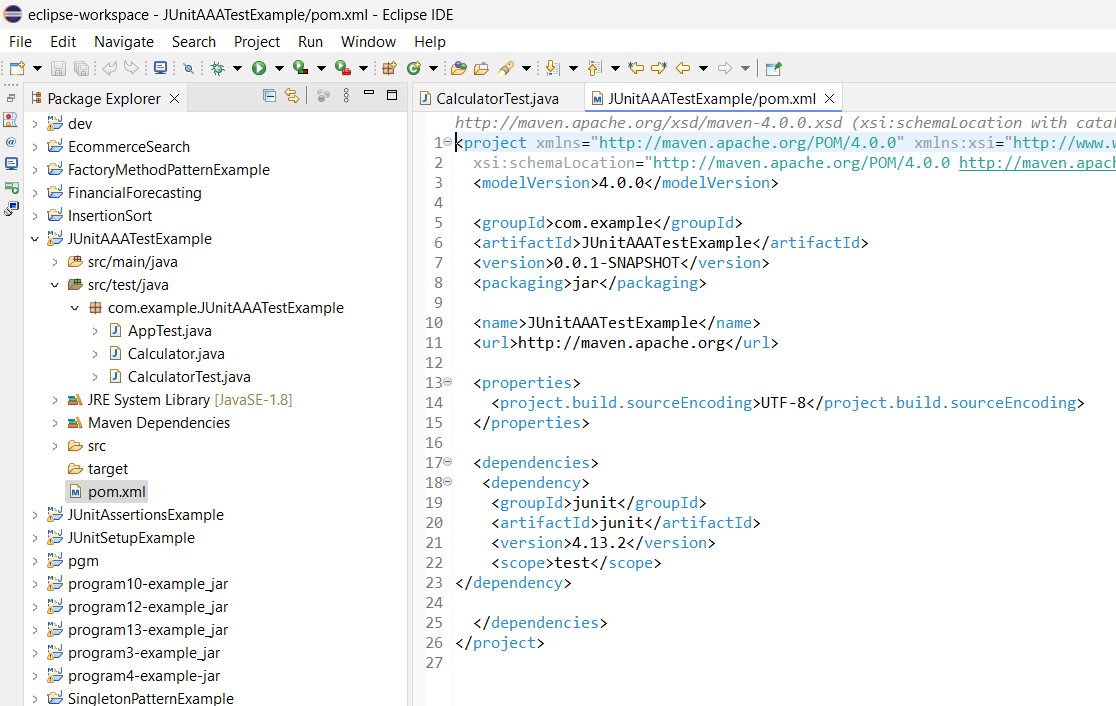
****

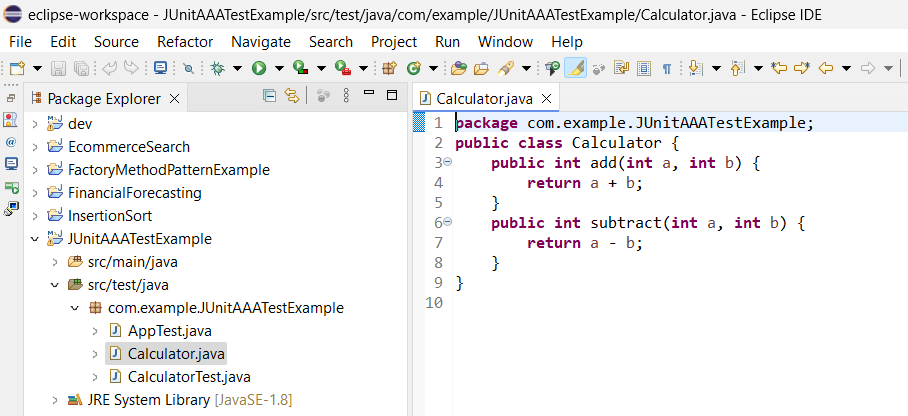
**OUTPUT:**

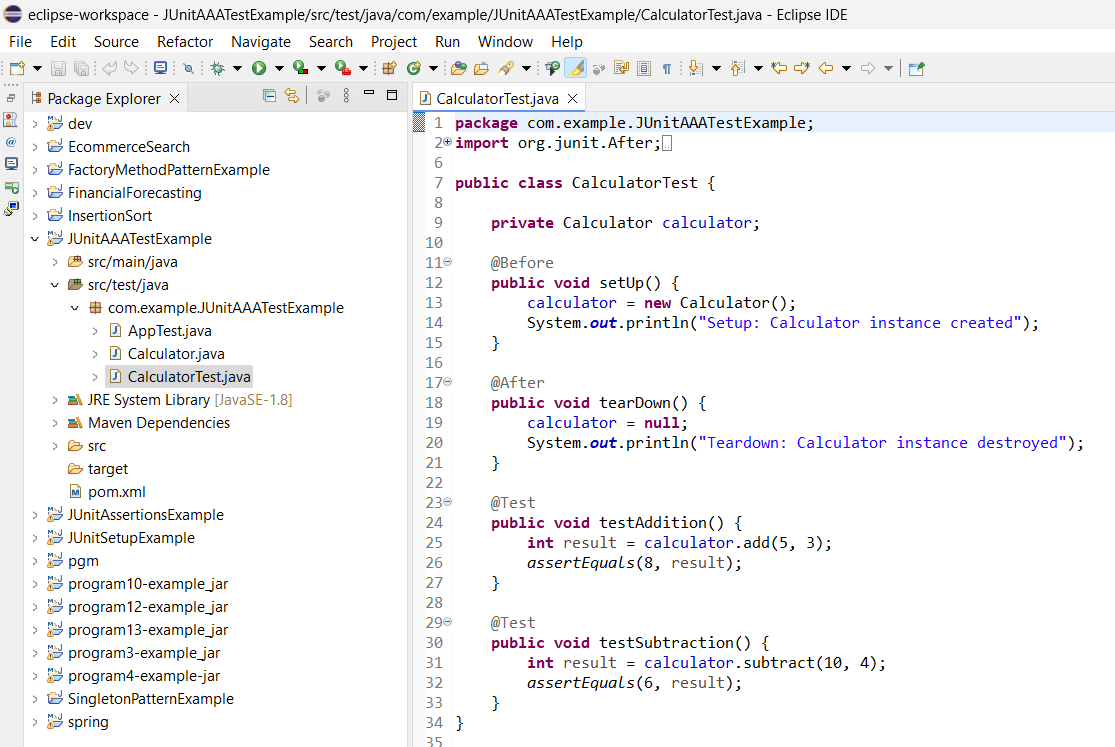
****

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

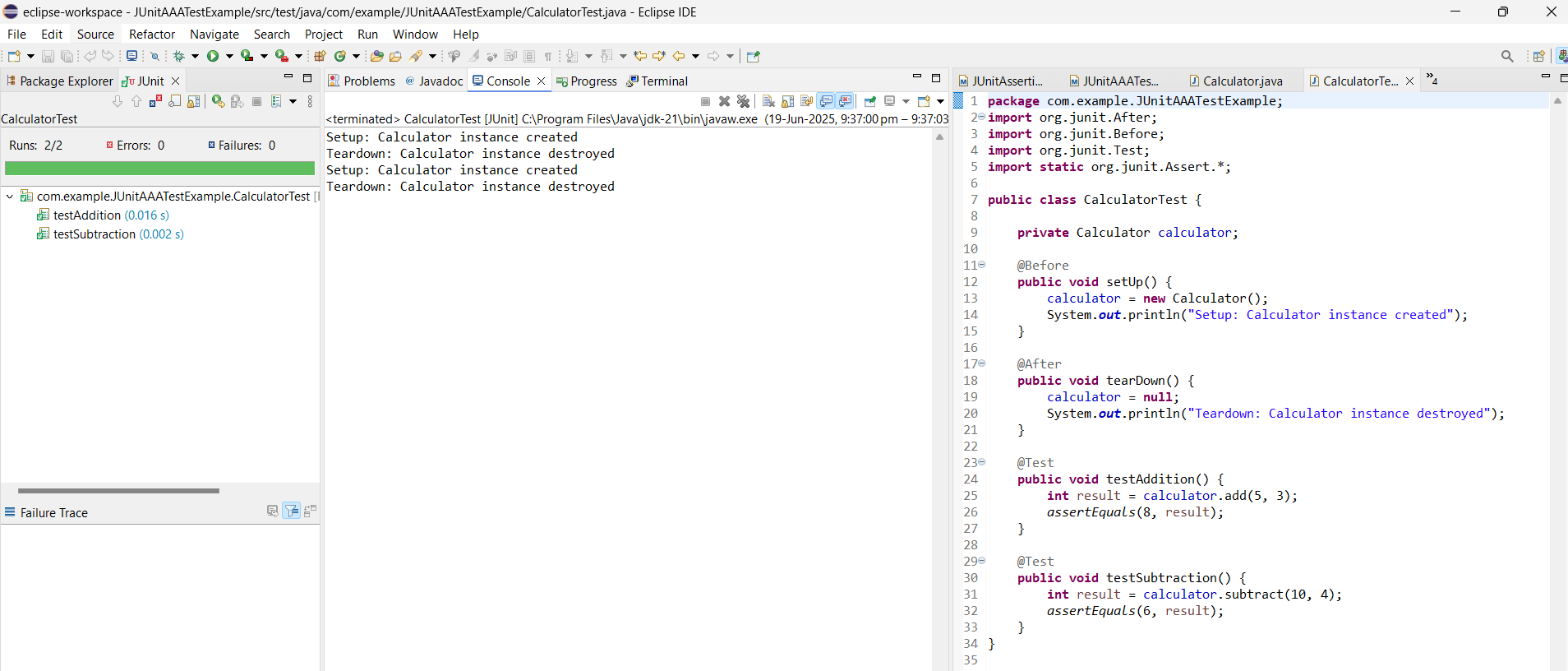
****

****

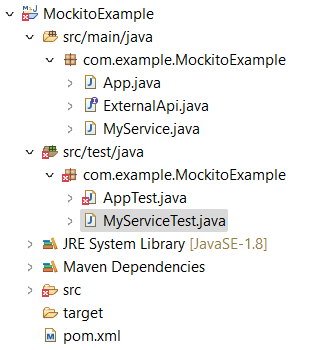
****

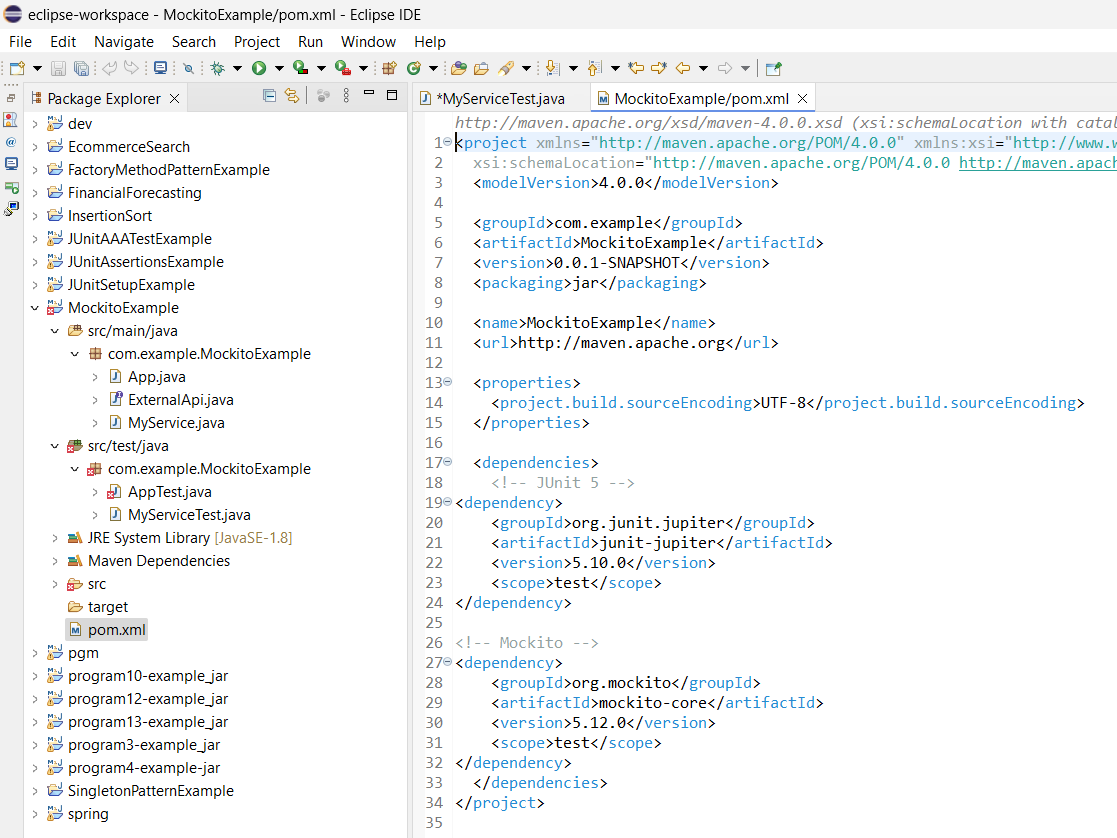
****

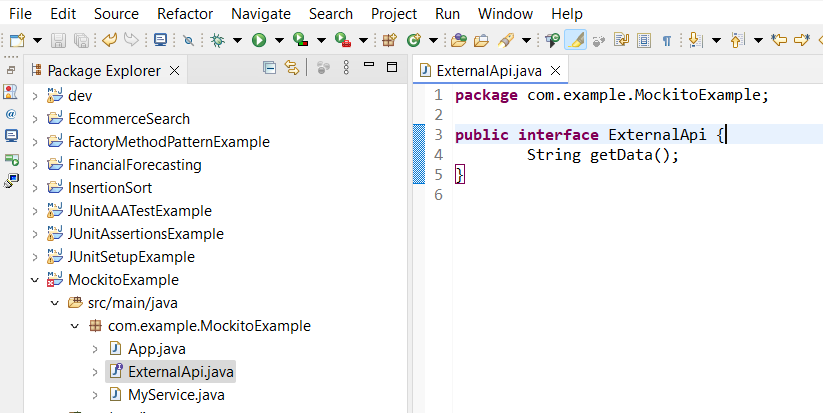
**OUTPUT:**

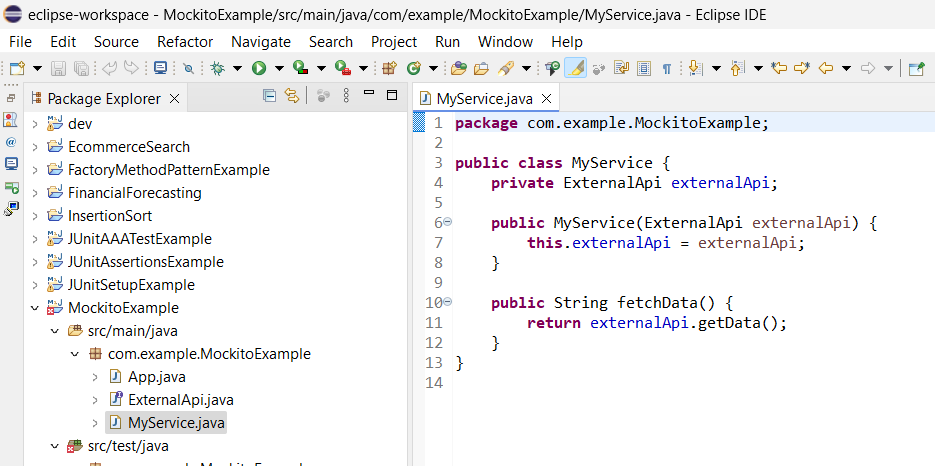
****

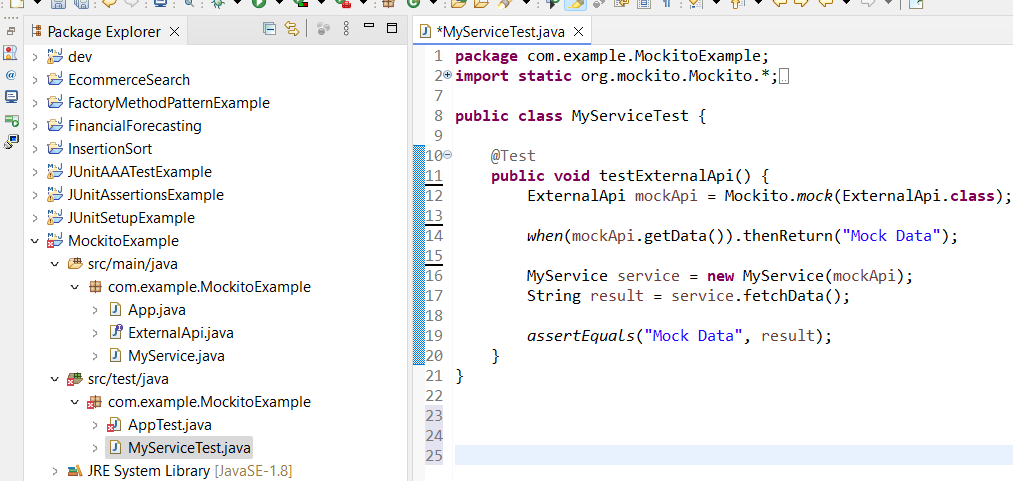
1. **Exercise 1: Mocking and Stubbing**

****

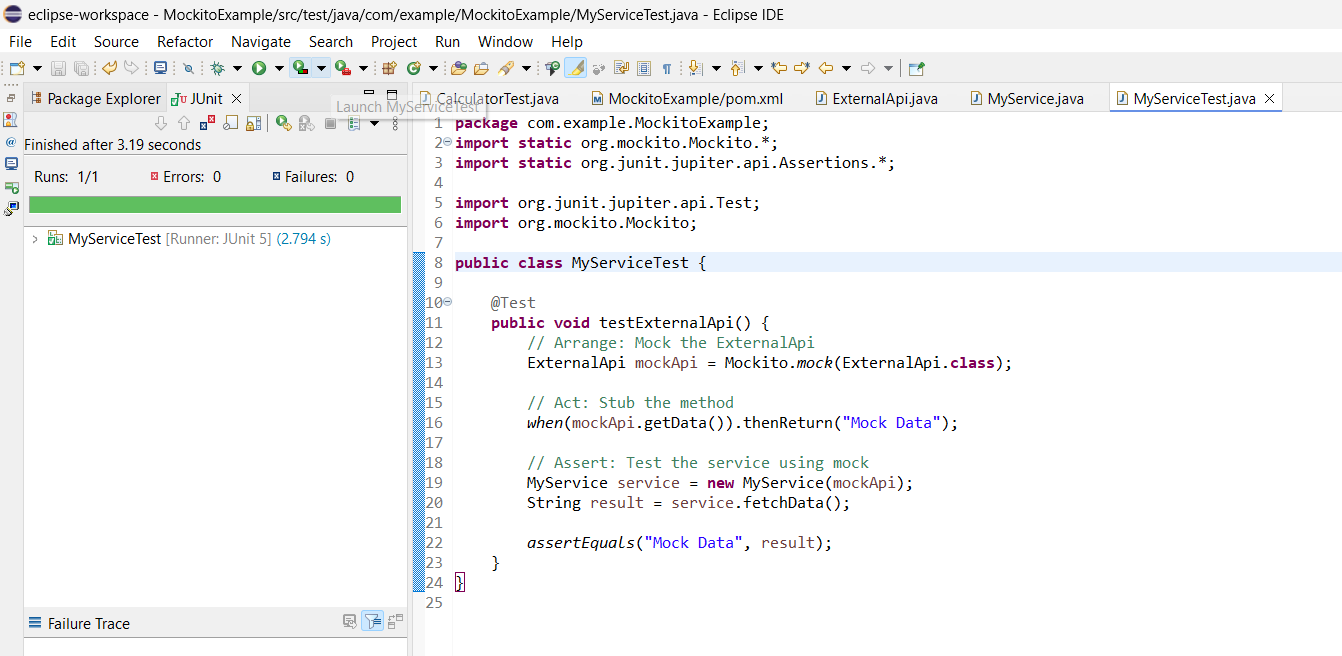
****

****

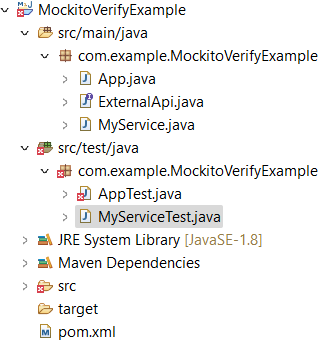
****

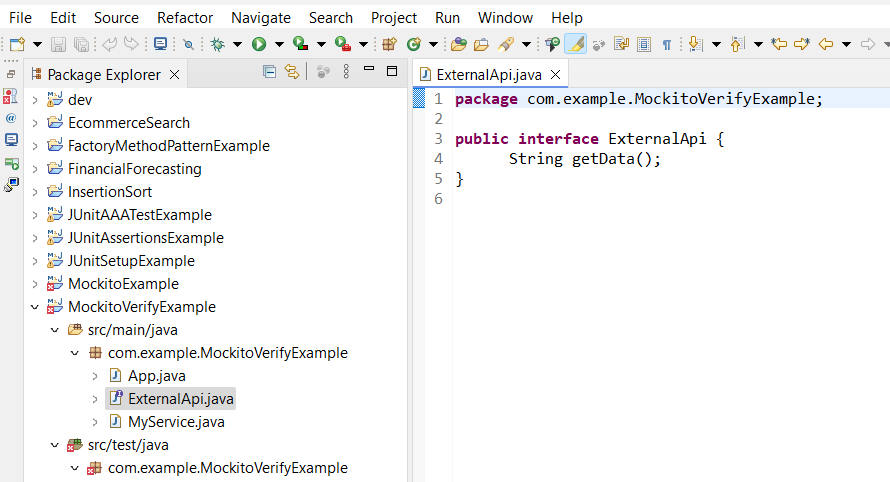
****

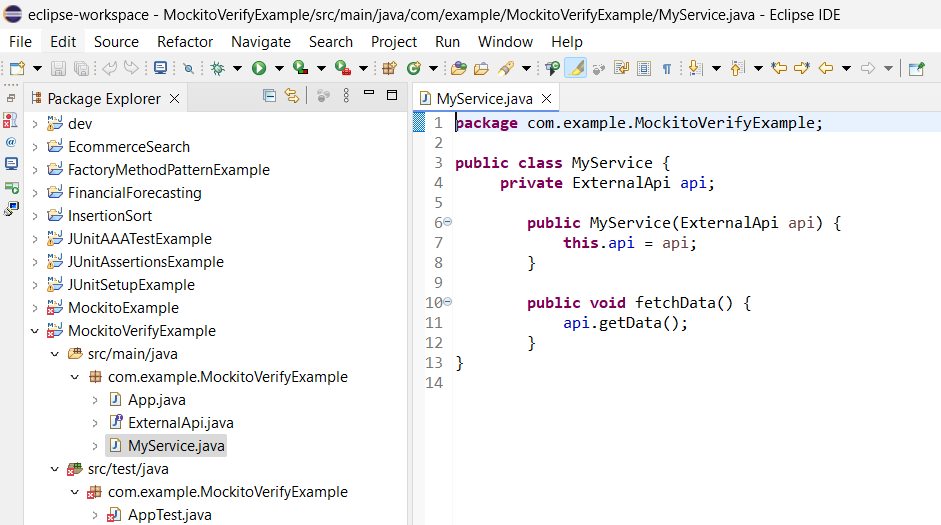
**OUTPUT:**

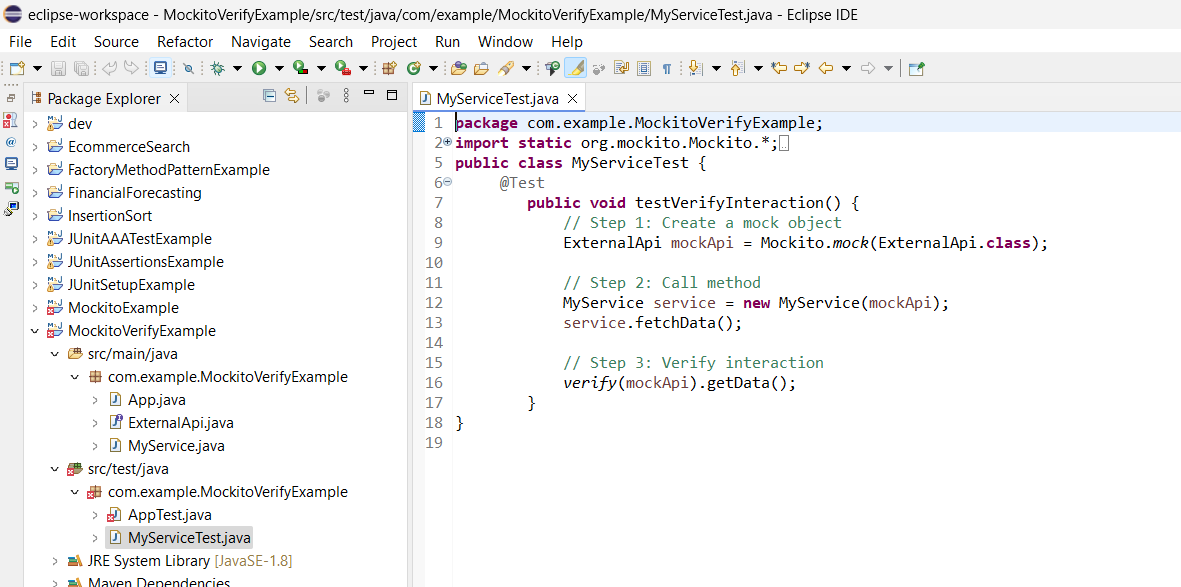
****

1. **Exercise 2: Verifying Interactions**

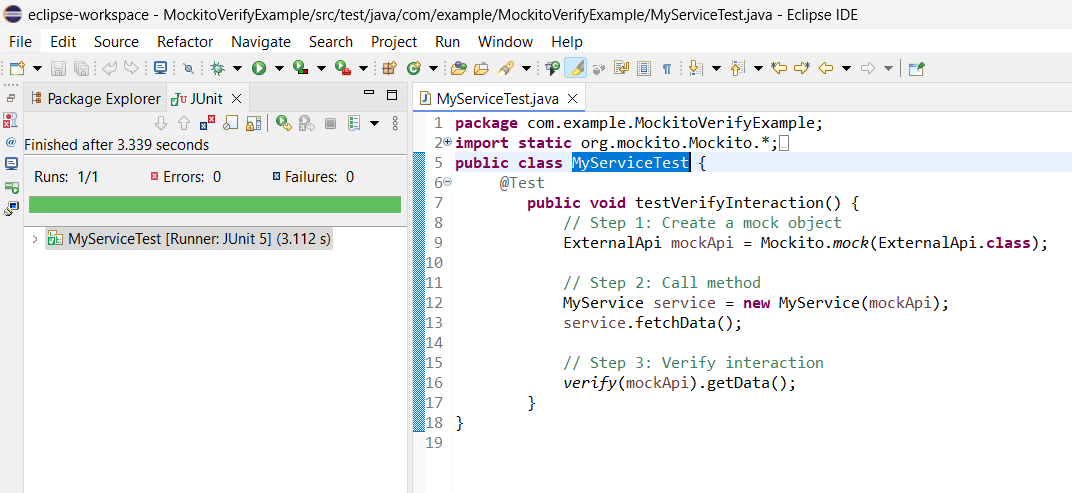
****

****

****

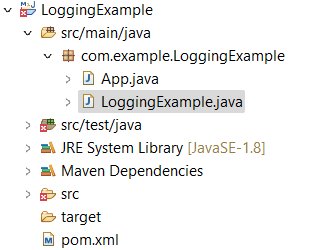
****

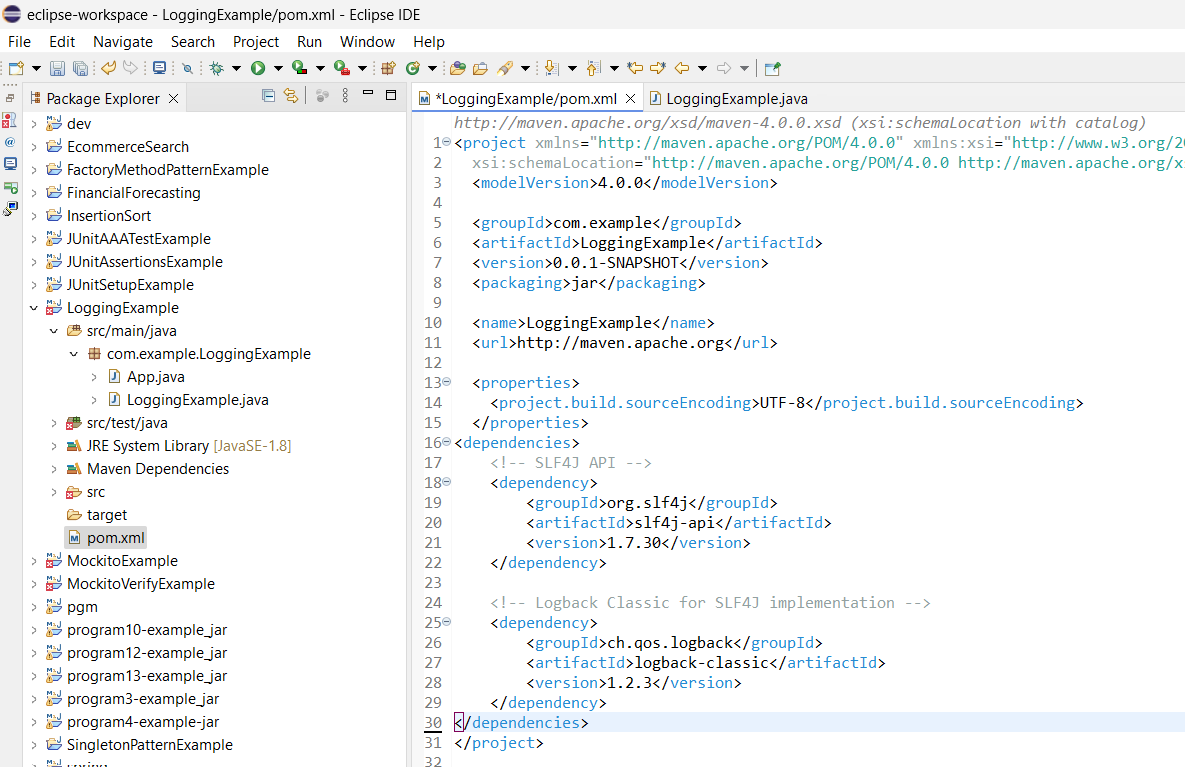
**OUTPUT:**

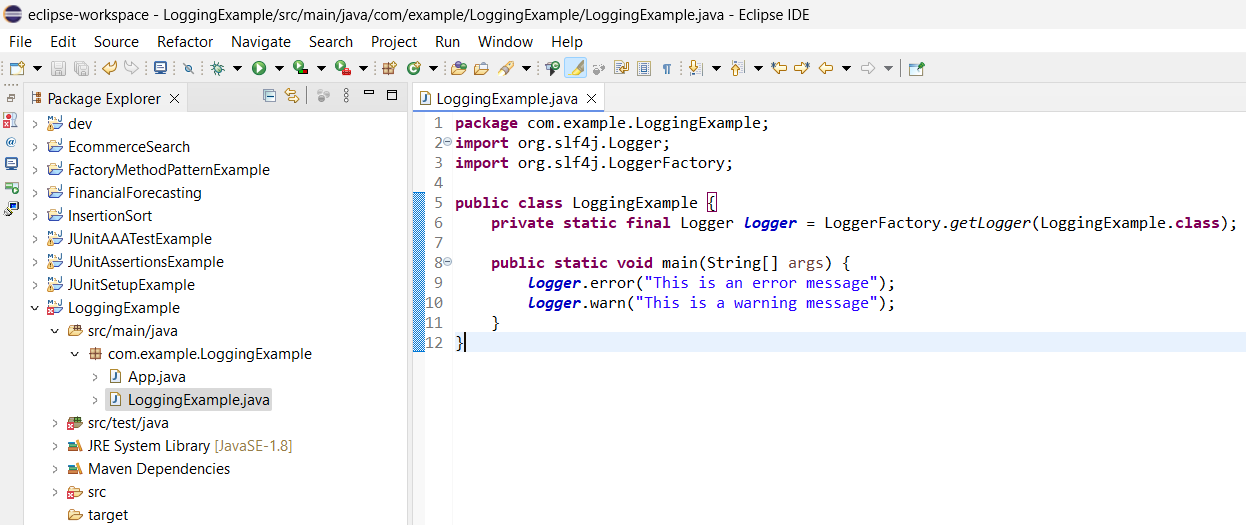
****

**SLF4J logging framework**

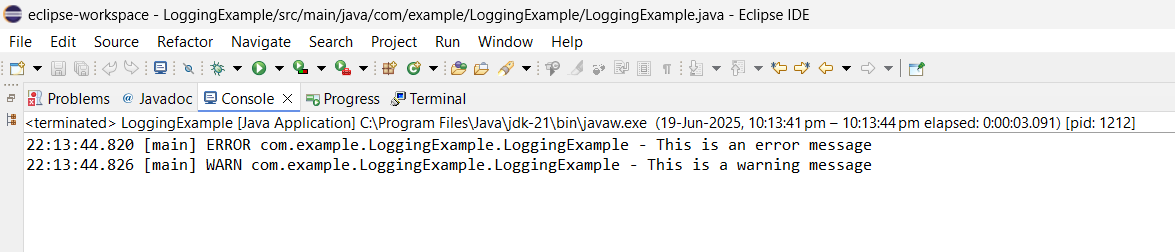
1. **Exercise 1: Logging Error Messages and Warning Levels**

****

****

****

**OUTPUT:**

****