**DIGITAL NURTURE 4.0 JAVA -FSE**

**Submitted By:**

**THAMIMUL ANSARI M**

**Superset ID: 6400989**

**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.

**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.

**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.

**Schema to be Created**

*CREATE TABLE Customers (*

*CustomerID NUMBER PRIMARY KEY,*

*Name VARCHAR2(100),*

*DOB DATE,*

*Balance NUMBER,*

*LastModified DATE*

*);*

*CREATE TABLE Loans (*

*LoanID NUMBER PRIMARY KEY,*

*CustomerID NUMBER,*

*LoanAmount NUMBER,*

*InterestRate NUMBER,*

*StartDate DATE,*

*EndDate DATE,*

*FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)*

*);*

**Example Scripts for Sample Data Insertion**

*INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)*

*VALUES (1, 'John Doe', TO\_DATE('1985-05-15', 'YYYY-MM-DD'), 1000, SYSDATE);*

*INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)*

*VALUES (2, 'Jane Smith', TO\_DATE('1990-07-20', 'YYYY-MM-DD'), 1500, SYSDATE);*

*INSERT INTO Loans (LoanID, CustomerID, LoanAmount, InterestRate, StartDate, EndDate)*

*VALUES (1, 1, 5000, 5, SYSDATE, ADD\_MONTHS(SYSDATE, 60));*

**-- Exercise 1: Control Structures**

**-- Schema to be Created**

**CREATE TABLE Customers (**

**CustomerID NUMBER PRIMARY KEY,**

**Name VARCHAR2(100),**

**DOB DATE,**

**Balance NUMBER,**

**LastModified DATE**

**);**

**CREATE TABLE Loans (**

**LoanID NUMBER PRIMARY KEY,**

**CustomerID NUMBER,**

**LoanAmount NUMBER,**

**InterestRate NUMBER,**

**StartDate DATE,**

**EndDate DATE,**

**FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)**

**);**

**-- Example Scripts for Sample Data Insertion**

**INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)**

**VALUES (1, 'John Doe', TO\_DATE('1985-05-15', 'YYYY-MM-DD'), 1000, SYSDATE);**

**INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)**

**VALUES (2, 'Jane Smith', TO\_DATE('1990-07-20', 'YYYY-MM-DD'), 1500, SYSDATE);**

**INSERT INTO Loans (LoanID, CustomerID, LoanAmount, InterestRate, StartDate, EndDate)**

**VALUES (1, 1, 5000, 5, SYSDATE, ADD\_MONTHS(SYSDATE, 60));**

**-- Scenario 1 – Senior Loan Discount**

**DECLARE**

**CURSOR cur\_customer IS**

**SELECT customerid, EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM dob) AS age**

**FROM customers;**

**v\_id customers.customerid%TYPE;**

**v\_age NUMBER;**

**BEGIN**

**FOR cust IN cur\_customer LOOP**

**v\_id := cust.customerid;**

**v\_age := cust.age;**

**IF v\_age > 60 THEN**

**UPDATE loans**

**SET interestrate = interestrate - 1**

**WHERE customerid = v\_id;**

**ELSE**

**DBMS\_OUTPUT.PUT\_LINE('Customer ID: ' || v\_id || ' | Age: ' || v\_age || ' | No discount');**

**END IF;**

**END LOOP;**

**COMMIT;**

**END;**

**/**

**-- Scenario 2 – Mark VIP Customers**

**ALTER TABLE customers ADD isvip VARCHAR2(10);**

**DECLARE**

**CURSOR cur\_vip IS**

**SELECT customerid, balance FROM customers;**

**v\_id customers.customerid%TYPE;**

**v\_bal customers.balance%TYPE;**

**BEGIN**

**FOR user IN cur\_vip LOOP**

**v\_id := user.customerid;**

**v\_bal := user.balance;**

**IF v\_bal > 10000 THEN**

**UPDATE customers SET isvip = 'TRUE' WHERE customerid = v\_id;**

**DBMS\_OUTPUT.PUT\_LINE('Marked Customer ' || v\_id || ' as VIP. Balance is Rs. ' || v\_bal);**

**ELSE**

**UPDATE customers SET isvip = 'FALSE' WHERE customerid = v\_id;**

**DBMS\_OUTPUT.PUT\_LINE('Customer ' || v\_id || ' not eligible for VIP. Current balance: Rs. ' || v\_bal);**

**END IF;**

**END LOOP;**

**COMMIT;**

**END;**

**/**

**-- Scenario 3 – Loan Due Reminders**

**DECLARE**

**CURSOR cur\_loans IS**

**SELECT l.loanid, l.customerid, c.name, l.enddate**

**FROM loans l**

**JOIN customers c ON c.customerid = l.customerid**

**WHERE l.enddate BETWEEN SYSDATE AND SYSDATE + 30;**

**v\_loanid loans.loanid%TYPE;**

**v\_custid loans.customerid%TYPE;**

**v\_name customers.name%TYPE;**

**v\_due loans.enddate%TYPE;**

**found BOOLEAN := FALSE;**

**BEGIN**

**OPEN cur\_loans;**

**LOOP**

**FETCH cur\_loans INTO v\_loanid, v\_custid, v\_name, v\_due;**

**EXIT WHEN cur\_loans%NOTFOUND;**

**found := TRUE;**

**DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ' || v\_loanid || ' for ' || v\_name || ' (ID: ' || v\_custid || ') is due on ' || TO\_CHAR(v\_due, 'DD-Mon-YYYY'));**

**END LOOP;**

**CLOSE cur\_loans;**

**IF NOT found THEN**

**DBMS\_OUTPUT.PUT\_LINE('No loans are nearing due date in next 30 days.');**

**END IF;**

**END;**

**/**

**1. JUnit\_Basic Testing Exercises**

### ✅ Exercise 1: Setting Up JUnit

**Files Created:**

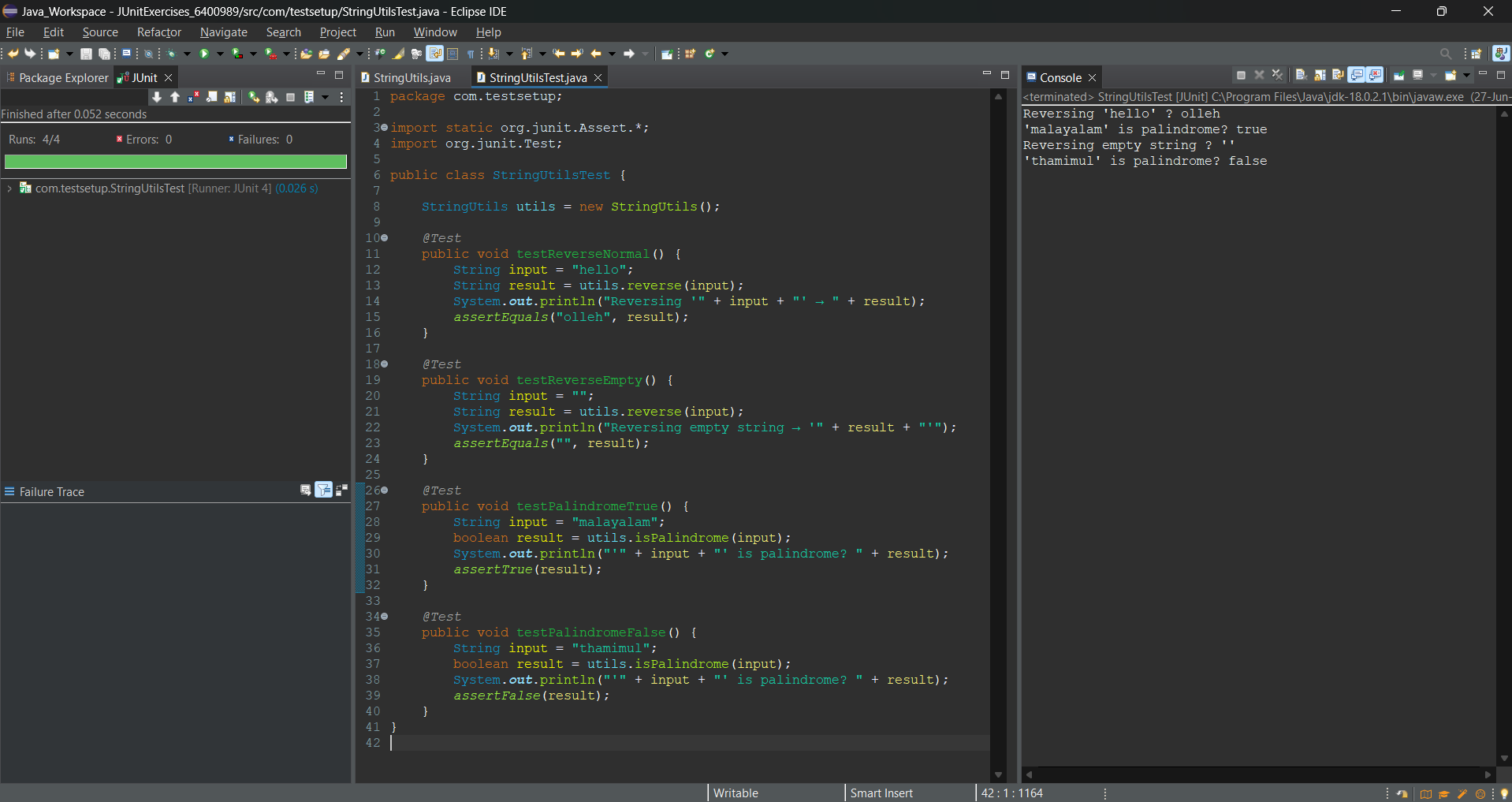
**StringUtils.java**

package com.testsetup;  
  
public class StringUtils {  
 public String reverse(String input) {  
 return new StringBuilder(input).reverse().toString();  
 }  
  
 public boolean isPalindrome(String input) {  
 String reversed = reverse(input);  
 return input.equalsIgnoreCase(reversed);  
 }  
}

**StringUtilsTest.java**

package com.testsetup;  
  
import static org.junit.Assert.\*;  
import org.junit.Test;  
  
public class StringUtilsTest {  
  
 StringUtils utils = new StringUtils();  
  
 @Test  
 public void testReverseNormal() {  
 String input = "hello";  
 String result = utils.reverse(input);  
 System.out.println("Reversing '" + input + "' → " + result);  
 assertEquals("olleh", result);  
 }  
  
 @Test  
 public void testReverseEmpty() {  
 String input = "";  
 String result = utils.reverse(input);  
 System.out.println("Reversing empty string → '" + result + "'");  
 assertEquals("", result);  
 }  
  
 @Test  
 public void testPalindromeTrue() {  
 String input = "malayalam";  
 boolean result = utils.isPalindrome(input);  
 System.out.println("'" + input + "' is palindrome? " + result);  
 assertTrue(result);  
 }  
  
 @Test  
 public void testPalindromeFalse() {  
 String input = "thamimul";  
 boolean result = utils.isPalindrome(input);  
 System.out.println("'" + input + "' is palindrome? " + result);  
 assertFalse(result);  
 }  
}

**Output Screenshot:**

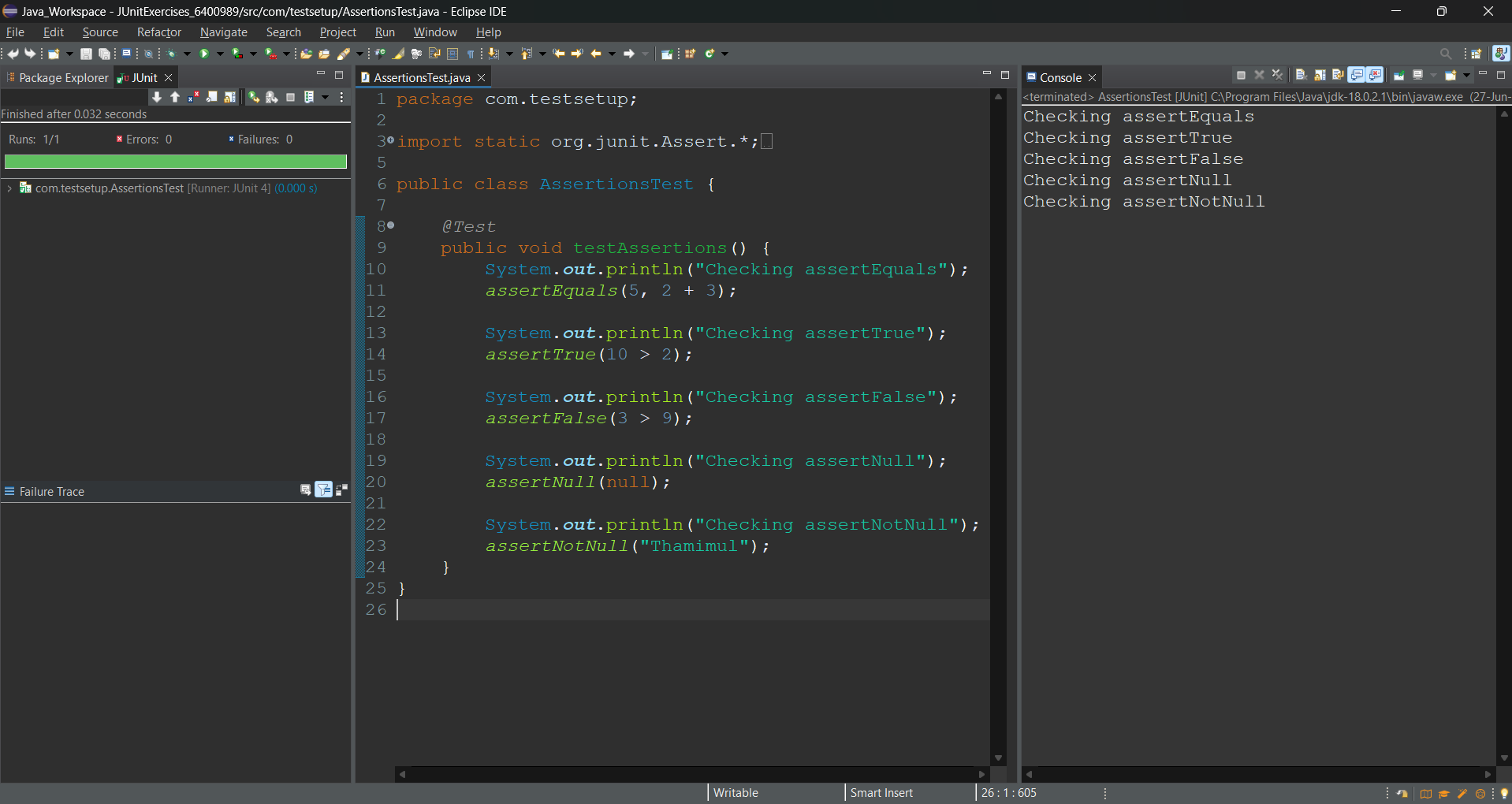


### ✅ Exercise 3: Assertions in JUnit

**AssertionsTest.java**

package com.testsetup;  
  
import static org.junit.Assert.\*;  
import org.junit.Test;  
  
public class AssertionsTest {  
 @Test  
 public void testAssertions() {  
 System.out.println("Checking assertEquals");  
 assertEquals(5, 2 + 3);  
  
 System.out.println("Checking assertTrue");  
 assertTrue(10 > 2);  
  
 System.out.println("Checking assertFalse");  
 assertFalse(3 > 9);  
  
 System.out.println("Checking assertNull");  
 assertNull(null);  
  
 System.out.println("Checking assertNotNull");  
 assertNotNull("Thamimul");  
 }  
}

**Output Screenshot:**



### 

### ✅ Exercise 4: AAA Pattern, Setup and Teardown (LoginService - Unique)

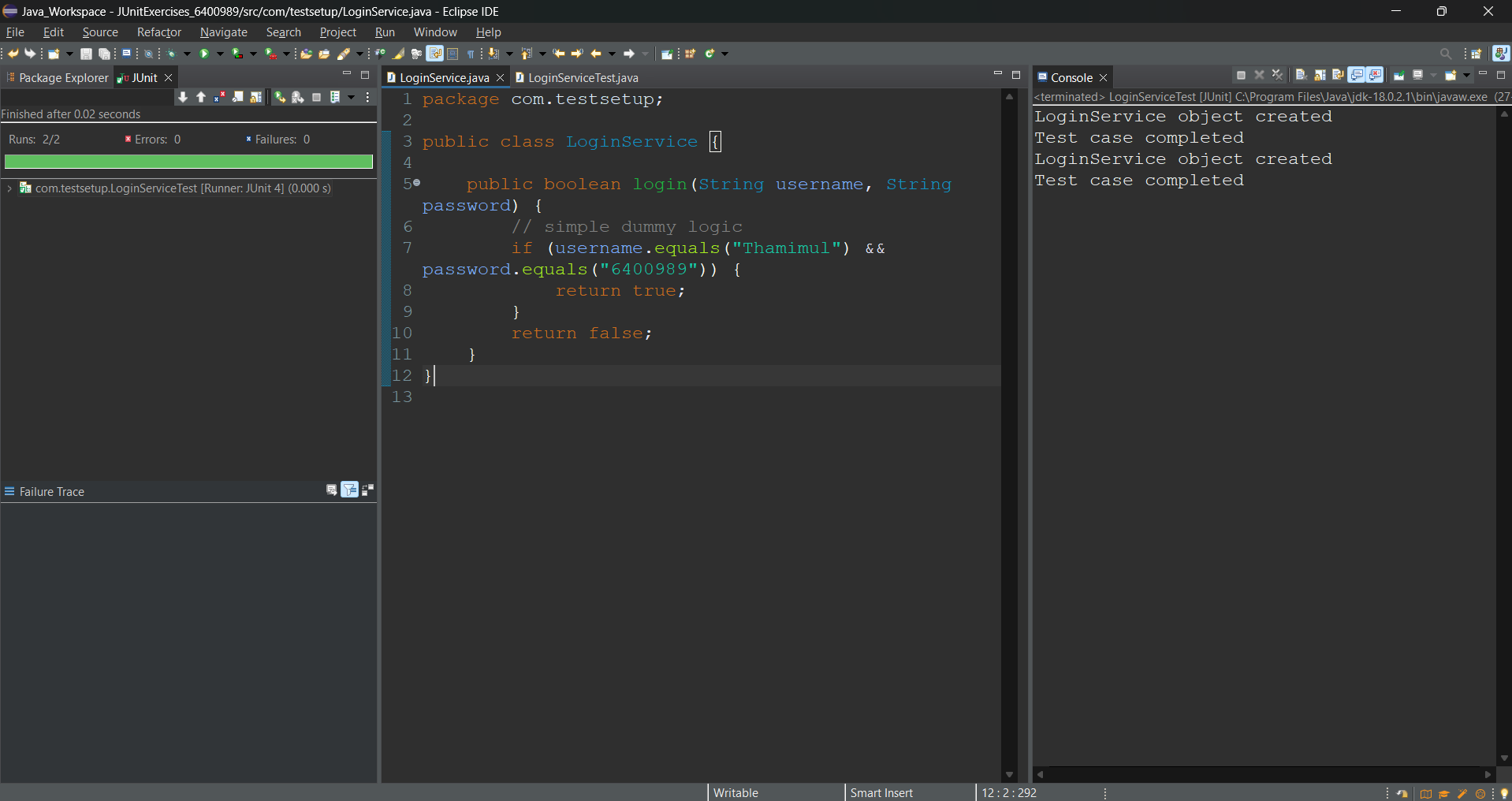
**LoginService.java**

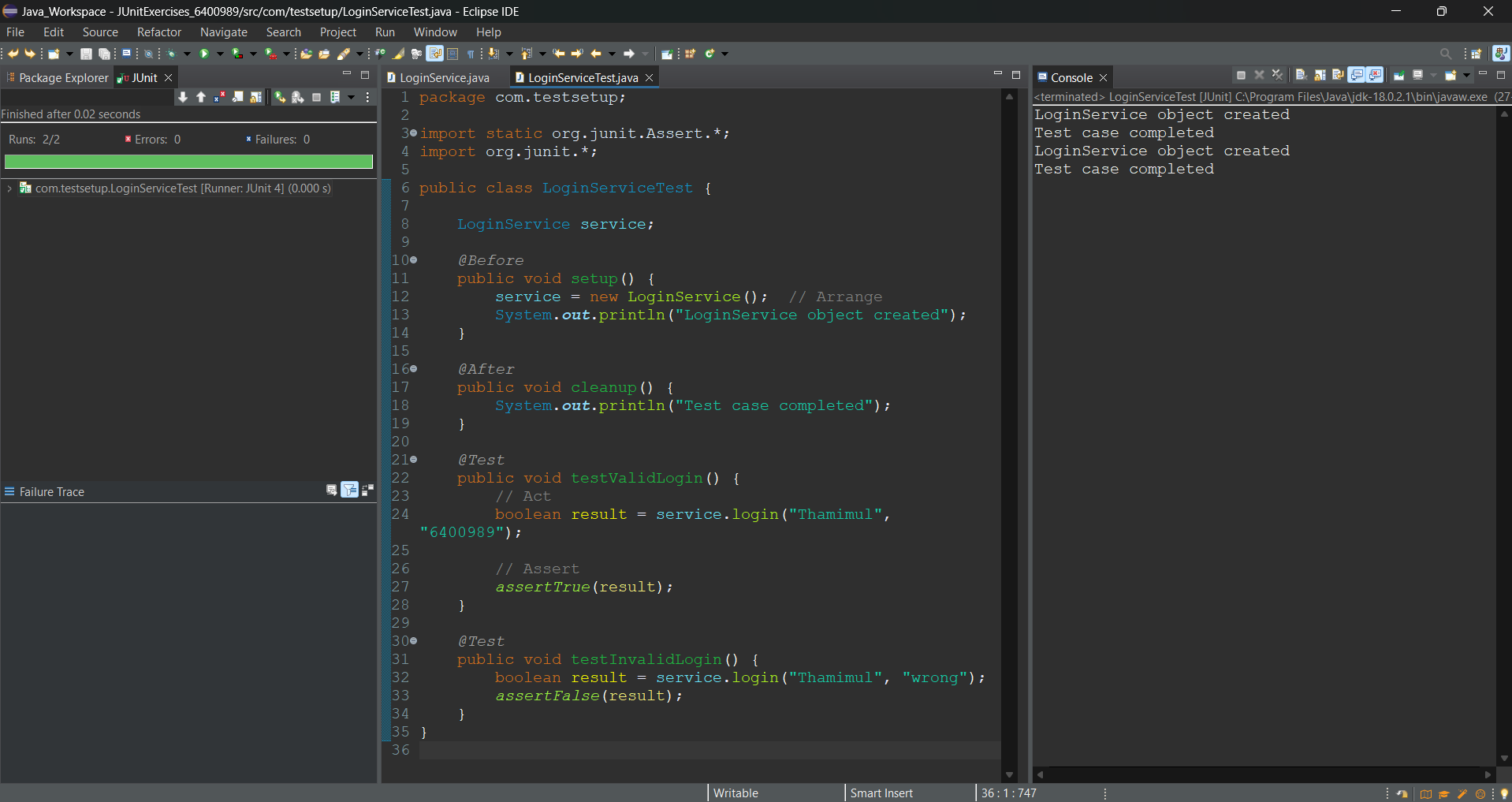
package com.testsetup;  
  
public class LoginService {  
 public boolean login(String username, String password) {  
 return username.equals("Thamimul") && password.equals("6400989");  
 }  
}

**LoginServiceTest.java**

package com.testsetup;  
  
import static org.junit.Assert.\*;  
import org.junit.\*;  
  
public class LoginServiceTest {  
  
 LoginService service;  
  
 @Before  
 public void setup() {  
 service = new LoginService();  
 System.out.println("LoginService object created");  
 }  
  
 @After  
 public void cleanup() {  
 System.out.println("Test case completed");  
 }  
  
 @Test  
 public void testValidLogin() {  
 boolean result = service.login("Thamimul", "6400989");  
 assertTrue(result);  
 }  
  
 @Test  
 public void testInvalidLogin() {  
 boolean result = service.login("user", "wrong");  
 assertFalse(result);  
 }  
}

**Output Screenshot:**





**3. Mockito exercises**

### ✅ Exercise 1: Mocking and Stubbing

**MyService.java**

package com.testsetup;  
  
public class MyService {  
 ExternalApi api;  
  
 public MyService(ExternalApi api) {  
 this.api = api;  
 }  
  
 public String fetchData() {  
 return api.getData();  
 }  
}

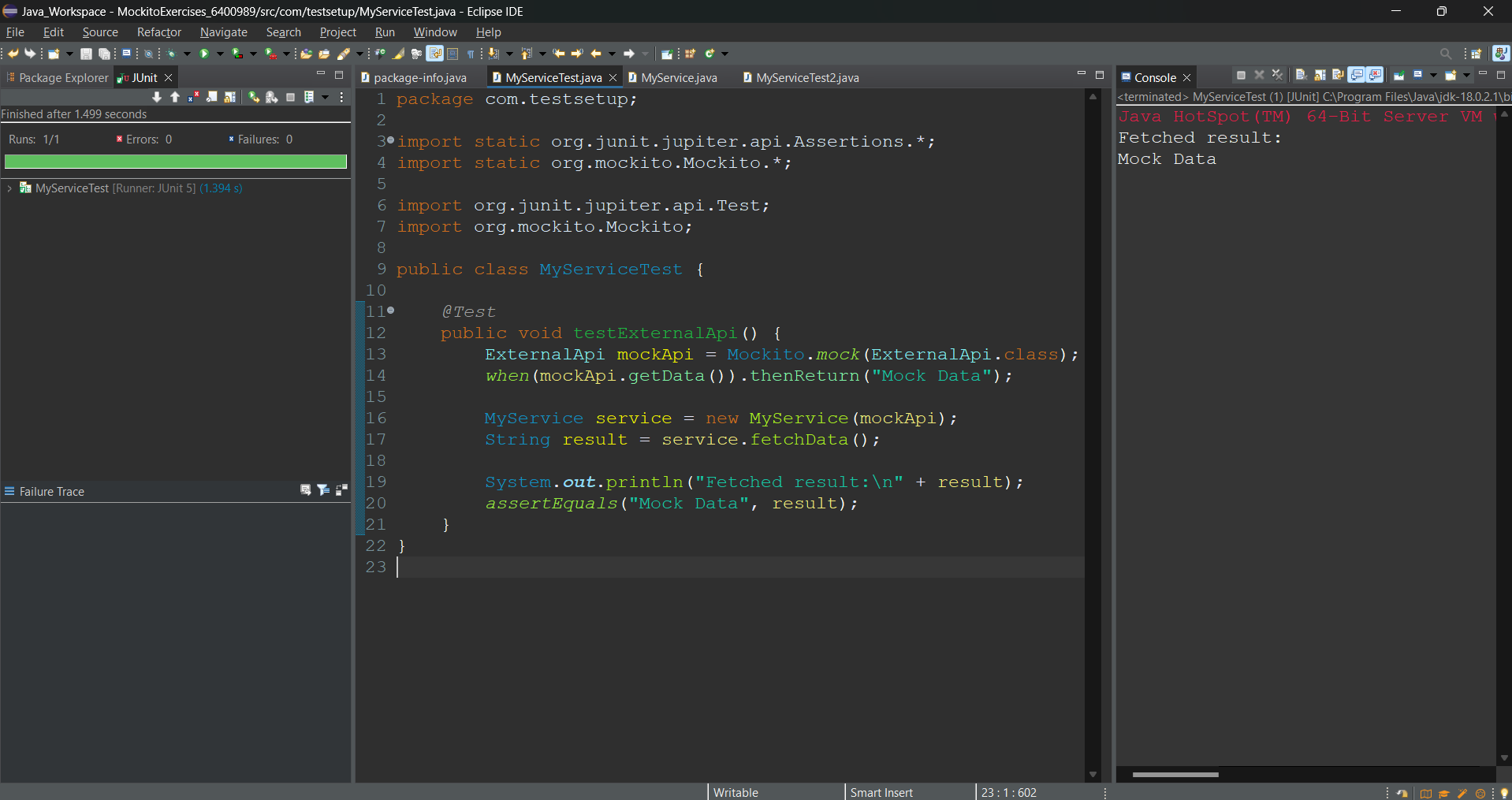
**ExternalApi.java**

package com.testsetup;  
  
public interface ExternalApi {  
 String getData();  
}

**MyServiceTest.java (Mocking Test)**

package com.testsetup;  
  
import static org.junit.jupiter.api.Assertions.\*;  
import static org.mockito.Mockito.\*;  
  
import org.junit.jupiter.api.Test;  
import org.mockito.Mockito;  
  
public class MyServiceTest {  
  
 @Test  
 public void testExternalApi() {  
 ExternalApi mockApi = Mockito.mock(ExternalApi.class);  
 when(mockApi.getData()).thenReturn("Mock Data");  
  
 MyService service = new MyService(mockApi);  
 String result = service.fetchData();  
  
 System.out.println("Fetched result:\n" + result);  
 assertEquals("Mock Data", result);  
 }  
}

**Output Screenshot:**

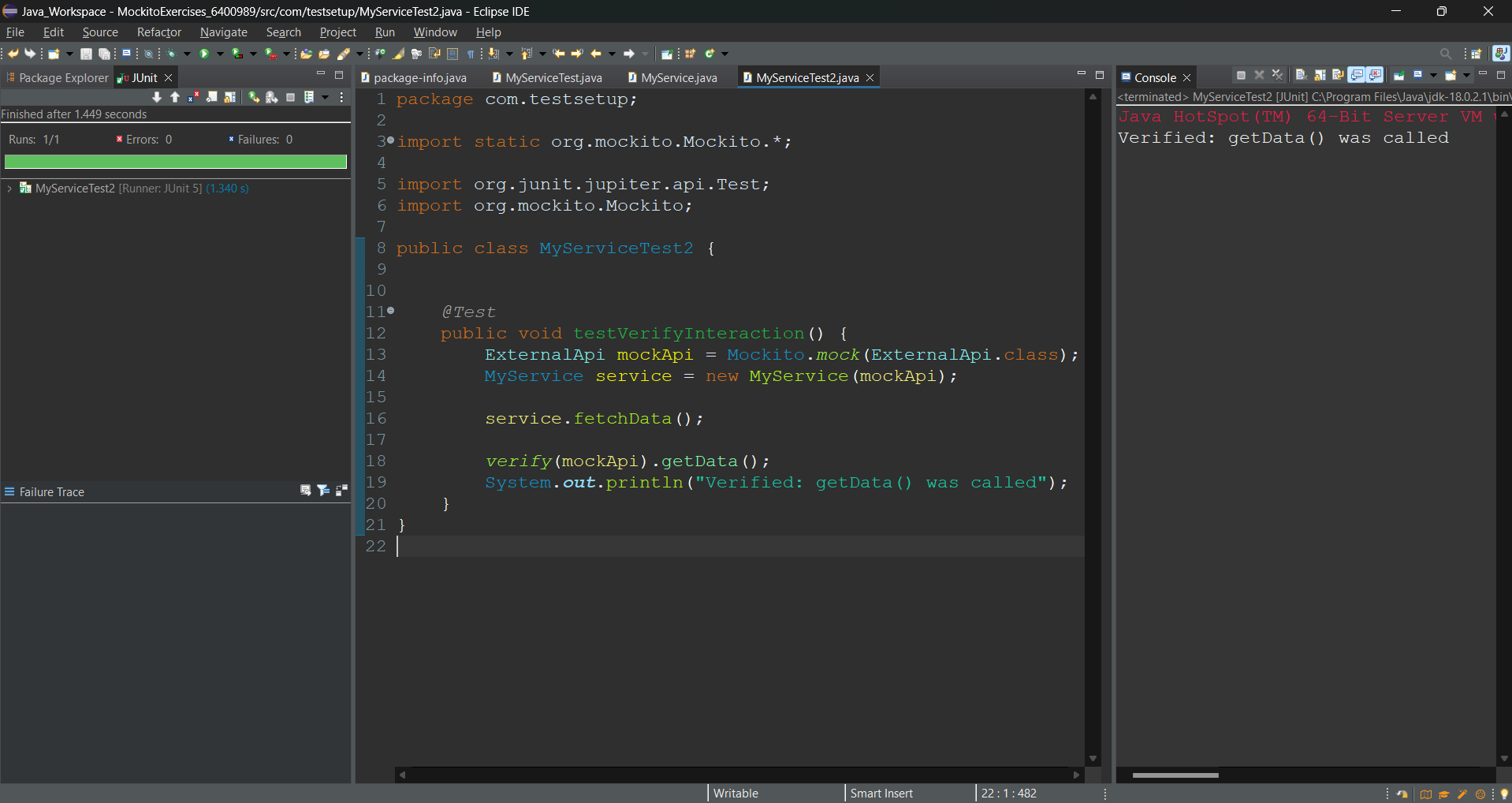


### ✅ Mockito Exercise 2: Verifying Interaction

**MyServiceTest.java (Verification Test)**

@Test  
public void testVerifyInteraction() {  
 ExternalApi mockApi = Mockito.mock(ExternalApi.class);  
 MyService service = new MyService(mockApi);  
  
 service.fetchData();  
  
 verify(mockApi).getData();  
 System.out.println("Verified: getData() was called");  
}

**Output Screenshot:**



### 

**6. SL4J Logging exercises**

### ✅ SLF4J Logging Exercise 1: Logging Error Messages and Warning Levels

**pom.xml (Dependencies)**

<dependencies>  
 <dependency>  
 <groupId>org.slf4j</groupId>  
 <artifactId>slf4j-api</artifactId>  
 <version>1.7.30</version>  
 </dependency>  
 <dependency>  
 <groupId>ch.qos.logback</groupId>  
 <artifactId>logback-classic</artifactId>  
 <version>1.2.3</version>  
 </dependency>  
</dependencies>

**LoggingExample.java**

package com.logdemo;  
  
import org.slf4j.Logger;  
import org.slf4j.LoggerFactory;  
  
public class LoggingExample {  
  
 private static final Logger logger = LoggerFactory.getLogger(LoggingExample.class);  
  
 public static void main(String[] args) {  
 logger.error("Error message using SL4J Logger");  
 logger.warn("Warning message using SL4J Logger");  
  
 System.out.println("Log messages sent. Check console.");  
 }  
}

**Output Screenshot:**

