**Day-13 Basics of Java**

Problem Statement 1: Setting Up, Writing, and Running Java Unit Tests

**//Product.java**

**public** **class** Product {

**private** **int** id;

**private** String name;

**private** **double** price;

**public** Product(String name, **double** price) {

**this**.name = name;

**this**.price = price;

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **double** getPrice() {

**return** price;

}

**public** **void** setPrice(**double** price) {

**this**.price = price;

}

@Override

**public** String toString() {

**return** "Product{" +

"id=" + id +

", name='" + name + '\'' +

", price=" + price +

'}';

}

}

//ProductDAO.Java

**import** java.sql.Connection;

**import** java.util.List;

**public** **class** ProductDAO {

**private** Connection connection;

**public** ProductDAO(Connection connection) {

**this**.connection = connection;

}

**public** **boolean** addProduct(Product product) {

**return** **false**;

}

**public** Product getProduct(**int** id) {

**return** **null**;

}

**public** List<Product> getAllProducts() {

**return** **null**;

}

**public** **boolean** updateProduct(Product product) {

**return** **false**;

}

**public** **boolean** deleteProduct(**int** id) {

**return** **false**;

}

}

//DBUtil.Java

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.SQLException;

**public** **class** DBUtil {

**private** **static** **final** String ***URL*** = "jdbc: mysql://localhost:3306/library ";

**private** **static** **final** String ***USER*** = "root”;

**private** **static** **final** String ***PASSWORD*** = "Sowmya";

**public** **static** Connection getConnection() **throws** SQLException {

**return** DriverManager.*getConnection*(***URL***, ***USER***, ***PASSWORD***);

}

**public** **static** **void** closeConnection(Connection connection) {

**if** (connection != **null**) {

**try** {

connection.close();

} **catch** (SQLException e) {

e.printStackTrace();

}

}

}

}

//ProductManagmentTest.Java

**import** org.junit.jupiter.api.AfterAll;

**import** org.junit.jupiter.api.BeforeAll;

**import** org.junit.jupiter.api.BeforeEach;

**import** org.junit.jupiter.api.Test;

**import** java.sql.Connection;

**import** java.sql.SQLException;

**import** java.util.List;

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

**public** **class** ProductManagementTest {

**private** **static** Connection *connection*;

**private** ProductDAO productDAO;

@BeforeAll

**public** **static** **void** init() **throws** SQLException {

*connection* = DBUtil.*getConnection*();

}

@BeforeEach

**public** **void** beforeEachTest() {

productDAO = **new** ProductDAO(*connection*);

}

@Test

**public** **void** addProductTest() {

Product product = **new** Product("TestProduct", 100.0);

**boolean** result = productDAO.addProduct(product);

Product retrievedProduct = productDAO.getProduct(product.getId());

*assertTrue*(result);

*assertNotNull*(retrievedProduct);

*assertEquals*(product.getName(), retrievedProduct.getName());

*assertEquals*(product.getPrice(), retrievedProduct.getPrice());

}

@Test

**public** **void** getAllProductsTest() {

Product product1 = **new** Product("Product1", 50.0);

Product product2 = **new** Product("Product2", 75.0);

productDAO.addProduct(product1);

productDAO.addProduct(product2);

List<Product> products = productDAO.getAllProducts();

*assertNotNull*(products);

*assertTrue*(products.size() >= 2);

}

@Test

**public** **void** updateProductTest() {

Product product = **new** Product("OldProduct", 100.0);

productDAO.addProduct(product);

product.setName("UpdatedProduct");

product.setPrice(150.0);

**boolean** result = productDAO.updateProduct(product);

Product updatedProduct = productDAO.getProduct(product.getId());

*assertTrue*(result);

*assertNotNull*(updatedProduct);

*assertEquals*("UpdatedProduct", updatedProduct.getName());

*assertEquals*(150.0, updatedProduct.getPrice());

}

@Test

**public** **void** deleteProductTest() {

Product product = **new** Product("ProductToDelete", 200.0);

productDAO.addProduct(product);

**boolean** result = productDAO.deleteProduct(product.getId());

Product deletedProduct = productDAO.getProduct(product.getId());

*assertTrue*(result);

*assertNull*(deletedProduct);

}

@AfterAll

**public** **static** **void** destroy() {

DBUtil.*closeConnection*(*connection*);

}

}