NAME	Prakash Joshi
UID	23BCS12438
CLASS	622-A

Experiment – 6 (Part – b) :

Title

CRUD Operations on Product Table Using JDBC.

Objective

To create a menu-driven Java program that performs **CRUD operations** (Create, Read, Update, Delete) on a Product table in a MySQL database with proper transaction handling.

Task Description

This program requires the use of JDBC to:

- Connect to a database and manage a table named Product with columns:
  - ProductID
  - ProductName
  - Price

  - Quantity
- Provide a menu to perform the following:
  - **Create**: Insert a new product record.
  - **Read**: Display all existing product records.
  - **Update**: Modify product details using the ProductID.
  - **Delete**: Remove a product by its ID.
- Include **transaction handling** using setAutoCommit(false) and commit()/rollback() to ensure data integrity and consistency, especially for update and delete operations.

This question tests your understanding of:

- Writing and executing SQL statements.
- Using PreparedStatement for safe and parameterized queries.
- Managing multiple database operations in a secure and consistent way.

Code :

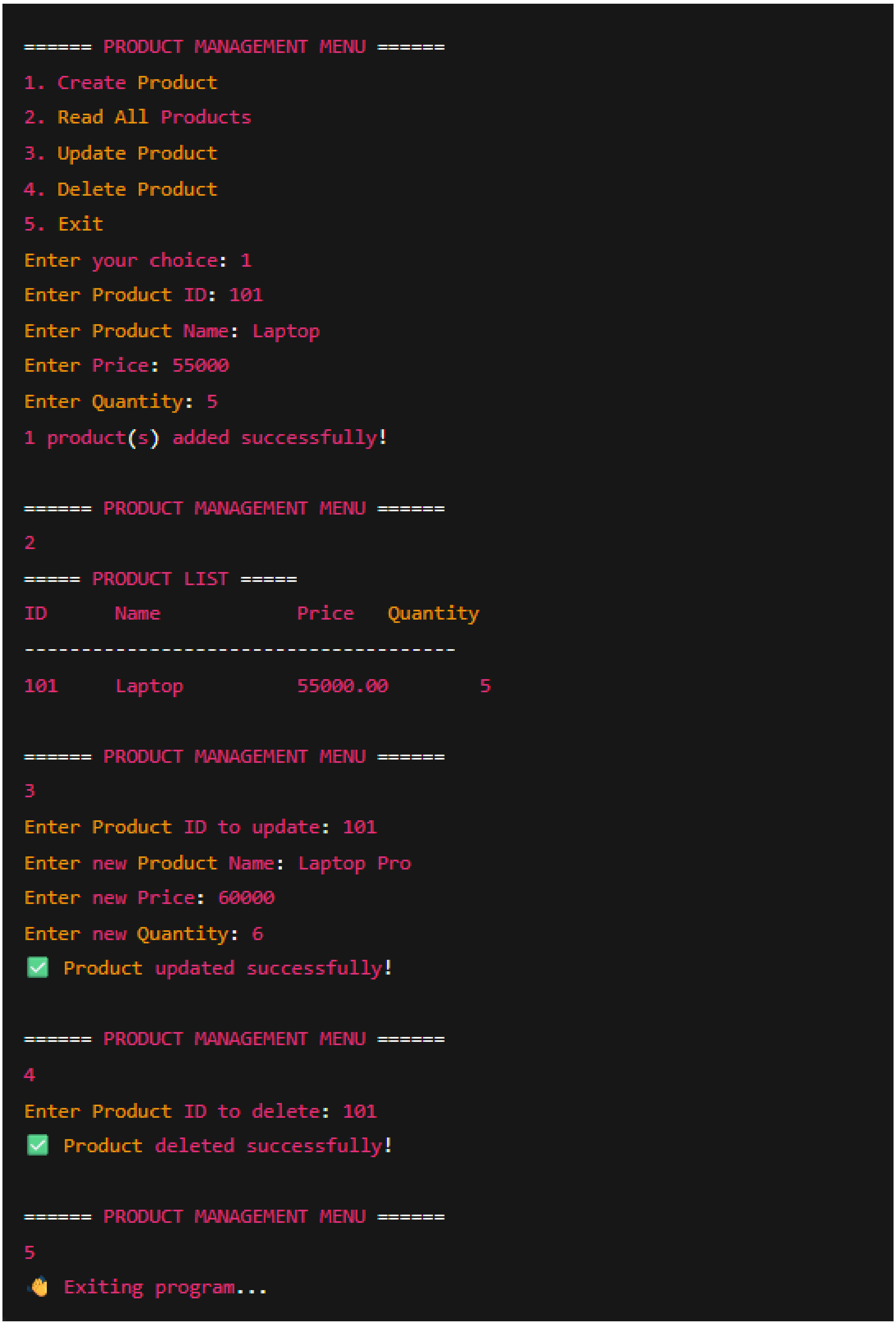
```
import java.sql.*;
import java.util.Scanner;

public class ProductCRUD {
  public static void main(String[] args) {
    String url = "jdbc:mysql://localhost:3306/companydb"; // Database name
    String username = "root";
    String password = "your_password"; // replace with your MySQL password

    try (Connection con = DriverManager.getConnection(url, username, password);
       Scanner sc = new Scanner(System.in)) {

      // Load JDBC Driver
      Class.forName("com.mysql.cj.jdbc.Driver");
```

```
            System.out.println("✅ Connected to the database successfully!");

            int choice;

            do {
                System.out.println("\n====== PRODUCT MANAGEMENT MENU ======");
                System.out.println("1. Create Product");
                System.out.println("2. Read All Products");
                System.out.println("3. Update Product");
                System.out.println("4. Delete Product");
                System.out.println("5. Exit");
                System.out.print("Enter your choice: ");
                choice = sc.nextInt();
                sc.nextLine(); // consume newline

                switch (choice) {
                    case 1 -> createProduct(con, sc);
                    case 2 -> readProducts(con);
                    case 3 -> updateProduct(con, sc);
                    case 4 -> deleteProduct(con, sc);
                    case 5 -> System.out.println("👋 Exiting program...");
                    default -> System.out.println("❌ Invalid choice. Try again.");
                }
            } while (choice != 5);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    // ✅ CREATE operation
    private static void createProduct(Connection con, Scanner sc) throws
SQLException {
        System.out.print("Enter Product ID: ");
        int id = sc.nextInt();
        sc.nextLine();
        System.out.print("Enter Product Name: ");
        String name = sc.nextLine();
        System.out.print("Enter Price: ");
        double price = sc.nextDouble();
        System.out.print("Enter Quantity: ");
        int qty = sc.nextInt();

        String sql = "INSERT INTO Product (ProductID, ProductName, Price, Quantity)
VALUES (?, ?, ?, ?)";
        try (PreparedStatement ps = con.prepareStatement(sql)) {
            ps.setInt(1, id);
            ps.setString(2, name);
            ps.setDouble(3, price);
            ps.setInt(4, qty);
            int rows = ps.executeUpdate();
            System.out.println(rows + " product(s) added successfully!");
        }
    }

    // ✅ READ operation
    private static void readProducts(Connection con) throws SQLException {
        String sql = "SELECT * FROM Product";
        try (Statement stmt = con.createStatement();
             ResultSet rs = stmt.executeQuery(sql)) {

            System.out.println("\n===== PRODUCT LIST =====");
            System.out.println("ID\tName\t\tPrice\tQuantity");
            System.out.println("--------------------------------------");
            while (rs.next()) {
                System.out.printf("%d\t%-10s\t%.2f\t%d%n",
                        rs.getInt("ProductID"),
                        rs.getString("ProductName"),
                        rs.getDouble("Price"),
                        rs.getInt("Quantity"));
            }
        }
    }

    // ☑ UPDATE operation (with transaction)
    private static void updateProduct(Connection con, Scanner sc) throws SQLException {
        con.setAutoCommit(false); // start transaction
        try {
            System.out.print("Enter Product ID to update: ");
            int id = sc.nextInt();
            sc.nextLine();

            System.out.print("Enter new Product Name: ");
            String name = sc.nextLine();
            System.out.print("Enter new Price: ");
            double price = sc.nextDouble();
            System.out.print("Enter new Quantity: ");
            int qty = sc.nextInt();

            String sql = "UPDATE Product SET ProductName=?, Price=?, Quantity=? WHERE ProductID=?";
            try (PreparedStatement ps = con.prepareStatement(sql)) {
                ps.setString(1, name);
                ps.setDouble(2, price);
                ps.setInt(3, qty);
                ps.setInt(4, id);

                int rows = ps.executeUpdate();
                if (rows > 0) {
                    con.commit();
                    System.out.println("☑ Product updated successfully!");
                } else {
                    con.rollback();
                    System.out.println("❌ Product not found. Transaction rolled back.");
                }
            }
        } catch (SQLException e) {
            con.rollback();
            System.out.println("⚠ Update failed. Transaction rolled back.");
        } finally {
            con.setAutoCommit(true);
        }
    }

    // ☑ DELETE operation (with transaction)
    private static void deleteProduct(Connection con, Scanner sc) throws
SQLException {
        con.setAutoCommit(false); // start transaction
        try {
            System.out.print("Enter Product ID to delete: ");
            int id = sc.nextInt();

            String sql = "DELETE FROM Product WHERE ProductID=?";
            try (PreparedStatement ps = con.prepareStatement(sql)) {
                ps.setInt(1, id);
                int rows = ps.executeUpdate();
                if (rows > 0) {
                    con.commit();
                    System.out.println("☑ Product deleted successfully!");
                } else {
                    con.rollback();
                    System.out.println("❌ Product not found. Transaction rolled back.");
            }
          }
        } catch (SQLException e) {
          con.rollback();
          System.out.println("⚠ Deletion failed. Transaction rolled back.");
        } finally {
          con.setAutoCommit(true);
        }
    }
}
```

My SQL Table Sample :

```
CREATE DATABASE companydb;
USE companydb;

CREATE TABLE Product (
   ProductID INT PRIMARY KEY,
   ProductName VARCHAR(50),
   Price DOUBLE,
   Quantity INT
);
```

Output :



```
====== PRODUCT MANAGEMENT MENU ======
1. Create Product
2. Read All Products
3. Update Product
4. Delete Product
5. Exit
Enter your choice: 1
Enter Product ID: 101
Enter Product Name: Laptop
Enter Price: 55000
Enter Quantity: 5
1 product(s) added successfully!

====== PRODUCT MANAGEMENT MENU ======
2
===== PRODUCT LIST =====
ID      Name            Price   Quantity
--------------------------------------
101     Laptop          55000.00        5

====== PRODUCT MANAGEMENT MENU ======
3
Enter Product ID to update: 101
Enter new Product Name: Laptop Pro
Enter new Price: 60000
Enter new Quantity: 6
✅ Product updated successfully!

====== PRODUCT MANAGEMENT MENU ======
4
Enter Product ID to delete: 101
✅ Product deleted successfully!

====== PRODUCT MANAGEMENT MENU ======
5
👋 Exiting program...
```