# Assignment 7

**1.** Build a program to perform CRUD operations (Create, Read, Update, Delete) on a database table Product with columns:

ProductID, ProductName, Price, and Quantity.

The program should include:

Menu-driven options for each operation.

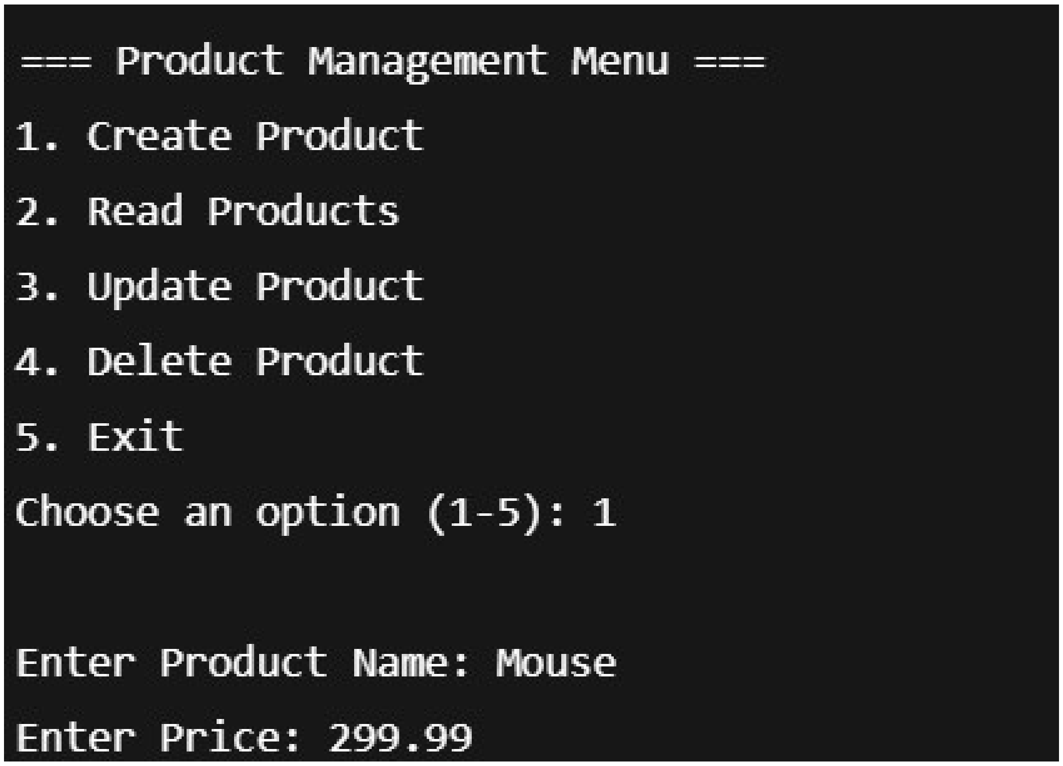
Transaction handling to ensure data integrity.

**Code:**

```
import sqlite3

# Connect to SQLite database (creates the file if not exists)

conn = sqlite3.connect("product.db")

cursor = conn.cursor()

# Create Product table if it doesn't exist

cursor.execute("""

CREATE TABLE IF NOT EXISTS Product (

    ProductID INTEGER PRIMARY KEY AUTOINCREMENT,

    ProductName TEXT NOT NULL,

    Price REAL NOT NULL,

    Quantity INTEGER NOT NULL

)

""")

conn.commit()
```

```
def create_product():
    try:
        name = input("Enter Product Name: ")
        price = float(input("Enter Price: "))
        quantity = int(input("Enter Quantity: "))
        cursor.execute("INSERT INTO Product (ProductName, Price, Quantity) VALUES (?, ?, ?)",
                       (name, price, quantity))
        conn.commit()
        print("✅Product added successfully.")
    except Exception as e:
        conn.rollback()
        print("❌Error:", e)
def read_products():
    try:
        cursor.execute("SELECT * FROM Product")
        rows = cursor.fetchall()
        if rows:
            print("\n� Product List:")
            for row in rows:
                print(f"ID: {row[0]}, Name: {row[1]}, Price: {row[2]}, Quantity: {row[3]}")
        else:
            print("□□ No products found.")
    except Exception as e:
        print(" ❌Error:", e)
def update_product():
    try:
        pid = int(input("Enter Product ID to update: "))
        name = input("Enter New Product Name: ")
        price = float(input("Enter New Price: "))
        quantity = int(input("Enter New Quantity: "))
        cursor.execute("""
            UPDATE Product
            SET ProductName = ?, Price = ?, Quantity = ?
            WHERE ProductID = ?
        """, (name, price, quantity, pid))
        if cursor.rowcount == 0:
            print("⚠️ Product not found.")
        else:
            conn.commit()
            print("✅Product updated successfully.")
    except Exception as e:
        conn.rollback()
        print(" ❌Error:", e)
def delete_product():
    try:
        pid = int(input("Enter Product ID to delete: "))
        cursor.execute("DELETE FROM Product WHERE ProductID = ?", (pid,))
        if cursor.rowcount == 0:
            print("□□ Product not found.")
        else:
            conn.commit()
            print("✅Product deleted successfully.")
    except Exception as e:
        conn.rollback()
        print(" ❌Error:", e)
def menu():
    while True:
        print("\n=== Product Management Menu ===")
        print("1. Create Product")
        print("2. Read Products")
        print("3. Update Product")
        print("4. Delete Product")
        print("5. Exit")
        choice = input("Choose an option (1-5): ")
        if choice == '1':
            create_product()
        elif choice == '2':
            read_products()
        elif choice == '3':
            update_product()
        elif choice == '4':
            delete_product()
        elif choice == '5':
            print("� Exiting program. Goodbye!")
            break
        else:
            print("❌Invalid choice. Please try again.")
# Run the menu
menu()
# Close the connection
conn.close()
```

**OUTPUT:**



```
=== Product Management Menu ===
1. Create Product
2. Read Products
3. Update Product
4. Delete Product
5. Exit
Choose an option (1-5): 1


Enter Product Name: Mouse
Enter Price: 299.99
```

**2.** Develop a Java application using JDBC and MVC architecture to manage student data. The application should:

Use a Student class as the model with fields like StudentID, Name, Department, and Marks.

Include a database table to store student data.

Allow the user to perform CRUD operations through a simple menu-driven view.

Implement database operations in a separate controller class.

**CODE:**

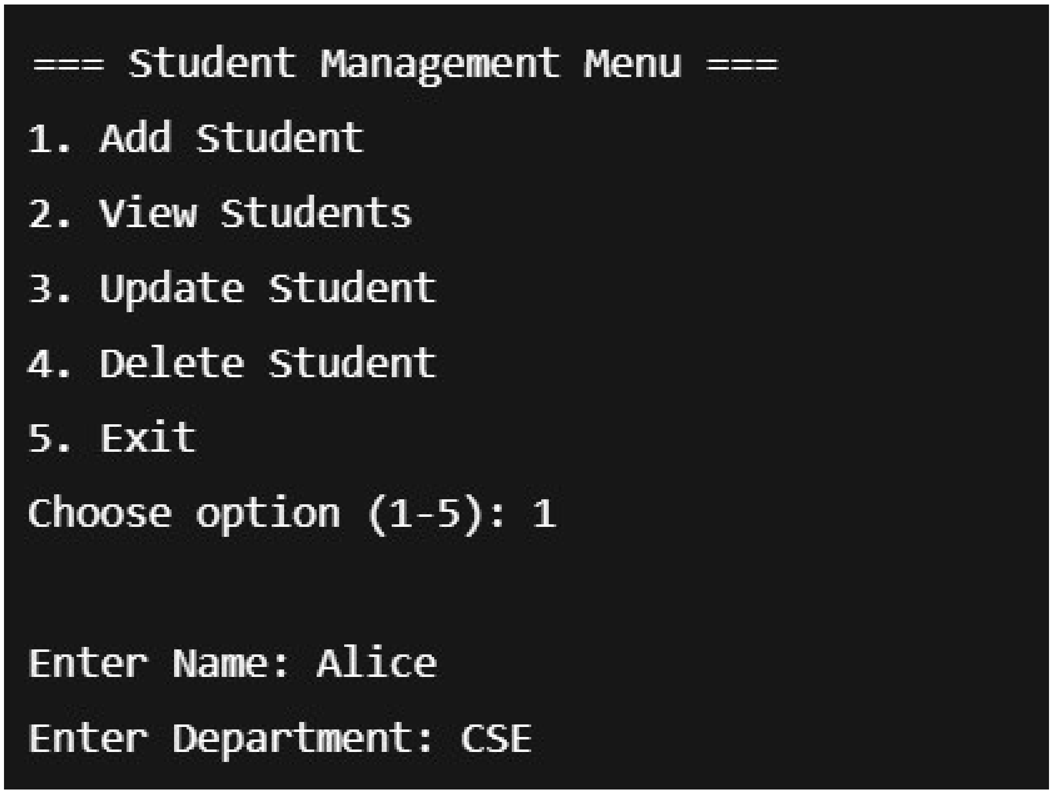
```
import java.sql.*;

import java.util.Scanner;

public class StudentManagementApp {

  // ----- MODEL -----

  static class Student {

    private int studentID;

    private String name;

    private String department;

    private double marks;

    public Student(int studentID, String name, String department, double marks) {

      this.studentID = studentID;

      this.name = name;

      this.department = department;

      this.marks = marks;

    }
    public Student(String name, String department, double marks) {
        this.name = name;
        this.department = department;
        this.marks = marks;
    }
    public int getStudentID() { return studentID; }
    public String getName() { return name; }
    public String getDepartment() { return department; }
    public double getMarks() { return marks; }
    public void setStudentID(int studentID) { this.studentID = studentID; }
    public void setName(String name) { this.name = name; }
    public void setDepartment(String department) { this.department = department; }
    public void setMarks(double marks) { this.marks = marks; }
    @Override
    public String toString() {
        return "StudentID: " + studentID + ", Name: " + name +
               ", Department: " + department + ", Marks: " + marks;
    }
}
// ----- CONTROLLER -----
static class StudentController {
    private final String url = "jdbc:mysql://localhost:3306/StudentDB";
    private final String user = "root";
    private final String password = "your_password"; // change this
    public Connection getConnection() throws SQLException {
        return DriverManager.getConnection(url, user, password);
    }
    public void addStudent(Student student) {
        String query = "INSERT INTO Students (Name, Department, Marks) VALUES (?, ?, ?)";
        try (Connection conn = getConnection(); PreparedStatement ps =
conn.prepareStatement(query)) {
            ps.setString(1, student.getName());
            ps.setString(2, student.getDepartment());
            ps.setDouble(3, student.getMarks());
            ps.executeUpdate();
            System.out.println("✅Student added successfully.");
        } catch (SQLException e) {
            System.out.println("❌Error: " + e.getMessage());
        }
    }
    public void viewStudents() {
        String query = "SELECT * FROM Students";
        try (Connection conn = getConnection(); Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery(query)) {
            boolean found = false;
            while (rs.next()) {
                found = true;
                Student s = new Student(
                        rs.getInt("StudentID"),
                    rs.getString("Name"),
                    rs.getString("Department"),
                    rs.getDouble("Marks"));
                System.out.println(s);
            }
            if (!found) {
                System.out.println("□□ No students found.");
            }
        } catch (SQLException e) {
            System.out.println(" ❌Error: " + e.getMessage());
        }
    }
    public void updateStudent(Student student) {
        String query = "UPDATE Students SET Name = ?, Department = ?, Marks = ? WHERE StudentID = ?";
        try (Connection conn = getConnection(); PreparedStatement ps = conn.prepareStatement(query)) {
            ps.setString(1, student.getName());
            ps.setString(2, student.getDepartment());
            ps.setDouble(3, student.getMarks());
            ps.setInt(4, student.getStudentID());
            int rows = ps.executeUpdate();
            if (rows > 0) {
                System.out.println("✅tudent updated successfully.");
            } else {
                    System.out.println("□□ Student not found.");
                }
            } catch (SQLException e) {
                System.out.println("❌Error: " + e.getMessage());
            }
        }
        public void deleteStudent(int studentID) {
            String query = "DELETE FROM Students WHERE StudentID = ?";
            try (Connection conn = getConnection(); PreparedStatement ps =
conn.prepareStatement(query)) {
                ps.setInt(1, studentID);
                int rows = ps.executeUpdate();
                if (rows > 0) {
                    System.out.println("✅tudent deleted successfully.");
                } else {
                    System.out.println("□□ Student not found.");
                }
            } catch (SQLException e) {
                System.out.println("❌Error: " + e.getMessage());
            }
        }
    }
    // ----- VIEW + MAIN -----
    public static void main(String[] args) {
```

```
StudentController controller = new StudentController();
Scanner scanner = new Scanner(System.in);
while (true) {
  System.out.println("\n=== Student Management Menu ===");
  System.out.println("1. Add Student");
  System.out.println("2. View Students");
  System.out.println("3. Update Student");
  System.out.println("4. Delete Student");
  System.out.println("5. Exit");
  System.out.print("Choose option (1-5): ");
  int choice = scanner.nextInt();
  switch (choice) {
    case 1 -> {
      scanner.nextLine(); // flush newline
      System.out.print("Enter Name: ");
      String name = scanner.nextLine();
      System.out.print("Enter Department: ");
      String dept = scanner.nextLine();
      System.out.print("Enter Marks: ");
      double marks = scanner.nextDouble();
      controller.addStudent(new Student(name, dept, marks));
    }
    case 2 -> controller.viewStudents();
    case 3 -> {
```

```
                System.out.print("Enter Student ID to update: ");
                int id = scanner.nextInt();
                scanner.nextLine();
                System.out.print("Enter New Name: ");
                String name = scanner.nextLine();
                System.out.print("Enter New Department: ");
                String dept = scanner.nextLine();
                System.out.print("Enter New Marks: ");
                double marks = scanner.nextDouble();
                controller.updateStudent(new Student(id, name, dept, marks));
            }
            case 4 -> {
                System.out.print("Enter Student ID to delete: ");
                int id = scanner.nextInt();
                controller.deleteStudent(id);
            }
            case 5 -> {
                System.out.println("▯ Exiting...");
                return;
            }
            default -> System.out.println(" ❌Invalid choice.");
        }
    }
}
```

}

**OUTPUT:**



```
=== Student Management Menu ===
1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Exit
Choose option (1-5): 1

Enter Name: Alice
Enter Department: CSE
```