

DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

### Experiment 2.4

Student Name: Abhinav Sharma

UID: 22BCS11022

Branch: CSE

Section: KPIT-902

Semester: 6<sup>th</sup>

DOP: 03/04/25

Subject: PBLJ

Subject Code: 22CSH-359

**Aim:** Create Java applications with JDBC for database connectivity, CRUD operations, and MVC architecture.

**Objective:** To Create Java applications with JDBC for database connectivity, CRUD operations, and MVC architecture.

**Easy Level:**

Create a Java program to connect to a MySQL database and fetch data from a single table. The program should:

Use DriverManager and Connection objects.

Retrieve and display all records from a table named Employee with columns EmpID, Name, and Salary.

**Code:**

```
import java.sql.*; import
```

```
java.util.Scanner;
```

```
public class EmployeeDatabase {    private static final String DB_URL =
```

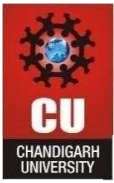
```
"jdbc:mysql://localhost:3808/test";    private static final String
```

```
USERNAME = "root";    private static final String PASSWORD =
```

```
"*****";
```

```
    public static void main(String[] args) {
```

```
        Scanner scanner = new Scanner(System.in);
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
while (true) {

    System.out.println("\n=== Employee Management System ===");

    System.out.println("1) View Employee List");

    System.out.println("2) Exit");

    System.out.print("Select an option: ");

    int option = scanner.nextInt();

    if (option == 1)

    {
fetchEmployees();

        } else if (option == 2) {

            System.out.println("Goodbye!");

            break;

        } else {

            System.out.println("Invalid choice! Please try again.");

        }

    }

    scanner.close();

}

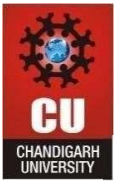
private static void fetchEmployees() {

    String query = "SELECT EmpID, Name, Salary FROM Employee";

    try (Connection conn = DriverManager.getConnection(DB_URL, USERNAME, PASSWORD);

        Statement stmt = conn.createStatement();

        ResultSet rs = stmt.executeQuery(query)) {
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
        System.out.println("\nEmployee Details:");

System.out.println("ID | Name | Salary");

        System.out.println("-----");

        while (rs.next()) {

            System.out.printf("%d | %s | %.2f\n", rs.getInt("EmpID"), rs.getString("Name"),
rs.getDouble("Salary"));

        }

    } catch (SQLException ex) {

        System.err.println("Database connection error: " + ex.getMessage());

    }

}

}
```

### Medium Level:

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:

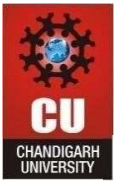
```
import java.sql.*; import

java.util.Scanner;

public class ProductManager {    private static final String DB_URL =

"jdbc:mysql://localhost:3808/test";    private static final String USER = "root";

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



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
public static void main(String[] args) {    Scanner

scanner = new Scanner(System.in);    boolean

running = true;

while (running) {

    System.out.println("\n===== Product Management =====");

    System.out.println("1) Add Product");

    System.out.println("2) View Products");

    System.out.println("3) Update Product");

    System.out.println("4) Delete Product");

    System.out.println("5) Exit");

    System.out.print("Choose an option: ");

    int choice = scanner.nextInt();

    scanner.nextLine(); // Clear newline buffer

    switch (choice) {        case 1 >
addProduct(scanner);        case 2 ->
viewProducts();        case 3 ->
updateProduct(scanner);        case 4 >
deleteProduct(scanner);        case 5
-> {

        System.out.println("Exiting application...");        running

= false;

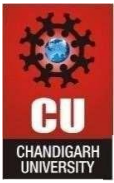
    }

    default -> System.out.println("Invalid option! Try again.");

    }

}

scanner.close();
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
}

private static void addProduct(Scanner scanner) {

    System.out.print("Enter product name: ");

    String name = scanner.nextLine();

    System.out.print("Enter price: ");

    double price = scanner.nextDouble();

    System.out.print("Enter quantity: ");    int
    quantity = scanner.nextInt();

    String sql = "INSERT INTO Product (ProductName, Price, Quantity) VALUES (?, ?, ?)";

    try (Connection conn = DriverManager.getConnection(DB_URL, USER, PASSWORD);
        PreparedStatement stmt = conn.prepareStatement(sql)) {

        stmt.setString(1, name);

        stmt.setDouble(2,    price);

        stmt.setInt(3, quantity);

        int rowsInserted = stmt.executeUpdate();

        if (rowsInserted > 0) {

            System.out.println("Product added successfully!");

        } else {

            System.out.println("Failed to add product.");

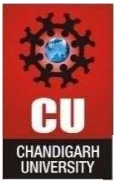
        }

    } catch (SQLException ex) {

        System.err.println("Error adding product: " + ex.getMessage());

    }

}
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
private static void viewProducts() {

    String sql = "SELECT * FROM Product";

    try (Connection conn = DriverManager.getConnection(DB_URL, USER, PASSWORD);

        Statement stmt = conn.createStatement();

        ResultSet rs = stmt.executeQuery(sql)) {

        System.out.println("\nProduct List:");

        System.out.println("ID | Name | Price | Quantity");

        System.out.println("-----");

        while (rs.next()) {

            System.out.printf("%d | %s | %.2f | %d\n",

rs.getInt("ProductID"),
rs.getString("ProductName"),
rs.getDouble("Price"),          rs.getInt("Quantity"));

        }

    } catch (SQLException ex) {

        System.err.println("Error retrieving products: " + ex.getMessage());

    }

}

private static void updateProduct(Scanner scanner)

{
    System.out.print("Enter product ID to update:

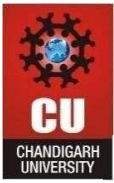
");
    int id = scanner.nextInt();
    scanner.nextLine();

    // Clear buffer

    System.out.print("Enter new product name: ");

    String name = scanner.nextLine();

    System.out.print("Enter new price: ");
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
double price = scanner.nextDouble();
```

```
System.out.print("Enter new quantity: ");    int
```

```
quantity = scanner.nextInt();
```

```
String sql = "UPDATE Product SET ProductName=?, Price=?, Quantity=? WHERE ProductID=?";
```

```
try (Connection conn = DriverManager.getConnection(DB_URL, USER, PASSWORD);
```

```
PreparedStatement stmt = conn.prepareStatement(sql)) {
```

```
    stmt.setString(1, name);
```

```
    stmt.setDouble(2, price);
```

```
    stmt.setInt(3, quantity);    stmt.setInt(4,
```

```
    id);
```

```
    int rowsUpdated = stmt.executeUpdate();
```

```
    if (rowsUpdated > 0) {
```

```
        System.out.println("Product updated successfully!");
```

```
    } else {
```

```
        System.out.println("Product ID not found.");
```

```
    }
```

```
    } catch (SQLException ex) {
```

```
        System.err.println("Error updating product: " + ex.getMessage());
```

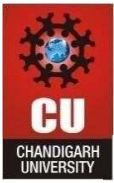
```
    }
```

```
}
```

```
private static void deleteProduct(Scanner scanner) {
```

```
System.out.print("Enter product ID to delete: ");    int
```

```
id = scanner.nextInt();
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
String sql = "DELETE FROM Product WHERE ProductID=?";

try (Connection conn = DriverManager.getConnection(DB_URL, USER, PASSWORD);
    PreparedStatement stmt = conn.prepareStatement(sql)) {

    stmt.setInt(1, id);

    int
    rowsDeleted = stmt.executeUpdate();

    if (rowsDeleted > 0) {

        System.out.println("Product deleted successfully!");

    } else {

        System.out.println("Product ID not found.");

    }

} catch (SQLException ex) {

    System.err.println("Error deleting product: " + ex.getMessage());

}

}
```

#### Hard Level:

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.

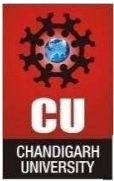
Allow the user to perform CRUD operations through a simple menu-driven view.

Implement database operations in a separate controller class.

#### Code:

#### Model



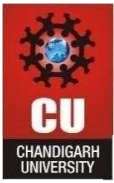


DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
public class Student {  
  
    private int id;    private  
    String fullName;    private  
    String dept;    private int  
    score;  
  
    public Student(int id, String fullName, String dept, int score) {  
        this.id = id;  
        this.fullName = fullName;  
        this.dept = dept;    this.score  
        = score;  
    }  
  
    // Getters and Setters    public int getId()  
    { return id; }    public void setId(int id) {  
        this.id = id; }  
  
    public String getFullName() { return fullName; }    public void  
    setFullName(String fullName) { this.fullName = fullName; }  
  
    public String getDept() { return dept; }    public  
    void setDept(String dept) { this.dept = dept; }  
  
    public int getScore() { return score; }    public void setScore(int  
    score) { this.score = score; }  
    @Override    public String toString() {    return  
    "Student ID: " + id + ", Name: " + fullName + ",  
    Department: " + dept + ", Score: " + score;  
    }  
}
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

View

```
import java.util.List;
```

```
import java.util.Scanner;
```

```
public class StudentView {    private final StudentController studentController  
= new StudentController();    private final Scanner inputScanner = new  
Scanner(System.in);
```

```
    public void showMenu() {
```

```
        int option;
```

```
    do {
```

```
        System.out.println("\n=== Student Management Portal ===");
```

```
        System.out.println("1. Register Student");
```

```
        System.out.println("2. Display All Students");
```

```
        System.out.println("3. Modify Student Details");
```

```
        System.out.println("4. Remove Student");
```

```
        System.out.println("5. Exit");
```

```
        System.out.print("Select an option: ");        option
```

```
        = inputScanner.nextInt();
```

```
        inputScanner.nextLine(); // Consume newline
```

```
        switch (option) {            case 1:
```

```
            registerStudent();            break;
```

```
        case 2:
```

```
            listStudents();            break;
```

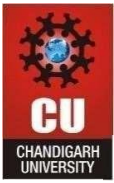
```
        case 3:
```

```
            modifyStudent();            break;
```

```
        case 4:
```

```
            removeStudent();            break;
```

```
        case 5:
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
        System.out.println("Closing application...");

        break;           default:

        System.out.println("Invalid option, please try again.");

    }

    } while (option != 5);

}

private void registerStudent() {

    System.out.print("Enter Student Name: ");

    String fullName = inputScanner.nextLine();

    System.out.print("Enter Department: ");

    String department = inputScanner.nextLine();

    System.out.print("Enter Marks: ");    int score

    = inputScanner.nextInt();

    Student newStudent = new Student(0, fullName, department, score);

    studentController.addStudent(newStudent);

}

private void listStudents() {

    List<Student> studentList = studentController.getAllStudents();    if

(studentList.isEmpty()) {

        System.out.println("No student records available.");

    } else {

        System.out.println("\n--- Student Records ---");        for

(Student student : studentList) {

            System.out.println(student);

        }

    }

}
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
}
```

```
private void modifyStudent() {
```

```
    System.out.print("Enter Student ID to update: ");    int
```

```
studentId = inputScanner.nextInt();
```

```
inputScanner.nextLine(); // Consume newline
```

```
System.out.print("Enter Updated Name: ");
```

```
    String updatedName = inputScanner.nextLine();
```

```
    System.out.print("Enter Updated Department: ");
```

```
String updatedDepartment = inputScanner.nextLine();
```

```
System.out.print("Enter Updated Marks: ");    int
```

```
updatedScore = inputScanner.nextInt();
```

```
    Student updatedStudent = new Student(studentId, updatedName, updatedDepartment, updatedScore);
```

```
studentController.updateStudent(updatedStudent);
```

```
}
```

```
private void removeStudent() {
```

```
    System.out.print("Enter Student ID to remove: ");    int
```

```
studentId = inputScanner.nextInt();
```

```
studentController.deleteStudent(studentId);
```

```
}
```

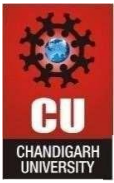
```
}
```

Controller

```
import java.sql.*; import
```

```
java.util.ArrayList; import
```

```
java.util.List;
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
public class StudentController {    private static final String DB_URL =
"jdbc:mysql://localhost:3306/javadb";    private static final String
DB_USER = "root";    private static final String DB_PASSWORD = "karan.111";

    public void insertStudent(Student student) {

        String sql = "INSERT INTO Students (Name, Department, Marks) VALUES (?, ?, ?)";

        try (Connection connection = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);

            PreparedStatement preparedStatement = connection.prepareStatement(sql)) {

            connection.setAutoCommit(false);                preparedStatement.setString(1,
student.getName());                preparedStatement.setString(2,
student.getDepartment());                preparedStatement.setInt(3,
student.getMarks());

            preparedStatement.executeUpdate();
connection.commit();

            System.out.println("Student successfully registered!");

        } catch (SQLException ex) {                ex.printStackTrace();
        }
    }

    public List<Student> fetchAllStudents() {

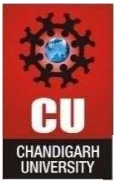
        List<Student> studentList = new ArrayList<>();

        String sql = "SELECT * FROM Students";

        try (Connection connection = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);

            Statement statement = connection.createStatement();

            ResultSet resultSet = statement.executeQuery(sql)) {
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
        while (resultSet.next()) {            studentList.add(new
Student(resultSet.getInt("StudentID"),
resultSet.getString("Name"),
resultSet.getString("Department"),            resultSet.getInt("Marks")));
        }

    } catch (SQLException ex) {            ex.printStackTrace();
    }    return studentList;
}

public void modifyStudent(Student student) {
    String sql = "UPDATE Students SET Name=?, Department=?, Marks=? WHERE StudentID=?";

    try (Connection connection = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);
        PreparedStatement preparedStatement = connection.prepareStatement(sql)) {
        connection.setAutoCommit(false);            preparedStatement.setString(1, student.getName());
        preparedStatement.setString(2, student.getDepartment());            preparedStatement.setInt(3, student.getMarks());
        preparedStatement.setInt(4, student.getStudentID());

        int affectedRows = preparedStatement.executeUpdate();

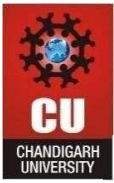
        if (affectedRows > 0) {

            connection.commit();

            System.out.println("Student details updated!");
        } else {

            System.out.println("No record found with the given Student ID.");
        }

    } catch (SQLException ex) {            ex.printStackTrace();
```



DEPARTMENT OF

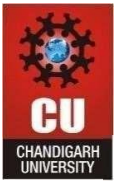
COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
    }  
}  
  
public void removeStudent(int studentID) {  
    String sql = "DELETE FROM Students WHERE StudentID=?";  
  
    try (Connection connection = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);  
        PreparedStatement preparedStatement = connection.prepareStatement(sql)) {  
  
        connection.setAutoCommit(false);  
preparedStatement.setInt(1, studentID);  
  
        int affectedRows = preparedStatement.executeUpdate();  
if (affectedRows  
> 0) {            connection.commit();  
        System.out.println("Student record deleted!");  
    } else {  
        System.out.println("No record found with the given Student ID.");  
    }  
  
    } catch (SQLException ex) {            ex.printStackTrace();  
    }  
}  
}
```

Main

```
public class StudentApplication {  
  
    public static void main(String[] args) {  
        StudentView studentView = new StudentView();        studentView.showMenu();  
    }  
}
```



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
}  
  
}
```

Output:

```
C:\Users\123sa\Desktop\Coding\JAVA\Class\exp 7>javac -cp ".;mysql-connector-j-9.2.0.jar" MySQLConnectionCode.java  
C:\Users\123sa\Desktop\Coding\JAVA\Class\exp 7>java -cp ".;mysql-connector-j-9.2.0.jar" MySQLConnectionCode  
  
Menu:  
1. Display Employees  
2. Exit  
Enter your choice: 1  
  
EmpID | Name | Salary  
-----  
1 | Saket Agarwal | 55000.0  
2 | Ram | 32000.5  
3 | Dam | 41000.75  
4 | Pam | 53000.25  
  
Menu:  
1. Display Employees  
2. Exit  
Enter your choice: 2  
Exiting...
```

1.1 Easy Problem





```
C:\Users\123sa\Desktop\Coding\JAVA\Class\exp 7>javac -cp ".;mysql-connector-j-9.2.0.jar" ProductCRUD.java
C:\Users\123sa\Desktop\Coding\JAVA\Class\exp 7>java -cp ".;mysql-connector-j-9.2.0.jar" ProductCRUD

--- Product Management System ---
1. Add Product
2. View Products
3. Update Product
4. Delete Product
5. Exit
Enter your choice: 2

ProductID | ProductName | Price | Quantity
-----
1 | Laptop | 75000.0 | 10
2 | Mouse | 1500.0 | 50
3 | Keyboard | 2500.0 | 30

--- Product Management System ---
1. Add Product
2. View Products
3. Update Product
4. Delete Product
5. Exit
Enter your choice: 4
Enter Product ID to delete: 3
Product deleted successfully!

--- Product Management System ---
1. Add Product
2. View Products
3. Update Product
4. Delete Product
5. Exit
Enter your choice: 2

ProductID | ProductName | Price | Quantity
-----
1 | Laptop | 75000.0 | 10
2 | Mouse | 1500.0 | 50

--- Product Management System ---
1. Add Product
2. View Products
3. Update Product
4. Delete Product
5. Exit
Enter your choice: 5
Exiting...
```

## 1.2 Medium Problem

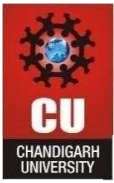
```
C:\Users\123sa\Desktop\Coding\JAVA\Class\exp 7>java -cp ".;mysql-connector-j-9.2.0.jar" StudentMain

--- Student Management System ---
1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Exit
Enter your choice: 2

Student List:
ID: 1, Name: Saket, Dept: Computer Science, Marks: 95
ID: 2, Name: Ram, Dept: Electronics, Marks: 78
ID: 3, Name: Dam, Dept: Mechanical, Marks: 92

--- Student Management System ---
1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Exit
Enter your choice: 5
Exiting...
```

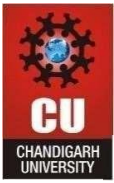
## 1.3 Hard Problem



DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.



## COMPUTER SCIENCE & ENGINEERING

### Learning Outcomes:

1. Integrating Java with Databases – Learn how Java applications interact with databases to store and retrieve data efficiently.
2. Enhancing Data Security – Explore best practices for securing database connections and preventing SQL injection attacks in Java applications.
3. Optimizing Query Performance – Understand how to write efficient SQL queries and use indexing to improve database performance.
4. Building Scalable Applications – Learn how to design a Java-based system that can handle increasing data loads while maintaining performance.