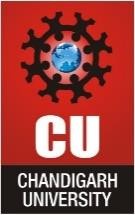
# DEPARTMENT OF



COMPUTER SCIENCE & ENGINEERING

**Student Name: Garv Branch:** BE-CSE **Semester:** 6th

**Experiment 2.4**

**UID:** 22BCS11750

**Section/Group:** 22BCS\_IOT-638/B

**Date of Performance:** 01/04/2025

**Subject Name:** Project based Learning in JAVA **Subject Code:** 22CSH-359

## Aim:

Easy Level: Problem Statement: 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.

Medium Level:

Problem Statement: 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.

Hard Level:

Problem Statement: 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.

## Objective:

* + Understand JDBC (Java Database Connectivity) for database interactions.
  + Implement CRUD operations using SQL queries in Java.
  + Learn transaction handling for data consistency.
  + Follow the MVC (Model-View-Controller) architecture in Java applications.

## Implementation/Code:

#ConnectionClass.java

package com.capgemini.main;

import java.sql.Connection; import java.sql.DriverManager; import java.sql.SQLException;

public class ConnectionClass {

public Connection getConnection() throws ClassNotFoundException, SQLException { Connection con = null;

Class.forName("com.mysql.cj.jdbc.Driver"); // load and register

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/cuims", "root", "Harshit@1"); System.out.println("Connection Established");

return con;

}

}

#MainClass.java

package com.capgemini.main; import java.sql.SQLException; import java.util.Scanner;

public class MainClass {

public static void main(String[] args) throws ClassNotFoundException, SQLException { OperationClass object = new OperationClass();

Scanner sc = new Scanner(System.in);

while (true) {

System.out.println("Enter 1 to insert the record:"); System.out.println("Enter 2 to delete the record:"); System.out.println("Enter 3 to update the record:"); System.out.println("Enter 4 to display one record:"); System.out.println("Enter 5 to display all records:"); System.out.println("Enter 6 to exit:");

int reply = sc.nextInt(); switch (reply) {

case 1:

object.insertRecord(); break;

case 2:

object.delete(); break;

case 3:

object.update(); break;

case 4:

object.disply\_one();

break; case 5:

object.display\_All(); break;

case 6:

System.out.println("Exiting..."); sc.close();

System.exit(0); default:

System.out.println("Invalid Input");

}

}

}

}

# OperationClass.java

package com.capgemini.main; import java.sql.Connection;

import java.sql.PreparedStatement; import java.sql.ResultSet;

import java.sql.SQLException; import java.util.Scanner;

public class OperationClass {

Scanner sc = new Scanner(System.in); Connection con = null;

ConnectionClass conObject = new ConnectionClass(); int count = 0;

public void insertRecord() throws ClassNotFoundException, SQLException { System.out.println("Enter student UID:");

String studentUID = sc.next(); System.out.println("Enter student name:"); String studentName = sc.next(); System.out.println("Enter student Course:"); String studentCourse = sc.next(); System.out.println("Enter student Batch:"); int studentBatch = sc.nextInt();

if (con == null) {

con = conObject.getConnection();

}

String countStatus = "SELECT MAX(idstudent) FROM student"; PreparedStatement psmt1 = con.prepareStatement(countStatus); ResultSet rs = psmt1.executeQuery();

if (rs.next()) {

count = rs.getInt(1);

}

String insertQuery = "INSERT INTO student VALUES(?,?,?,?,?)"; PreparedStatement psmt2 = con.prepareStatement(insertQuery); count += 1;

psmt2.setInt(1, count); psmt2.setString(2, studentUID); psmt2.setString(3, studentName); psmt2.setString(4, studentCourse); psmt2.setInt(5, studentBatch);

int status = psmt2.executeUpdate(); if (status > 0) {

System.out.println("Record inserted successfully");

} else {

System.out.println("Record not inserted");

}

psmt1.close(); psmt2.close();

}

public void update() throws ClassNotFoundException, SQLException { System.out.println("Enter student UID:");

String studentUID = sc.next();

String searchQuery = "SELECT \* FROM student WHERE studentUID=?"; con = conObject.getConnection();

PreparedStatement psmt1 = con.prepareStatement(searchQuery); psmt1.setString(1, studentUID);

ResultSet rs = psmt1.executeQuery();

if (rs.next()) {

System.out.println("Student UID: " + rs.getString(2)); System.out.println("Student Name: " + rs.getString(3)); System.out.println("Student Course: " + rs.getString(4)); System.out.println("Student Batch: " + rs.getString(5));

} else {

System.out.println("Record not found"); return;

}

System.out.println("Enter updated course: "); String updatedCourse = sc.next();

String updatedQuery = "UPDATE student SET studentCourse=? WHERE studentUID=?"; PreparedStatement psmt2 = con.prepareStatement(updatedQuery);

psmt2.setString(1, updatedCourse); psmt2.setString(2, studentUID);

int status = psmt2.executeUpdate(); if (status > 0) {

System.out.println("Record updated successfully");

} else {

System.out.println("Update failed");

}

psmt1.close(); psmt2.close(); con.close();

}

public void delete() throws ClassNotFoundException, SQLException { System.out.println("Enter student UID:");

String studentUID = sc.next();

String searchQuery = "SELECT \* FROM student WHERE studentUID=?"; con = conObject.getConnection();

PreparedStatement psmt1 = con.prepareStatement(searchQuery); psmt1.setString(1, studentUID);

ResultSet rs = psmt1.executeQuery();

if (rs.next()) {

System.out.println("Student UID: " + rs.getString(2)); System.out.println("Student Name: " + rs.getString(3)); System.out.println("Student Course: " + rs.getString(4)); System.out.println("Student Batch: " + rs.getString(5)); System.out.println("Confirm deletion? 1/Yes, 0/No");

int reply = sc.nextInt(); if (reply == 1) {

String deleteQuery = "DELETE FROM student WHERE studentUID=?"; PreparedStatement psmt2 = con.prepareStatement(deleteQuery); psmt2.setString(1, studentUID);

int status = psmt2.executeUpdate(); if (status > 0) {

System.out.println("Record deleted successfully");

} else {

System.out.println("Deletion failed");

}

psmt2.close();

}

} else {

System.out.println("Record not found");

}

psmt1.close(); con.close();

}

public void displayOne() throws ClassNotFoundException, SQLException { System.out.println("Enter student UID:");

String studentUID = sc.next();

String searchQuery = "SELECT \* FROM student WHERE studentUID=?"; con = conObject.getConnection();

PreparedStatement psmt1 = con.prepareStatement(searchQuery); psmt1.setString(1, studentUID);

ResultSet rs = psmt1.executeQuery();

if (rs.next()) {

System.out.println("Student UID: " + rs.getString(2)); System.out.println("Student Name: " + rs.getString(3));

System.out.println("Student Course: " + rs.getString(4)); System.out.println("Student Batch: " + rs.getString(5));

} else {

System.out.println("Record not found");

}

psmt1.close(); con.close();

}

public void displayAll() throws ClassNotFoundException, SQLException { String searchQuery = "SELECT \* FROM student";

con = conObject.getConnection();

PreparedStatement psmt1 = con.prepareStatement(searchQuery); ResultSet rs = psmt1.executeQuery();

while (rs.next()) {

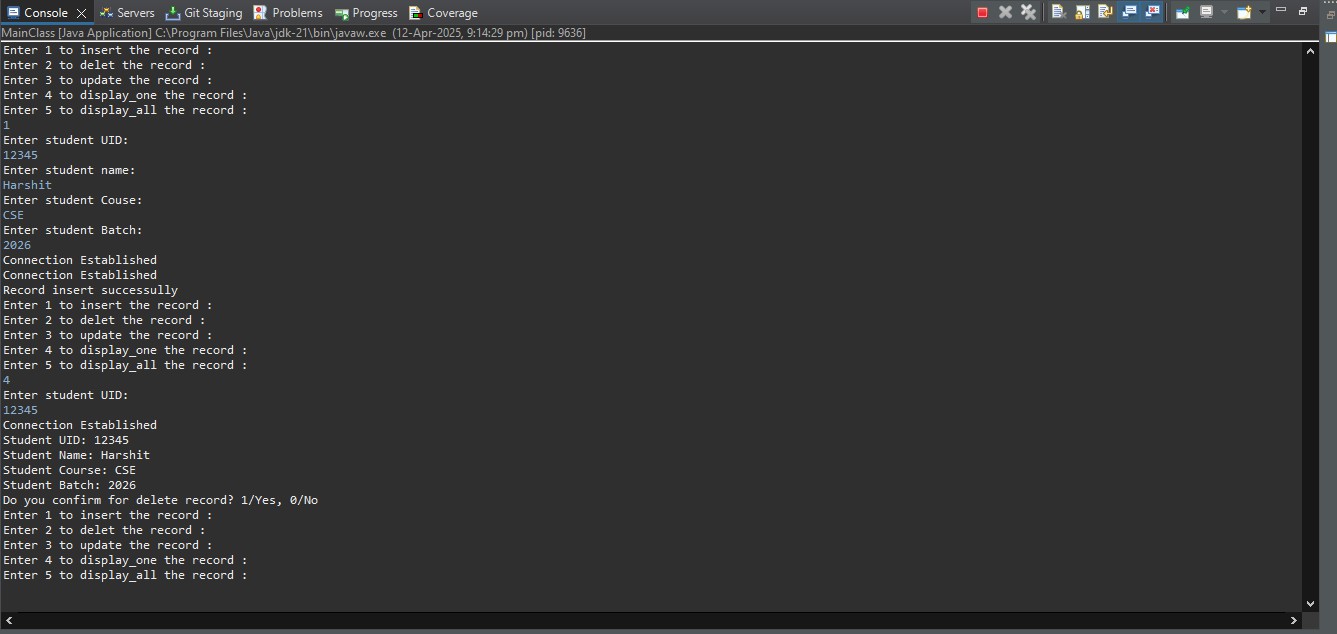
System.out.println("Student UID: " + rs.getString(2)); System.out.println("Student Name: " + rs.getString(3)); System.out.println("Student Course: " + rs.getString(4)); System.out.println("Student Batch: " + rs.getString(5)); System.out.println(" ");

}

psmt1.close(); con.close();

}

}

1. **Output:**

## Learning Outcome:

* + Gain hands-on experience with JDBC and MySQL.
  + Learn how to implement CRUD operations using Java.
  + Understand transaction handling for data consistency.
  + Apply MVC architecture in a database-driven application.
  + Improve skills in handling database connections and user interactions in Java.

i.