# Experiment 7

Student Name: Satyam Rawat
UID: 22BCS14274
Branch: CSE
Section: 22BCS_KRG_IOT_3B
Semester: 6th
Date of Performance: 12/03/25
Subject: Project Based Learning In Java
Subject Code: 22CSH-359

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

**Objective:** To develop Java applications that integrate with databases using JDBC, implement CRUD (Create, Read, Update, Delete) operations, and follow the MVC (Model-View-Controller) architecture for structured application development.

## Code:

```
import java.sql.*;
public class App {

    public static void main(String[] args) {
        String url =
" jdbc: sql server:/ /loc alhost: 1433;databaseName=j dbc;e ncryp t=true;trustServ erC ertificate=true;integrated
ecurity=true";
        String username = "ANUSHKA\\anu.pc";
        String password = "Anushka@7";
        try{
            //connection
           Connection conn=DriverManager.getConnection(url, username, password);
           System.out.println("connected to db");
          // statement
          Statement stmt=conn.createStatement();
           String newTable = "IF NOT EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES
 WHERE TABLE_NAME = 'students') "
                + "BEGIN "
                + "CREATE TABLE students ("
                + "id INT PRIMARY KEY IDENTITY(1,1), "
                + "name VARCHAR(20), "
                + "age INT"
                + "); "
                + "END;";


           stmt.executeUpdate(newTable);
           System.out.println("table created successfully");
```

```
        stmt.executeUpdate(insertQuery);
        String updateQuery="update students set age=21 where name='Anushka'";
        stmt.executeUpdate(updateQuery);
        System.out.println("update successfull");
        String deleteQuery="delete from students where name='Anushka'";
        stmt.executeUpdate(deleteQuery);
        System.out.println("deletion successfull");
        String selectQuery="select * from students";
        ResultSet rs=stmt.executeQuery(selectQuery);
        while(rs.next()){

            System.out.println("Id: "+rs.getInt("id")
            +" Name: "+rs.getString("name")+" Age: "
            + rs.getInt("age"));
        }
                }catch(SQLException e){
                System.out.println(e);
            }
        }
    }
```

## Output:

```
PS C:\Users\hp.pc\Desktop\programming languages\javaPackages\backendWithJava>
PS C:\Users\hp.pc\Desktop\programming languages\javaPackages\backendWithJava>  c
top\programming languages\javaPackages\backendWithJava'; & 'C:\Program Files\Jav
:\Users\hp.pc\AppData\Local\Temp\cp_54e5oyyxuhvloo2qgry1j26pi.argfile' 'App'
connected to db
table created successfully
update successfull
Id: 1005 Name: Anushka Age: 21
PS C:\Users\hp.pc\Desktop\programming languages\javaPackages\backendWithJava>
```

```
top\programming languages\javaPackages\backendWithJava'; & 'C:\Program Files\Java
:\Users\hp.pc\AppData\Local\Temp\cp_54e5oyyxuhvloo2qgry1j26pi.argfile' 'App'
connected to db
table created successfully
update successfull
deletion successfull
PS C:\Users\hp.pc\Desktop\programming languages\javaPackages\backendWithJava>
```

## Learning Outcomes:

By the end of this course/module, learners will be able to:

1. Understand JDBC Fundamentals – Explain the purpose of JDBC and set up database connectivity in Java applications.
2. Perform CRUD Operations – Implement Create, Read, Update, and Delete functionalities using JDBC.
3. Apply MVC Architecture – Design Java applications following the Model-View-Controller pattern for better modularity and maintainability.
4. Handle Database Connections Efficiently – Use connection pooling and manage database resources properly to optimize performance.