Experiment 7

Student Name: Nikita **UID:22BCS13534**
Branch: CSE **Section/Group:905-A**
Semester: 6 **Date of Performance:2/3/25**
Subject Name: Java with Lab **Subject Code: 22CSH-359**

1. Aim:

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..

2. Objective:

- Develop Java programs using lambda expressions and stream operations for sorting, filtering, and processing large datasets efficiently.

- Implement easy, medium, and hard-level tasks involving sorting employees, filtering and sorting students, and processing products using streams.

3. Implementation/Code:

a.

```
import java.sql.*;


public class
MySQLConnecti
on {
    public static
void
main(String[]
args) {
        // Database
connection
parameters
        String url =
"jdbc:mysql://loc
```

```
alhost:3306/your
_database_name"
;
    String user
=
"your_username"
;
    String
password =
"your_password"
;

    // SQL
query to retrieve
data from the
Employee table
    String query
= "SELECT
EmpID, Name,
Salary FROM
Employee";

    try (
      //
Establish
connection

Connection conn
=
DriverManager.g
etConnection(url,
user, password);
      // Create
a statement
```

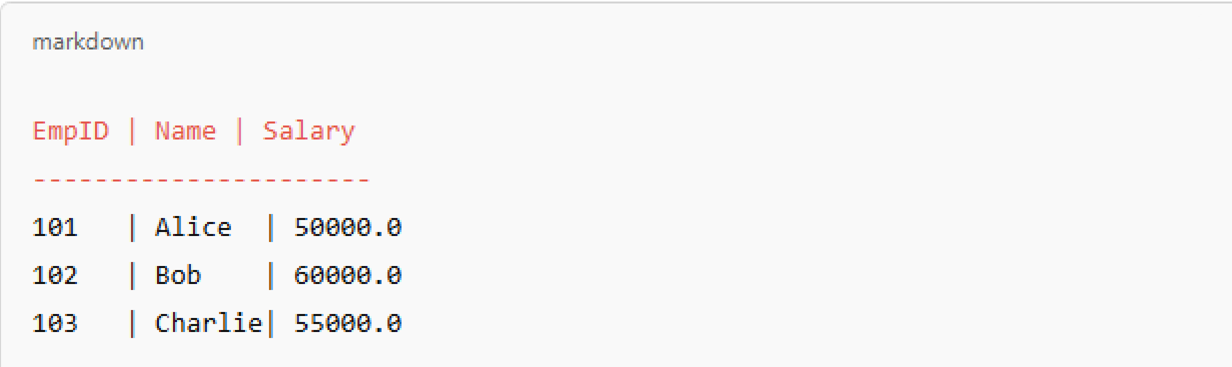
```
            Statement
stmt =
conn.createState
ment();
            // Execute
the query
            ResultSet
rs =
stmt.executeQuer
y(query)
        ) {
            // Display
results

System.out.printl
n("EmpID |
Name | Salary");

System.out.printl
n("-----------------
-----");
            while
(rs.next()) {
                int
empID =
rs.getInt("EmpID
");
                String
name =
rs.getString("Na
me");
                double
salary =
    rs.getDouble("Sa
    lary");

    System.out.printl
    n(empID + " | " +
    name + " | " +
    salary);
                }
            } catch
    (SQLException
    e) {

    e.printStackTrace
    ();
            }
        }
}
b.  import java.io.*;
import java.util.*;

public              class
EmployeeApp {
    private static final
String FILE_NAME
= "employees.txt";

    public  static  void
main(String[] args) {
        Scanner
scanner     =     new
Scanner(System.in);
        int choice;

        do {
```

```
System.out.println("
\nMenu:");

System.out.println("
1. Add an
Employee");

System.out.println("
2. Display All
Employees");

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

System.out.print("E
nter your choice: ");
      choice =
scanner.nextInt();

scanner.nextLine();

      switch
(choice) {
        case 1:

addEmployee(scann
er);
          break;
        case 2:

displayEmployees();
          break;
        case 3:
```

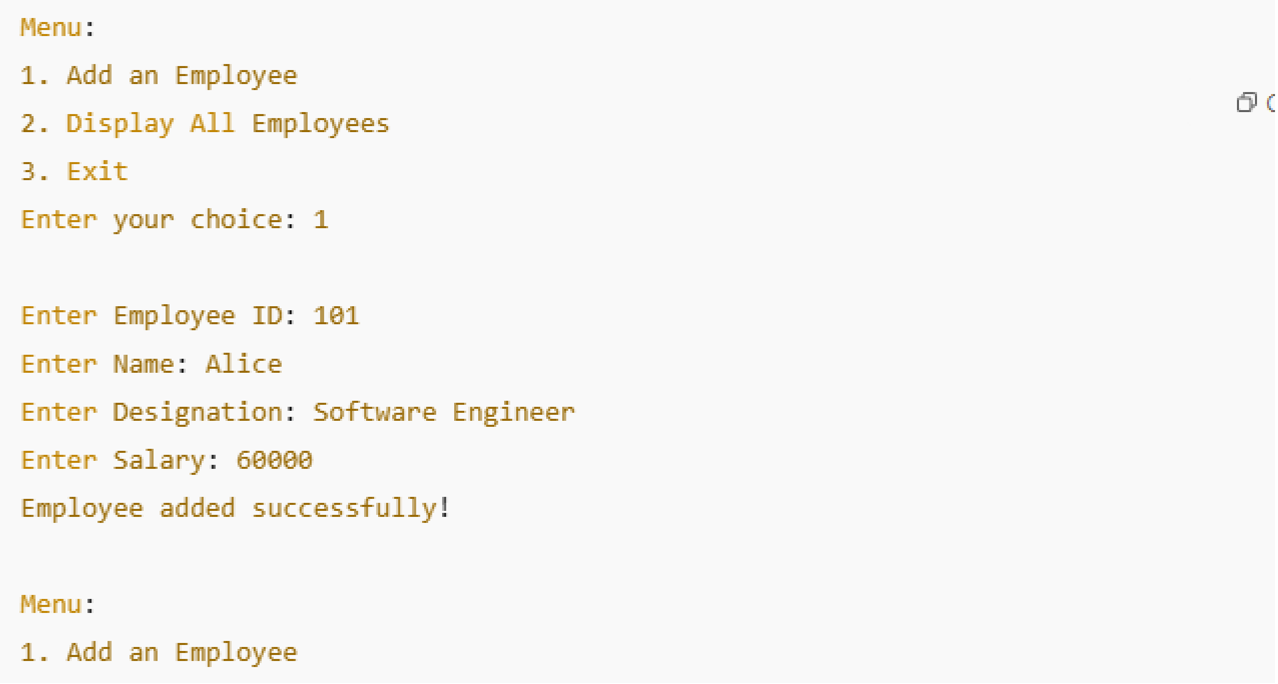
```
System.out.println("
Exiting
application...");
          break;
        default:

System.out.println("
Invalid choice.
Please try again.");
      }
    } while (choice
!= 3);

    scanner.close();
  }

  private static void
addEmployee(Scann
er scanner) {
    try  (FileWriter
fw = new
FileWriter(FILE_N
AME, true);

BufferedWriter bw =
new
BufferedWriter(fw);
      PrintWriter
out = new
PrintWriter(bw)) {


System.out.print("E
nter   Employee   ID:
");
            int  empID  =
scanner.nextInt();

scanner.nextLine();


System.out.print("E
nter Name: ");
            String   name
=
scanner.nextLine();


System.out.print("E
nter Designation: ");
            String
designation            =
scanner.nextLine();


System.out.print("E
nter Salary: ");
            double salary
=
scanner.nextDouble(
);

scanner.nextLine();


out.println(empID  +
","  +  name  +  ","  +
designation + "," +
salary);

System.out.println("
Employee added
successfully!");
    } catch
(IOException e) {

System.out.println("
Error writing to file:
" + e.getMessage());
    }
  }

  private static void
displayEmployees()
{
    try
(BufferedReader br
= new
BufferedReader(ne
w
FileReader(FILE_N
AME))) {
      String line;

System.out.println("
\nEmployee
Details:");
      while ((line =
br.readLine()) !=
null) {
```

```
                String[]
details                =
line.split(",");

System.out.println("
EmpID:       "       +
details[0] + ", Name:
" + details[1] + ",
Designation:   "   +
details[2]     +     ",
Salary:       "       +
details[3]);
            }
        }             catch
(IOException e) {

System.out.println("
Error reading file or
file does not exist.");
        }
    }
}
```

4. Output:



```
markdown

EmpID | Name | Salary
-----------------------
101    | Alice   | 50000.0
102    | Bob     | 60000.0
103    | Charlie| 55000.0
```



```
Menu:
1. Add an Employee
2. Display All Employees
3. Exit
Enter your choice: 1

Enter Employee ID: 101
Enter Name: Alice
Enter Designation: Software Engineer
Enter Salary: 60000
Employee added successfully!

Menu:
1. Add an Employee
```

5. Learning Outcome:

- **File Handling** – Uses FileWriter & BufferedReader to store and retrieve employee data.
- **Menu System** – do-while loop with switch case for user interaction.
- **User Input** – Handles int, String, double with Scanner.
- **String Manipulation** – Uses split(",") for parsing stored data.
- **Exception Handling** – try-catch ensures smooth file operations.