**Week 10 Day 3 Lab Coding Challenges**

# 

# Question 1

**Objective:**

To enable the Mfg Based business to automate the Data retrieval of **Employees** information

- create Java Based backend modules to perform Read Operations from RDBMS using JDBC

**Problem Statement:**

You are part of a solutions team for a Manufacturing Unit and as part of their automation process, they are in need of developing java modules for performing Read operations on the tables as needed by userRDBMS - MySQL (Create the complete Template but implement Read Operation, other operations would be implemented on incremental basis in subsequent Lab exercises)

**Steps:**

Create a java application with separate classes for

**Connection** - MyConnection which return the Connection to the required Database

**Model** - **POJO** class representing the required Table - **Employee**

**Service** class - which invokes the DB related functionalities - **EmployeeService**

**DAO** class - which connects to DB to perform operations - **EmployeeDAO**(Data Access Object Class)

**Client** class with Menu options to Read., Delete , Update , Insert & Search By ID

(But in this **Lab exercise - You will only implement the menu option for read Employees from DB**, in the subsequent Lab Problem statements, you will add features in terms of other CRUD operations in incremental fashion for the other Menu options)

The activity to show up the menu to perform the above operations, should be repetitively carried out in a loop as long as the user wishes

Create separate Packages for the above.

Create the necessary Database & Table at the backend.(MySQL)

**Stub Code:**

Maintain the employee Code in “EXXX” format for eg. E001,E002 etc, for which we need to create auto generation code in the later Lab Exercises

**MYConnection class**

public class MyConnection {

Connection con=null;

String user = "root";

String url = "jdbc:mysql://localhost:3306/ctsdata";

String password = "xyz";

public Connection getMyConnections()

{

}

}

**Model Class:**

public class Employee {

/\* CREATE

\* Constructor

\* Overloaded Constructor

\* Getters/Setters

\* toString()

\*

\*/

String employeeId;

String employeeName;

String employeeAddress;

String employeePhone;

int employeeSalary;

float employeeTax;

}

**EmployeeDAO class**

public class EmployeeDAO {

Connection con;

ResultSet rs;

Statement stmt;

PreparedStatement pstmt;

Employee employee;

ArrayList <Employee> employees;

MyConnection mycon;

public ArrayList <Employee> getAllEmployees()

{

**//IMPLEMENT HERE**

}

public Employee getEmployeeById(String empId)

{

}

public boolean deleteEmployeeById(String empId)

{

}

public boolean updateEmployee(Employee empU)

{

}

public boolean insertEmployee(Employee empI)

{

}

public ArrayList <String> getEmployeeIds()

{

}

}

**EmployeeService class**

public class EmployeeService {

EmployeeDAO edao;

public EmployeeService()

{

}

public ArrayList <Employee> getEmployeeRecords()

{

**//IMPLEMENT HERE**

}

public Employee searchEmployeeById(String empId)

{

}

public boolean insertEmployeeRecord(Employee emplI)

{

}

public boolean updateEmployeeRecord(Employee emplU)

{

}

public boolean deleteEmployeeRecord(String empId)

{

}

public ArrayList <String> getAllEmployeeIds()

{

}

}

**//For Menu Options create EmployeeDataManagement Class**

public class EmployeeDataManagement

{

public void showMenu()

{

**//IMPLEMENT HERE**

}

}

**//Client Class**

public class ClientUI {

public static void main(String[] args) {

// TODO Auto-generated method stub

EmployeeDataManagement edm = new EmployeeDataManagement();

edm.showMenu();

}

}

**Sample Output 1:**

Employee Details Management Menu.........

1.View All Employees

2.Search Employee By ID

3.Insert Employee Record

4.Delete Employee ID

5.Update Employee

6.Exit

**Please enter Your Option...**

**1**

**Viewing All Employees...**

Employee [employeeId=E001, employeeName=SreeHarsha, employeeAddress=RTNagar, employeePhone=9838388383, employeeSalary=10000, employeeTax=10.3]

Employee [employeeId=E002, employeeName=SumanthKumar, employeeAddress=Indiranagar, employeePhone=9786388383, employeeSalary=12000, employeeTax=12.3]

Employee [employeeId=E003, employeeName=KumareshGupta, employeeAddress=Malleswaram, employeePhone=9876543210, employeeSalary=30000, employeeTax=12.34]

Employee [employeeId=E004, employeeName=Muralidhar, employeeAddress=RTNagar, employeePhone=9812345687, employeeSalary=20000, employeeTax=13.34]

**Do You Wish to Continue yes/no?**

**Yes**

1.View All Employees

2.Search Employee By ID

3.Insert Employee Record

4.Delete Employee ID

5.Update Employee

6.Exit

**Sample Output 2:**

Employee Details Management Menu.........

1.View All Employees

2.Search Employee By ID

3.Insert Employee Record

4.Delete Employee ID

5.Update Employee

6.Exit

**Please enter Your Option...**

**1**

**Viewing All Employees...**

Employee [employeeId=E001, employeeName=SreeHarsha, employeeAddress=RTNagar, employeePhone=9838388383, employeeSalary=10000, employeeTax=10.3]

Employee [employeeId=E002, employeeName=SumanthKumar, employeeAddress=Indiranagar, employeePhone=9786388383, employeeSalary=12000, employeeTax=12.3]

Employee [employeeId=E003, employeeName=KumareshGupta, employeeAddress=Malleswaram, employeePhone=9876543210, employeeSalary=30000, employeeTax=12.34]

**Do You Wish to Continue yes/no?**

**NO**

**We are out of Menu...**

# Learning Objective:

Student is able to Learn the usage of JDBC APIs to do the following tasks

1) Loading the Driver

2) Establish Connection with MySQL Database

3) Fetch Records from Database

4) Create a Menu Based application with menu options for CRUD & Search operations based on the users input **(With only Read Operation implementation in this exercise and other Operations are implemented in an incremental way in the subsequent Lab & Self Practice exercises)**

5) Use Modular Approach using separate classes for business logic etc.

# 

# 

# Question 2

**Objective: In continuation of Question1 above…**

To enable the Mfg Based business to automate the Data search of **Employees** information for a given EmployeeID

- create Java Based backend modules to perform search Operations from RDBMS using JDBC

**Step 1:** In the Project which is already Created as per Question 1 ,

implement the functionality searchEmployeeById(String empId) in **EmployeeService class**

**Step2:**  In the Project which is already Created as per Question 1 ,

implement the functionality getEmployeeById(String empId) in **EmployeeDAO class**

**Step3:** In the Project which is already Created as per Question 1 ,

Implement the Menu Option for searching the EmployeeBy Id in **EmployeeDataManagement** Class

**Stub Code:** In addition to templates mentioned in Lab Question 1 implement the following in

**EmployeeDAO class**

public class EmployeeDAO {

Connection con;

ResultSet rs;

Statement stmt;

PreparedStatement pstmt;

Employee employee;

ArrayList <Employee> employees;

MyConnection mycon;

public ArrayList <Employee> getAllEmployees()

{

}

public Employee getEmployeeById(String empId)

{

**//IMPLEMENT HERE**

}

public boolean deleteEmployeeById(String empId)

{

}

public boolean updateEmployee(Employee empU)

{

}

public boolean insertEmployee(Employee empI)

{

}

public ArrayList <String> getEmployeeIds()

{

**//IMPLEMENT HERE**

}

}

**EmployeeService class**

public class EmployeeService {

EmployeeDAO edao;

public EmployeeService()

{

}

public ArrayList <Employee> getEmployeeRecords()

{

}

public Employee searchEmployeeById(String empId)

{

**//IMPLEMENT HERE**

}

public boolean insertEmployeeRecord(Employee emplI)

{

}

public boolean updateEmployeeRecord(Employee emplU)

{

}

public boolean deleteEmployeeRecord(String empId)

{

}

public ArrayList <String> getAllEmployeeIds()

{

**//IMPLEMENT HERE**

}

}

//For Menu Options create EmployeeDataManagement Class

public class EmployeeDataManagement

{

public void showMenu()

{

**//IMPLEMENT HERE**

}

}

**//Client Class**

public class ClientUI {

public static void main(String[] args) {

// TODO Auto-generated method stub

EmployeeDataManagement edm = new EmployeeDataManagement();

edm.showMenu();

}

}

**Sample Output 1:**

............. Employee Details Management Menu.........

1.View All Employees

2.Search Employee By ID

3.Insert Employee Record

4.Delete Employee ID

5.Update Employee

6.Exit

**Please enter Your Option...**

**2**

Searching Employee By Id...

**The Employee Ids Are...**

E001

E002

E003

E004

E006

E007

E008

E009

E010

E011

E012

**Please Enter the Employee Id whose record you wish to see...**

E002

The Employee is Employee [employeeId=E002, employeeName=SumanthKumar, employeeAddress=Indiranagar, employeePhone=9786388383, employeeSalary=12000, employeeTax=12.3]

**Do You Wish to Continue yes/no?**

No

We are out of Menu...

**Sample Output 2:**

............. Employee Details Management Menu.........

1.View All Employees

2.Search Employee By ID

3.Insert Employee Record

4.Delete Employee ID

5.Update Employee

6.Exit

**Please enter Your Option...**

2

**Searching Employee By Id...**

The Employee Ids Are...

E001

E002

E003

E004

E006

E007

E008

E009

E010

E011

E012

**Please Enter the Employee Id whose record you wish to see..**.

E000

**Sorry Employee ID Does not Exist....**

**Do You Wish to Continue yes/no?**

No

We are out of Menu...

# 

# Learning Objective:

Students learn to create a Menu Based application with menu options & implement the option for Search operations on Employees table based on the users input of EmployeeId , to fetch the corresponding record from the back end.

# Question 3

# Objective: In continuation of the above questions…

To enable the Mfg Based business to automate the autoGeneration of Employee Id that is to be inserted into Employees Table

- create Java Based back end modules to perform AutoGeneration of Employee Code

**Step 1:** fetch max of employeeId() if suppose it is E014

**Step2:** Split andmake it into 2 String “E” & “014”

**Step3:** Convert second part to numeric “014” to 14

**Step4:** Increment it to 15

**Step5**: Depending upon the digits concatenate it as follows

E+ 015 -> E015

If it is E & 9 → make it E009 and so on

**Stub Code: Along with the Stub** As in Lab Question 1

**EmployeeDAO class**

public class EmployeeDAO {

Connection con;

ResultSet rs;

Statement stmt;

PreparedStatement pstmt;

Employee employee;

ArrayList <Employee> employees;

MyConnection mycon;

public ArrayList <Employee> getAllEmployees()

{

}

public Employee getEmployeeById(String empId)

{

}

public boolean deleteEmployeeById(String empId)

{

}

public boolean updateEmployee(Employee empU)

{

}

public boolean insertEmployee(Employee empI)

{

}

public ArrayList <String> getEmployeeIds()

{

}

public String getMaxEmployeeId()

{

**//IMPLEMENT HERE**

}

}

**EmployeeService class**

public class EmployeeService {

EmployeeDAO edao;

public EmployeeService()

{

}

public ArrayList <Employee> getEmployeeRecords()

{

}

public Employee searchEmployeeById(String empId)

{

}

public boolean insertEmployeeRecord(Employee emplI)

{

}

public boolean updateEmployeeRecord(Employee emplU)

{

}

public boolean deleteEmployeeRecord(String empId)

{

}

public ArrayList <String> getAllEmployeeIds()

{

}

public String fetchMaxEmployeeId()

{

**//IMPLEMENT HERE**

}

}

**//For Menu Options create EmployeeDataManagement Class**

public class EmployeeDataManagement

{

public void showMenu()

{

**//IMPLEMENT HERE**

}

public String generateEmployeeId()

{

**//IMPLEMENT HERE**

}

}

**//Client Class**

public class ClientUI {

public static void main(String[] args) {

// TODO Auto-generated method stub

EmployeeDataManagement edm = new EmployeeDataManagement();

edm.showMenu();

}

}

**Sample Output1:**

1.View All Employees

2.Search Employee By ID

3.Insert Employee Record

4.Delete Employee ID

5.Update Employee

6.Exit

Please enter Your Option...

3

Inserting Employee Record...

**The New Id Generated is E014**

***Note: Insertion of Records will be carried out in Lab Exercise 4, which in turn would use this auto generated ID***

# Learning Objective:

Students learn to create a Menu Based application with menu options & implement the AutoGeneration of EmployeeId which would be used while inserting records.