

EMPLOYEE MANAGEMENT SYSTEM



□ TEAM MEMBERS:

- Vandana
- Hymavathi
- Sruthi
- Muskhan
- Rishika
- Lekha

ABSTRACT

- *Employee Management System is a web-based application designed to streamline and automate various HR processes within organizations. This system aims to enhance efficiency, accuracy, and accessibility of employee information.*
- *The system is built using [mention technologies used, e.g., java servlet , Jdbc ,Jsp] and is designed to be user-friendly and scalable. By automating HR tasks, the Employee Management System aims to improve organizational productivity, reduce administrative burden, and foster a more efficient work environment.*
- **Key features include:**
 - Employee Database
 - Leave Management
 - Attendance
 - Payroll Processing
 - Performance Management

INTRODUCTION

- *Employee performance is defined as how well a person executes their job duties and responsibilities. Many companies assess their employees' performance on an annual or quarterly basis to define certain areas that need improvement and to encourage further success in areas that are meeting or exceeding expectations.*

TOOLS AND TECHNOLOGIES USED

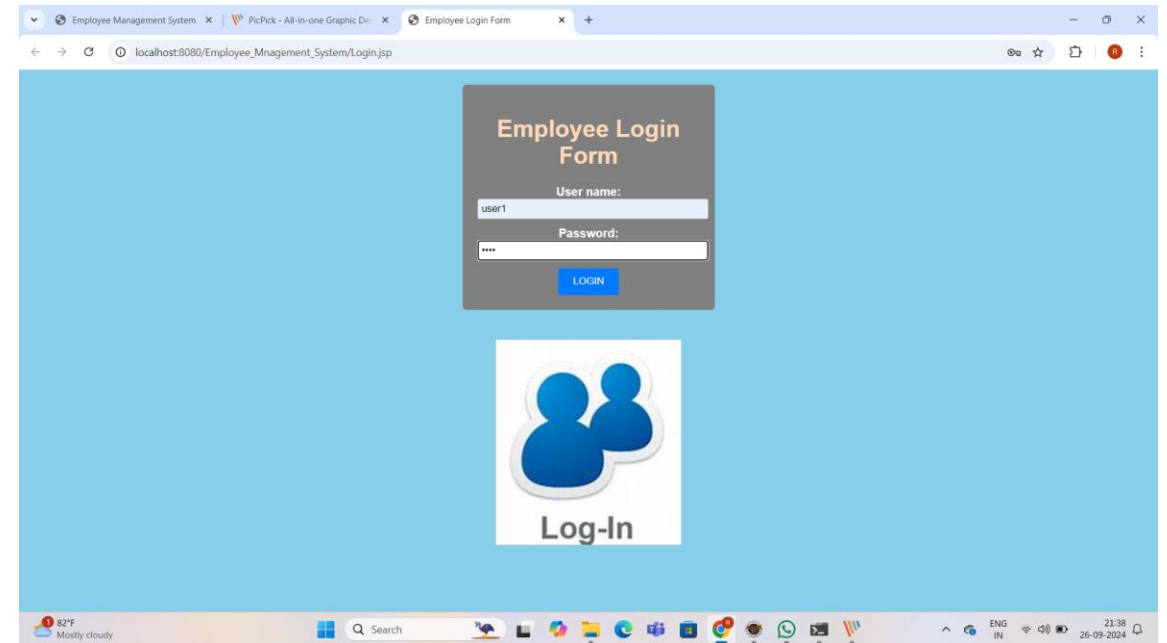
- **Front End:** *Html , Css , Java Script , Jsp*
- **Back End:** *Java Servlet*
- **Database :** *MySQL- mysql-connector-java-8.0.13.jar*
- **Middleware :** *JDBC(Java Database Connectivity)*
- **Server :** *Apache Tomcat*
- **Development Environment :** *Eclipse IDE For java and web applications*
- **Version Control :** *Git/GitHub*

MODULES

- ❖ *USER AUTHENTICATION*
- ❖ *MANAGER MENU*
- ❖ *EMPLOYEE MENU*
- ❖ *ADDING AND UPDATE EMPLOYEE AND EVALUATION DATA*
- ❖ *DISPLAY EMPLOYEE DATA*

❑ USER AUTHENTICATION

- *The program includes a basic login system for both managers and employees. Users are required to enter their usernames and passwords.*
- *Manager and employee login credentials are stored in separate tables: **MANAGER_LOGIN** and **EMPLOYEE_LOGIN**.*



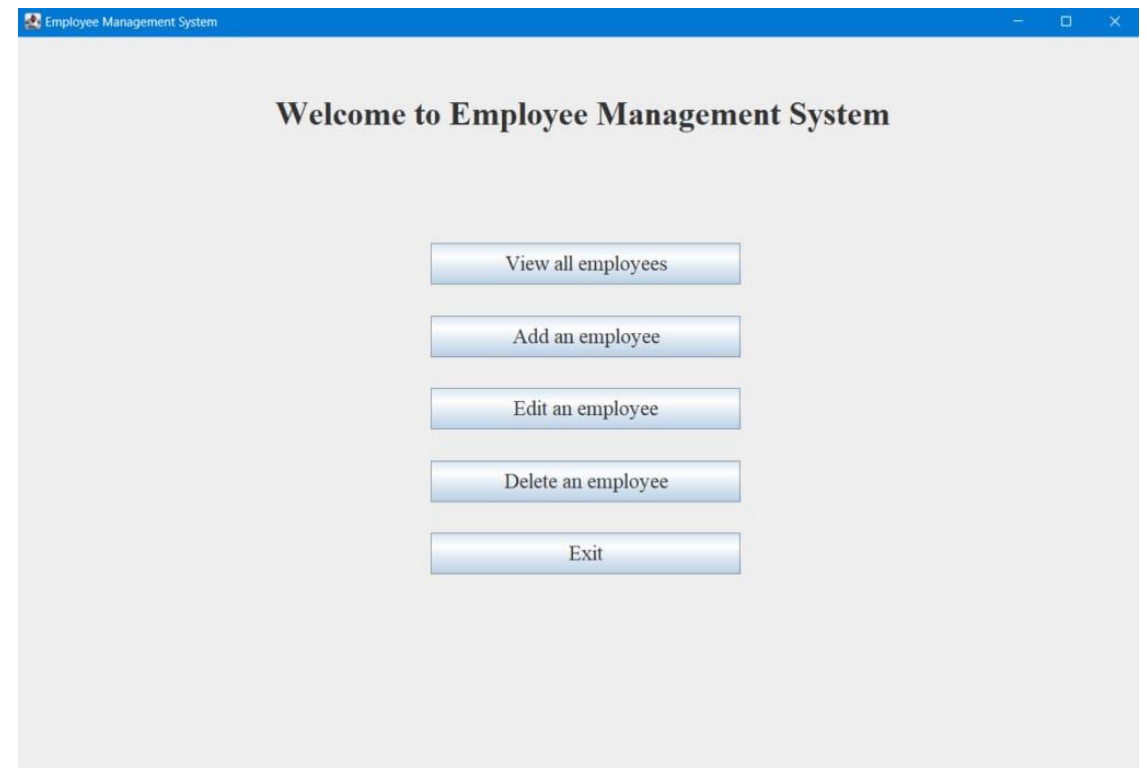
❑ MANAGER MENU

- *After a manager successfully logs in, they have access to a menu with the following options:*
- *Add Employee Details: Allows the manager to add new employee records to the database, including details like name, email, gender, hire date, designation, salary, and mobile number.*
- *Add Evaluation Details: Enables the manager to add performance evaluation details for employees, including evaluation date, rating, and feedback.*



❑ EMPLOYEE MENU

- *After an employee successfully logs in, they have access to a menu with the following options:*
- *Display Employee Performance: Displays a list of employee performance details, including their evaluation date, rating, and feedback.*
- *Logout: Logs the employee out of the system and returns to the main menu.*



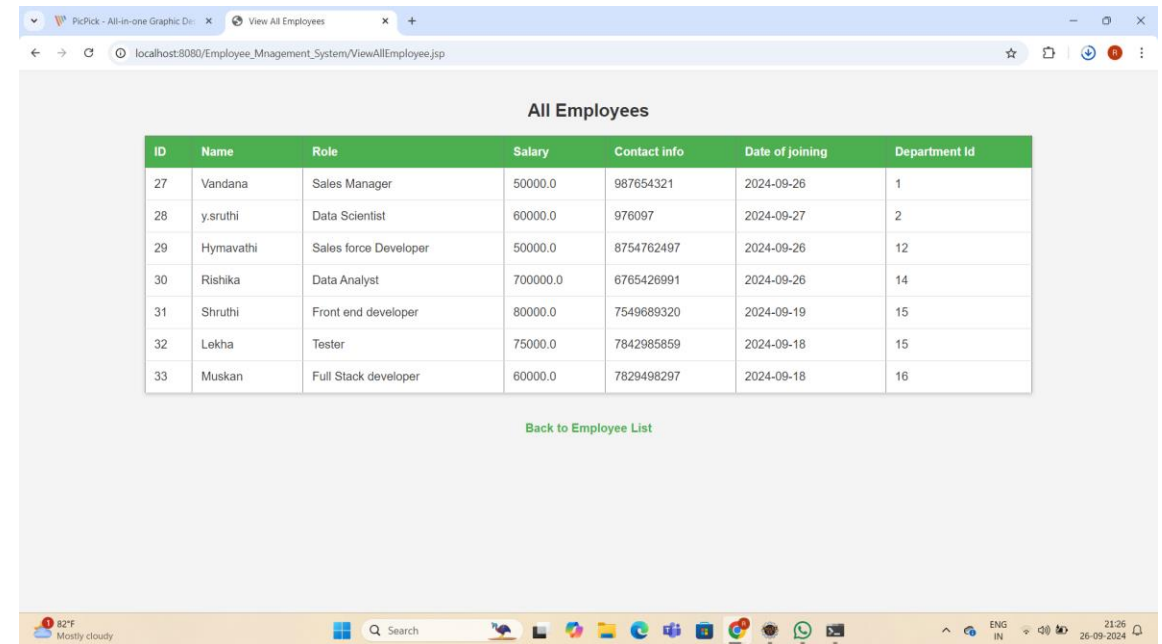
❑ ADDING & UPDATING EMPLOYEE DATA

- *Managers can add new employee and evaluation records to the database.*
- *Managers can update evaluation details, but only for existing records.*
- *Managers can delete employee and evaluation records based on their respective IDs.*

The screenshot displays a web application interface with two main forms on a yellow background. The left form, titled 'Add Employee Details', contains input fields for Name (Vandana), Address (Nellore), Contact Info (972467049), Date Joining (26-09-2024), Job Title (Data Scientist), Salary (80000), and Department ID (12). It includes 'Add Employee', 'Reset', and 'Back' buttons. The right form, titled 'Update Employee Record', features a 'Select Employee ID' dropdown (27), 'New Address' (Chennai), 'New Contact Info' (756389414), and 'Update Employee' and 'Back' buttons. The browser tabs show 'Employee Management System' and 'Add Employee', and the address bar indicates the local development environment.

❑ DISPLAY EMPLOYEE DATA

- *Employees can view their own performance details, including evaluation date, rating, and feedback.*

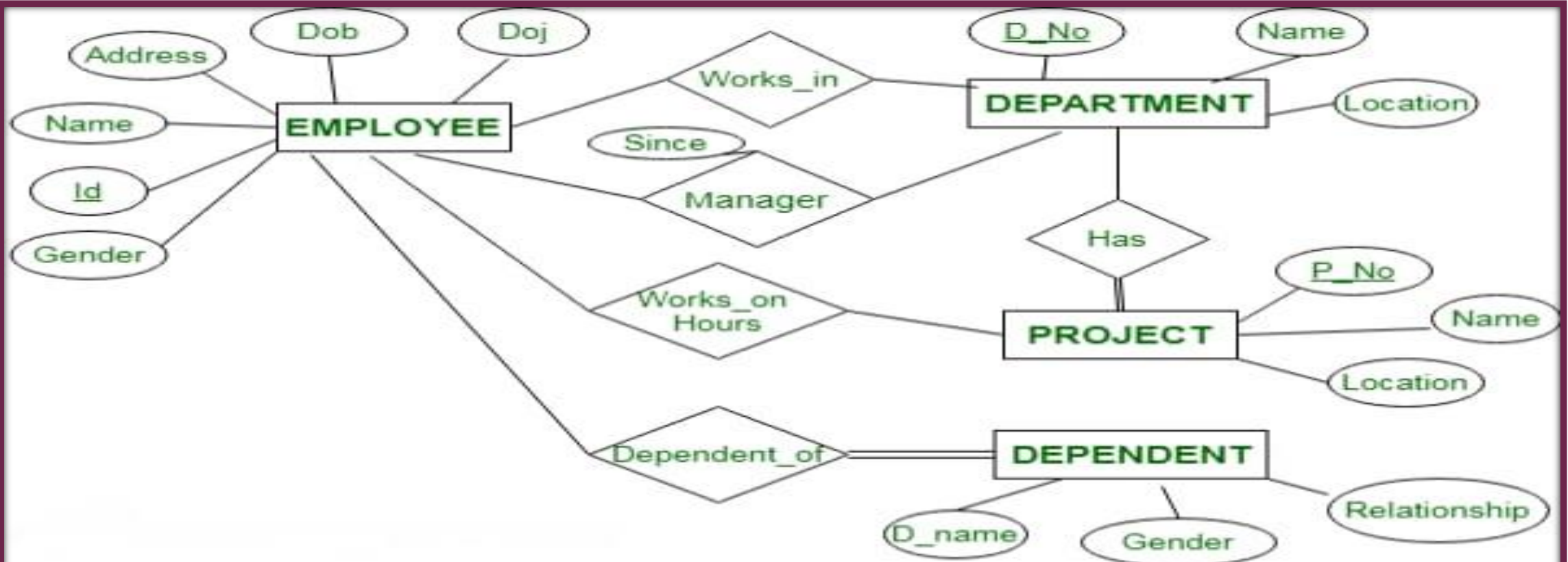


The screenshot shows a web browser window with the title 'View All Employees'. The address bar shows the URL 'localhost:8080/Employee_Management_System/ViewAllEmployee.jsp'. The main content area displays a table titled 'All Employees' with the following data:

ID	Name	Role	Salary	Contact Info	Date of joining	Department Id
27	Vandana	Sales Manager	50000.0	987654321	2024-09-26	1
28	y.sruthi	Data Scientist	60000.0	976097	2024-09-27	2
29	Hymavathi	Sales force Developer	50000.0	8754762497	2024-09-26	12
30	Rishika	Data Analyst	700000.0	6765426991	2024-09-26	14
31	Shruthi	Front end developer	80000.0	7549689320	2024-09-19	15
32	Lekha	Tester	75000.0	7842985859	2024-09-18	15
33	Muskan	Full Stack developer	60000.0	7829498297	2024-09-18	16

Below the table, there is a green link labeled 'Back to Employee List'.

ER DIAGRAM



• VALIDATION

- In my project two validations are there:

- ✓ Salary;

company policy : make sure the data adheres to company policy

Authorization : Ensure that the data is authorized or approved

- ✓ Dept_Id;

In an employee management system , the department ID field can be validated by checking if the department ID entered is valid or not

• TEST CASES

➤ Data Insert checking;

If data is inserted into database then Test case passed successfully. Otherwise test case will be failed.

➤ Data update checking;

If data is update into database then Test case passed successfully. Otherwise test case will be failed.

• **MANAGER LOGIN**

- *CREATE TABLE MANAGER_LOGIN(USERNAME VARCHAR(100), PASSWORD VARCHAR(100));*
- *insert into manager_login values('user', 'user123');*

COLUMNS	DATATYPE	RULE
USERNAME	VARCHAR(100)	NOT NULL
PASSWORD	VARCHAR(100)	NOT NULL

- ***EMPLOYEE LOGIN***

- ***CREATE TABLE EMPLOYEE_LOGIN(Id INT PRIMARY KEY***
- ***AUTO_INCREMENT, USERNAME VARCHAR(100), PASSWORD VARCHAR(100));***

COLUMNS	DATATYPE	RULE
USERNAME	VARCHAR(100)	NOT NULL
PASSWORD	VARCHAR(100)	NOT NULL

• ***EMPLOYEE ATTRIBUTES***

Columns name
1.Emp_id
2.empname
3.address
4.contactinfo
5.date_joining
6.job_title
7.salary

```
create table employee (emp_id int  
primary key auto_increment,  
empname varchar(100),  
address varchar(50),  
contactinfo varchar(100),  
date_joining date,  
job_title varchar(50),  
salary float,  
deptid int  
);
```

- EMPLOYEE DETAILS**

Emp_Id	Emp_name	Address	Contactinfo	Date_Joining	Job_Title	Salary	Deptid
1	Vandana	Nellore	5467829922	2024-09-24	Developer	53000	12
2	Hymavathi	Guntur	3002839442	2024-09-25	Sales Manager	343356	23
3	Shruthi	Bengaluru	9037303342	2024-09-23	Assistant	346571	34
4	Rishika	Belagavi	9287329202	2024-09-11	Project Manager	456772	54
5	Mus Khan	Shivamogga	2457687799	2024-09-04	Executive officer	567893	56
6	Lekha	Vizag	3409765432	2024-09-18	Custom service	678906	67
7	Nandhini	Chennai	2789208890	2024-09-1	Manager	567898	78

EMPLOYEE STATUS

```
CREATE TABLE Attendance ( attendance_id INT PRIMARY KEY  
AUTO_INCREMENT,  
employee_id INT,  
date DATE,  
status VARCHAR(10),  
check_in_time TIME,  
check_out_time TIME,  
);
```

Attendance_id	Employee_id	date	Int_time	Out_time	status
1	101	2024-09-29	09:00:00	17:00:00	Present
2	102	2024-09-27	09:15:00	17:30:00	Absent
3	103	2024-09-26	08:45:00	16:45:00	Late
4	104	2024-09-25	16:23:00	15:31:00	present

• SOURCE CODE

```
package EmployeeManagement;
```

Package is a container which can segregate the files in the java project.

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

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

These import statements are used to include the necessary classes and libraries in your Java program.

```
public class Employee {
```

This line declares the main class of the program named "Employee."

```
static Connection conn;
```

```
static Statement stmt;
```

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

Here, static variables `conn` and `stmt` for database connection and statement, and a Scanner object for user input are declared.

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

The program's main method begins here.

```
boolean loggedIn = false;
```

Declares a boolean variable "loggedIn" to control program flow.

```
while (!loggedIn) {
```

Starts a loop that continues until a successful database connection is established.

```
try {
```

The start of a try block for handling exceptions.

```
Class.forName("com.mysql.cj.jdbc.Driver");
```

This line loads the MySQL JDBC driver class, necessary for database connectivity.

```
Conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/  
emppperf", "root", "root");
```

It establishes a data base connection by using the JDBC to a MySQL database with the URL, username, and password provided.

"jdbc:mysql://":This is the protocol for connecting to a MySQL database.

- "localhost":This is the hostname of the database server. It indicates that the database is running on the local machine.
- ":3306":This is the port number on which MySQL is listening. The default MySQL port is 3306.
- "empperf": This is the name of the specific database you want to connect to. Replace it with the actual name of your database.
- "root" (first occurrence): This is the username used to authenticate the database connection. In this case, you are using the default MySQL username "root." Replace it with the actual username you use to connect to your database.
- "root" (second occurrence): This is the password used to authenticate the database connection. In this case, you are using "root" as the password.
Replace it with the actual password you use to connect to your database.

```
stmt = conn.createStatement();
```

A statement object is created to execute SQL queries on the database.

```
loggedIn = true; //
```

Set loggedIn to true to exit the loop when the connection is established

Sets the "loggedIn" variable to "true" to exit the loop after a successful database connection.

```
catch (ClassNotFoundException |SQLException e) {
```

To check the exception stack trace to see which class is missing and where it is referenced.

```
e.printStackTrace();
```

This line prints the exception's stack trace to the console.

The code continues with a while loop to display a menu of options for the user to interact with the database.

✓The code then defines various methods (add Employee, add Evaluation, Update Evaluation, Display Individual Employee Details, Display all Employee Details, and exit) to perform actions related to employee management and evaluation details. Each method handles user input and interacts with the database accordingly.

✓These methods interact with the database using SQL queries and Prepared Statement objects to perform CRUD (Create, Read, Update, View) operations on employee and evaluation data.

✓Overall, this code provides a basic framework for managing employee information and performance evaluations through a console-based Java application connected to a My SQL database. Users can log in, add employee details, add evaluation details, update evaluations, Display individual employees, Display all employees and evaluations.

❑ CONCLUSION

- ❖ The provided code appears to be the implementation of a basic Employee Performance Evaluation system using Java and a My SQL database.
- ❖ The project allows for two types of users to log in: managers and employees. Managers can perform actions such as adding employee details, adding evaluation details, updating evaluation details, displaying employees, and evaluation details. Employees, on the other hand, can log in to view their performance details.



THANK YOU