

# **EMPLOYEE PAYROLL MANAGEMENT SYSTEM**



**A PROJECT REPORT**

*Submitted by*

**MONESHAA P(8115U23AM028)**

*in partial fulfillment of requirements for the award of the course*

**CGB1201 - JAVA PROGRAMMING**

*In*

**DEPARTMENT OF**

**COMPUTER SCIENCE AND ENGINEERING**

**(ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)**

**K. RAMAKRISHNAN COLLEGE OF ENGINEERING**

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

**SAMAYAPURAM – 621 112**

**DECEMBER - 2024**

**K. RAMAKRISHNAN COLLEGE OF ENGINEERING**

**(Autonomous Institution affiliated to Anna University, Chennai)**

**SAMAYAPURAM-621 112**

**BONAFIDE CERTIFICATE**

Certified that this project report on “**EMPLOYEE PAYROLL MANAGEMENT SYSTEM**” is the bonafide work of **MONESHAA P(8115U23AM028)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

**SIGNATURE**

**MR.B.KIRAN BALA. B.Tech., M.E., M.B.A.,  
Ph.D., M.I.S.T.E., U.A.C.E.E., IAENG**

**HEAD OF THE DEPARTMENT**

Department Of Artificial Intelligence And  
Machine Learning,

K.Ramakrishnan College of Engineering  
(Autonomous)

Samayapuram-621112.

**SIGNATURE**

**Mrs.P. GEETHA M.E.,**

**ASSISTANT PROFESSOR**

Department of Artificial  
Intelligence And Data science,

K.Ramakrishnan College of  
Engineering (Autonomous)

Samayapuram-621112.

Submitted for the End Semester Examination held on .....

**INTERNAL EXAMINER**

**EXTERNAL EXAMINE**

## **DECLARATION**

I jointly declare that the project report on **“EMPLOYEE PAYROLL MANAGEMENT SYSTEM”** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of BACHELOR OF ENGINEERING. This project report is submitted on the partial fulfillment of the requirement of the award of the course **CGB1201- JAVA PROGRAMMING**

**SIGNATURE**

---

MONESHAA P

## ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and indebtedness to our institution, "**K.RAMAKRISHNAN COLLEGE OF ENGINEERING (Autonomous)**", for providing us with the opportunity to do this project.

I extend our sincere acknowledgment and appreciation to the esteemed and honorable Chairman, **Dr. K. RAMAKRISHNAN, B.E.**, for having provided the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director, **Dr.S. KUPPUSAMY, MBA, Ph.D.**, for forwarding our project and offering an adequate duration to complete it .

I would like to thank **Dr. D. SRINIVASAN, M.E., Ph.D., FIE., MIW.,MISTE., MISAE., C. Engg.**, Principal, who gave the opportunity to frame the project to full satisfaction.

I would like to thank **Dr. B. KIRAN BALA, B.Tech., M.E., M.B.A., Ph.D., M.I.S.T.E., U.A.C.E.E., IAENG**, Head of the Department of Artificial Intelligence and Machine Learning, for providing his encouragement in pursuing this project.

I wish to convey our profound and heartfelt gratitude to our esteemed project guide **Mr.P.GEETHA M.E**, Department of Artificial Intelligence and data Science, for her incalculable suggestions, creativity, assistance and patience, which motivated us to carry out this project.

I render our sincere thanks to the Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress

## **INSTITUTE VISION AND MISSION**

### **VISION OF THE INSTITUTE:**

To achieve a prominent position among the top technical institutions.

### **MISSION OF THE INSTITUTE:**

**M1:** To best standard technical education par excellence through state of the art infrastructure, competent faculty and high ethical standards.

**M2:** To nurture research and entrepreneurial skills among students in cutting edge technologies.

**M3:** To provide education for developing high-quality professionals to transform the society.

## **DEPARTMENT VISION AND MISSION**

### **DEPARTMENT OF CSE(ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)**

#### **Vision of the Department**

To become a renowned hub for Artificial Intelligence and Machine Learning Technologies to produce highly talented globally recognizable technocrats to meet Industrial needs and societal expectations.

#### **Mission of the Department**

**M1:** To impart advanced education in Artificial Intelligence and Machine Learning, Built upon a foundation in Computer Science and Engineering.

**M2:** To foster Experiential learning equips students with engineering skills to Tackle real-world problems.

**M3:** To promote collaborative innovation in Artificial Intelligence, machine Learning, and related research and development with industries.

**M4:** To provide an enjoyable environment for pursuing excellence while upholding Strong personal and professional values and ethics.

### **Programme Educational Objectives (PEOs):**

Graduates will be able to:

**PEO1:** Excel in technical abilities to build intelligent systems in the fields of Artificial Intelligence and Machine Learning in order to find new opportunities.

**PEO2:** Embrace new technology to solve real-world problems, whether alone or As a team, while prioritizing ethics and societal benefits.

**PEO3:** Accept lifelong learning to expand future opportunities in research and Product development.

### **Programme Specific Outcomes (PSOs):**

**PSO1:** Ability to create and use Artificial Intelligence and Machine Learning Algorithms, including supervised and unsupervised learning, reinforcement Learning, and deep learning models.

**PSO2:** Ability to collect, pre-process, and analyze large datasets, including data Cleaning, feature engineering, and data visualization..

### **PROGRAM OUTCOMES(POs)**

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
- 3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
- 4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## ABSTRACT

The **Employee Payroll Management System (EPMS)** is a comprehensive Java-based application aimed at automating and enhancing the efficiency of payroll management processes in organizations. It incorporates a secure login system, ensuring only authorized personnel can access the application. The system is designed with modular functionality to address key aspects of payroll operations. The **Employee Management** module allows seamless addition, updating, and maintenance of employee records, providing a centralized and accessible database. The **Salary Calculation** module automates the computation of net salaries by incorporating factors such as allowances, bonuses, overtime, and deductions, thereby eliminating errors associated with manual calculations. This project leverages Java programming concepts, including object-oriented programming, graphical user interfaces (Swing), and event-driven design, to create a robust and user-friendly application. By automating critical payroll tasks, the EPMS not only simplifies payroll handling but also improves overall organizational productivity. The system is scalable, adaptable, and designed to meet the diverse needs of modern workplaces, offering a reliable solution for effective payroll management.



## ABSTRACT WITH POs AND PSOs MAPPING

<b>ABSTRACT</b>	<b>POs MAPPED</b>	<b>PSOs MAPPED</b>
<p>Design and develop an Employee Payroll Management System (EPMS) that automates the payroll process for an organization. The system should calculate and manage employee salaries, deductions, bonuses, and generate payroll reports. The goal is to streamline the payroll process, ensure accuracy in salary calculations, and provide a secure and user- friendly platform for both HR administrators and employees.</p>	<p>PO1,PO2, PO3,PO5, PO9,PO12</p>	<p>PSO1, PSO2</p>

Note: 1- Low, 2-Medium, 3- High

# TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	<b>ABSTRACT</b>	vi
<b>1</b>	<b>INTRODUCTION</b>	
	1.1 Objective	1
	1.2 Overview	1
	1.3 Java Programming concepts	2
<b>2</b>	<b>PROJECT METHODOLOGY</b>	
	2.1 Proposed Work	4
	2.2 Block Diagram	7
<b>3</b>	<b>MODULE DESCRIPTION</b>	
	3.1 Employee Management	8
	3.2 Salary Calculation	8
	3.3 Attendance Management	9
	3.4 Payroll Report Generation	9
<b>4</b>	<b>RESULTS AND DISCUSSION</b>	11
<b>5</b>	<b>CONCLUSION</b>	15
	<b>APPENDIX</b>	16
	<b>REFERENCE</b>	26

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 Objective**

The Employee Payroll Management System (EPMS) is developed with the aim to automate, simplify, and enhance the efficiency of managing payroll within an organization. The objective of this system is to reduce the complexity of manual payroll processing by incorporating features that automate the calculation of employee salaries, considering various components such as basic salary, allowances, bonuses, deductions, overtime, and attendance. By eliminating human error, the system ensures accurate and timely payroll processing, making it an essential tool for organizations of all sizes. Additionally, the system is designed to allow HR administrators to efficiently track employee attendance, manage roles, set bonuses, and monitor deductions, all while ensuring compliance with payroll regulations. The primary goal is to create a robust, reliable, and transparent system that enhances employee satisfaction by providing clear payslips and detailed payroll reports. Moreover, security is ensured through user authentication, and the system is designed to accommodate multiple users with different access levels. This automation allows HR departments to focus on other strategic tasks, thereby increasing operational efficiency within the organization.

### **1.2 Overview**

The Employee Payroll Management System (EPMS) serves as an integrated solution for managing employee payrolls efficiently. The system automates crucial processes such as salary calculations, attendance tracking, bonus adjustments, and the generation of payroll reports and payslips. The interface allows HR administrators to easily add, update, and view employee records, ensuring that the latest information is always available for salary computations. Through the inclusion of essential features like attendance and overtime tracking, bonus and incentive calculations, as well as deductions, the system ensures accurate net salary computation for each employee. Furthermore, the payroll report feature

provides a detailed overview of salary components for all employees, helping organizations maintain transparency and compliance. With the ability to manage and monitor payroll across multiple employees, the system is scalable, user-friendly, and equipped to handle large data sets effectively.

## **1.3 Java Programming Concepts**

### **1. Object-Oriented Programming (OOP)**

- **Classes and Objects:** You used the Employee class to represent the employee's data and methods for calculating salaries. This is a core concept in Java OOP.
- **Encapsulation:** You encapsulated employee data using private variables and provided public methods to access and modify them.
- **Methods and Constructors:** You created methods for operations like calculating salaries, updating records, and displaying employee details.

### **2. Swing and AWT for GUI Development**

- **JFrame, JPanel, and Layouts:** These Swing components are used to build the graphical user interface (GUI). You utilized JFrame to hold the window and JPanel with layout managers (e.g., GridBagLayout and GridLayout) to arrange components like buttons, text fields, and labels.
- **Swing Components:** You used components like JButton, JTextField, JTextArea, JLabel, etc., to interact with the user and display data in the form of input fields, labels, and buttons.

### **3. Event Handling**

- **ActionListener:** You handled user interactions like button clicks using event listeners. For example, when the "Add Employee" button is clicked, it triggers the relevant action to add a new employee to the system.

### **4. Collections (ArrayList)**

- **ArrayList:** You used ArrayList<Employee> to store employee data dynamically. This allows the system to manage a growing number of employees without being restricted by a fixed size, supporting flexible data management and retrieval.

## **5. Basic Calculations and String Handling**

- **Salary Calculations:** You implemented the logic to compute employee salaries, considering factors like basic salary, attendance, overtime, and deductions. This involves basic arithmetic operations for calculating the net salary.
- **String Formatting:** You used `String.format()` to format and display employee details in a readable manner, ensuring a structured output of information

## CHAPTER 2

### PROJECT METHODOLOGY

#### 2.1 Proposed Work

##### 1.Start:

The process begins when the user starts the Employee Payroll Management System (EPMS). This initiates the system, making it ready for user input.

##### 2.Display Main Menu:

The system displays the main menu where users can select the various functionalities they wish to use. These options typically include:

**Add Employee:** Add a new employee to the database.

**Mark Attendance:** Record an employee's attendance.

**Mark Permission:** Mark an employee's absence or permission taken.

**View Monthly Attendance:** View attendance records for the month.

**Exit:** Close the system.

##### 3.User Selection:

The user selects one of the options from the main menu. Based on the selection, the flow of the program branches out to different modules:

**Add Employee:** If the user selects this option, they are prompted to input employee details (e.g., name, ID, salary, etc.), which are then saved in the database.

**Mark Attendance:** If this option is chosen, the system will prompt the user to select an employee and mark their attendance for the day. The system will update the attendance

record based on the input.

**Mark Permission:** If permission is to be marked, the system will update the attendance of the employee to reflect the absence due to permission.

**View Monthly Attendance:** If the user selects this, the system will display the attendance details for all employees for the month, showing the number of days attended and any permissions taken. **Exit:** If the user chooses to exit, the system will terminate, saving any changes made.

#### **4. Employee Management (Add/Modify/Remove):**

For employee-related actions, such as adding a new employee, the system will ask for the necessary details like employee name, ID, and salary. Once entered, the system saves this information and confirms successful addition.

If an employee needs to be updated, the system will prompt the user to search for the employee by ID or name, allowing the user to modify their details (e.g., salary adjustments).

For employee removal, the system will allow the user to search for an employee and then remove them from the payroll system if required.

#### **5. Salary Calculation:**

Once the attendance is marked, the system calculates the salary based on the number of days worked, overtime hours (if applicable), bonuses, and any other factors like deductions. This is done by retrieving attendance records and applying the predefined rules for salary computation.

#### **6. Payslip Generation:**

After salary calculation, the system generates a payslip for each employee. This payslip contains a detailed breakdown of their earnings, including basic salary, overtime, bonuses, deductions, and the net salary to be paid.

## **7. Payroll Report Generation:**

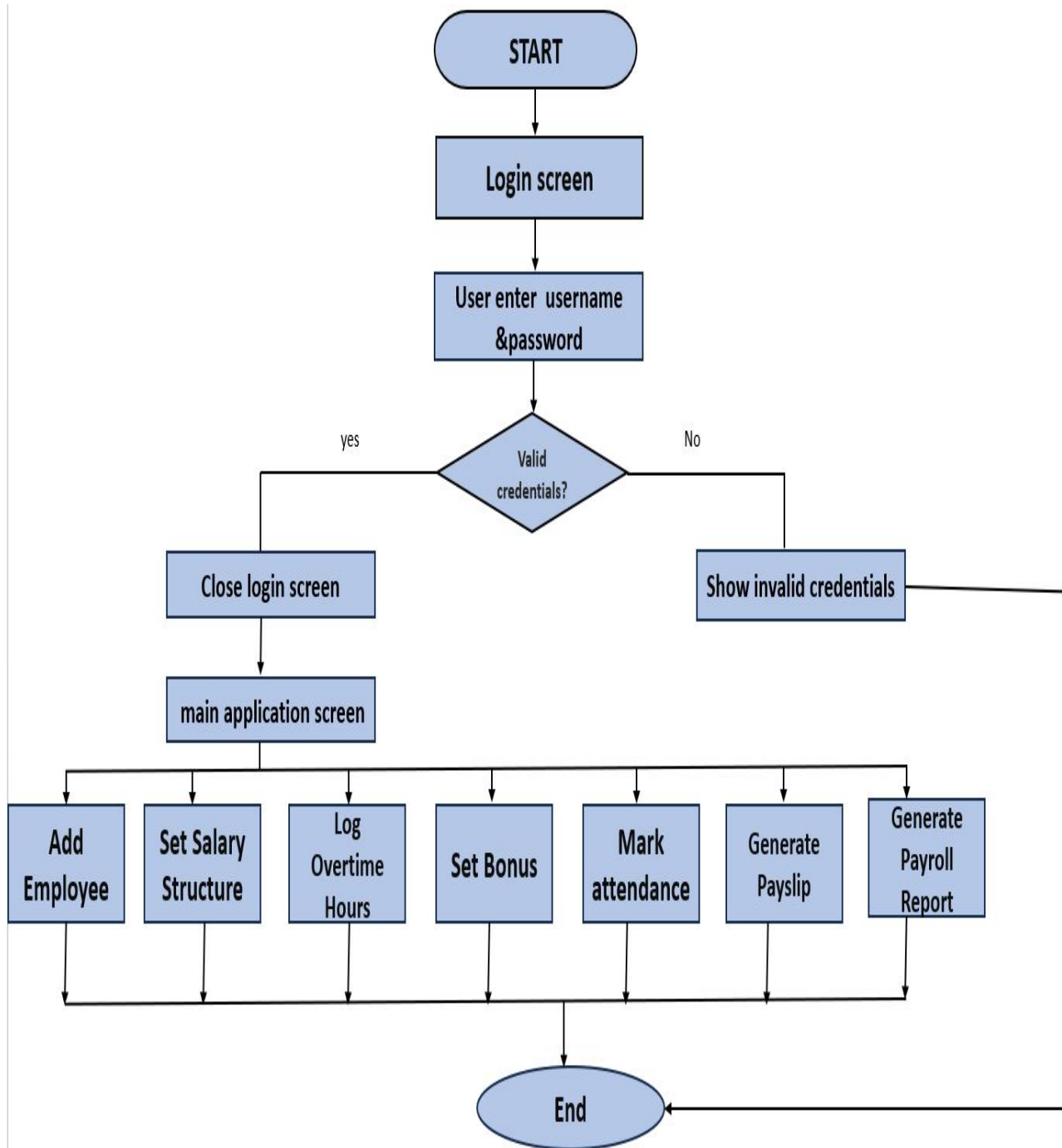
If the user opts to view or generate a payroll report, the system compiles all relevant data for the employees over a defined period (usually monthly) and presents a report summarizing the total salaries, deductions, bonuses, and other payroll information.

## **8. End:**

After the required tasks are completed, the system waits for further user input or exits based on the user's selection.



## 2.2 Block Diagram



## **CHAPTER 3**

### **MODULE DESCRIPTION**

#### **3.1 Employee Management**

**Explanation:** This module is responsible for handling all employee-related tasks. The main functions of this module are:

- **Add Employee:** This functionality allows the system administrator to input details of a new employee, such as name, employee ID, department, salary, and joining date. Once entered, the employee's data is stored in the database.
- **Update Employee Information:** If there is a need to update employee details (e.g., salary changes, department transfers), this function allows modifications to be made.
- **Remove Employee:** Instead of deleting employees entirely from the system, this functionality will allow you to mark an employee as inactive or remove them from the payroll system, maintaining their record for historical reference.
- **Search Employee:** The system enables searching for employee records using various identifiers like employee ID or name.

#### **3.2 Salary Calculation**

**Explanation:** The Salary Calculation module is crucial for ensuring that each employee's salary is accurately calculated based on various parameters:

- **Basic Salary Calculation:** This is the core of salary calculation, which is determined based on the employee's role and contract.
- **Attendance Impact:** This feature calculates salary based on the number of days worked in a month. The system checks attendance records to adjust salary according to present days.
- **Overtime & Bonuses:** If an employee has worked overtime or is eligible for a bonus, this module accounts for additional payments.
- **Deductions:** This includes deductions for tax, insurance, or any other predefined

deductions that apply to the employee.

- **Final Salary Computation:** The module computes the final salary by adding bonuses and overtime while subtracting deductions. This value is then reflected in the payroll system.

### 3.3 Attendance Management

**Explanation:** Attendance Management handles the recording and tracking of employee attendance. It includes:

- **Mark Attendance:** Employees' attendance for the day is recorded here. This can be marked as present, absent, or on leave.
- **Mark Permission:** If an employee needs to take a short leave or permission, this module allows the user to record it.
- **Attendance Review:** This allows the system administrator to review the attendance history of employees, ensuring that all records are up-to-date.
- **Monthly Attendance Report:** The system compiles all the attendance data into a monthly report, summarizing days worked, leaves taken, and permissions granted for each employee.

### 3.4 Payroll Report Generation

**Explanation:** The Payroll Report Generation module helps in generating detailed payroll reports for the entire organization or individual employees:

- **Monthly Payroll Report:** This generates a summary of the salary calculations for the entire month for all employees, including basic salary, overtime, bonuses, deductions, and final payments.
- **Individual Payslips:** The system can generate individual payslips for each employee, providing a breakdown of their earnings and deductions.

- Year-End Summary: At the end of the year, the system can generate a summary of the total salary paid to each employee, including any bonuses, overtime, or adjustments.
- Exporting Data: The system can also provide functionality to export these payroll reports into formats like Excel or PDF for easy sharing or record-keeping.

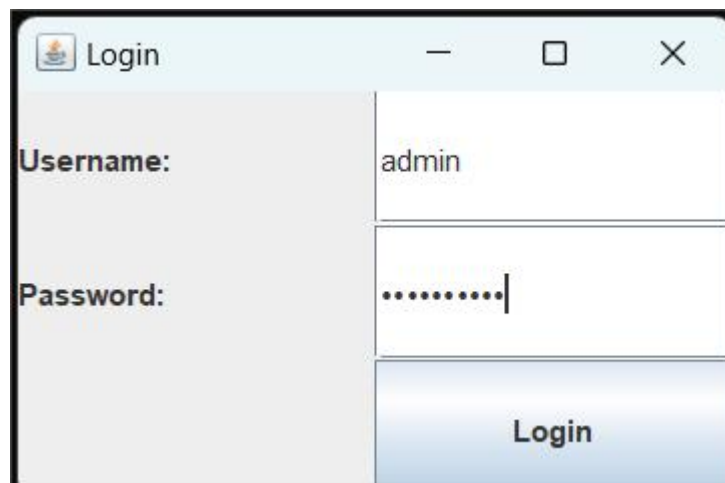
## CHAPTER 4

### RESULTS AND DISCUSSION

#### 4.1 Login Screen

**Result:**

The login screen successfully authenticates users based on predefined credentials (username: "admin", password: "password"). The system verifies the login credentials before granting access to the main application. This ensures that only authorized users can interact with the payroll system. **Discussion:** The login functionality ensures a secure access point to the system, promoting user authentication and preventing unauthorized usage.



#### 4.2 Employee Management

**Result:** The Employee Management module allows users to add, view, and update employee details such as name, employee ID, and salary. The system stores these details and displays confirmation after adding a new employee.

**Discussion:** This feature enables efficient management of employee data, ensuring that relevant employee information is easily accessible for payroll calculation and reporting. It minimizes errors in manual record-keeping.

Employee Payroll System

Name:

Employee ID:

Salary:

Attended Days (0-30):

Overtime Hours:

Bonus:

Allowances:

Deductions:

Employee added: k.abinaya

Buttons: Add Employee, Set Salary, Log Overtime, Set Bonus, Mark Attendance, Generate Payslip, Generate Report

### 4.3 Salary Calculation

**Result:** The salary calculation feature allows the user to input the salary components like basic salary, allowances, deductions, and overtime. It calculates the net salary based on attendance, overtime, and other factors.

**Discussion:** By automating salary calculations, this module reduces errors and saves time in payroll processing, ensuring accurate payments and better transparency for employees.

Employee Payroll System

Name:

Employee ID:

Salary:

Attended Days (0-30):

Overtime Hours:

Bonus:

Allowances:

Deductions:

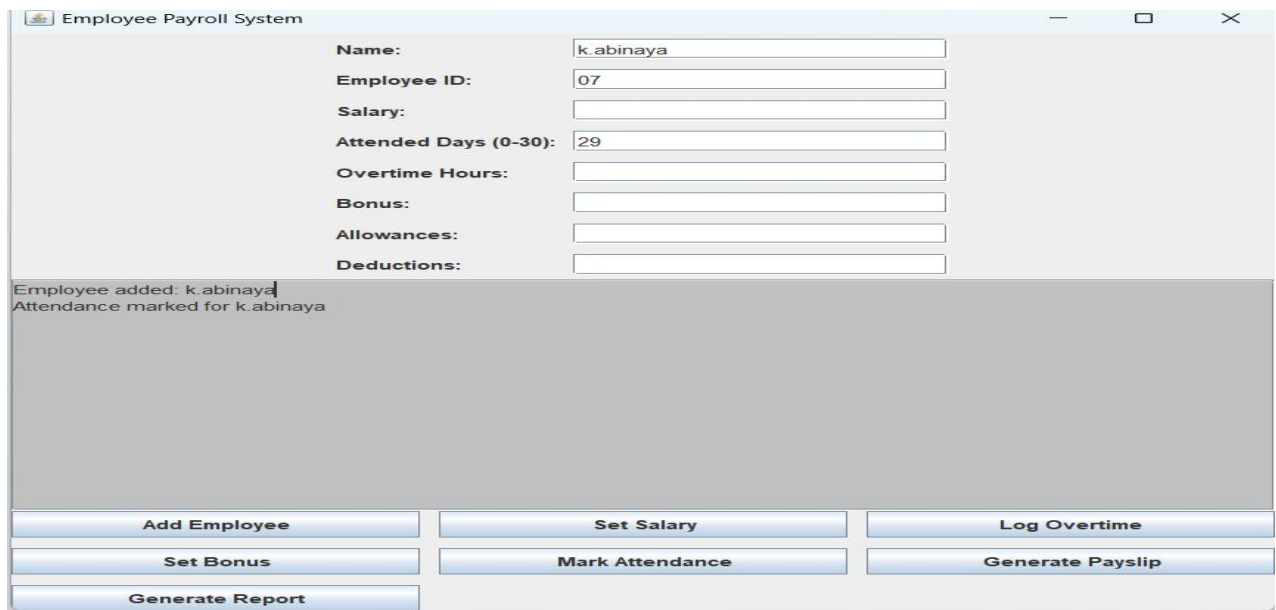
Salary set for Alice

Buttons: Add Employee, Set Salary, Log Overtime, Set Bonus, Mark Attendance, Generate Payslip, Generate Report

## 4.4 Attendance Management

**Result:** The attendance management functionality allows users to mark the number of days an employee attended work. It computes the attendance deduction based on the employee's salary and the number of days worked.

**Discussion:** Accurate attendance tracking is crucial for payroll systems. This feature ensures that deductions are made based on actual attendance, helping to maintain fairness in the payroll process.



The screenshot displays the 'Employee Payroll System' window. It features a form with the following fields: Name (k.abinaya), Employee ID (07), Salary (empty), Attended Days (0-30) (29), Overtime Hours (empty), Bonus (empty), Allowances (empty), and Deductions (empty). Below the form, a status bar indicates 'Employee added: k.abinaya' and 'Attendance marked for k.abinaya'. At the bottom, there are seven buttons: 'Add Employee', 'Set Salary', 'Log Overtime', 'Set Bonus', 'Mark Attendance', 'Generate Payslip', and 'Generate Report'.

## 4.5 Payroll Report Generation

**Result:** This module generates detailed payslips for each employee, displaying their salary breakdown, including basic salary, allowances, deductions, overtime pay, and attendance deductions.

**Discussion:** The payroll report generation feature provides transparency, enabling employees to understand the components of their salary. It also assists the administration in creating reports for audit and compliance purposes.

Employee Payroll System

Name:

Employee ID:

Salary:

Attended Days (0-30):

Overtime Hours:

Bonus:

Allowances:

Deductions:

Payslip for Bob:

ID: 2    Name: Bob    Basic Salary: 50000.00 Allowances: 0.00 Deductions: 0.00 Bonuses: 0.00 Overtime: 0.00 Attendance Deduction: 50000.00 Net Salary: 0.00

Payroll Report:

ID: 1    Name: Alice    Basic Salary: 60000.00 Allowances: 0.00 Deductions: 0.00 Bonuses: 0.00 Overtime: 0.00 Attendance Deduction: 60000.00 Net Salary: 0.00  
ID: 2    Name: Bob    Basic Salary: 50000.00 Allowances: 0.00 Deductions: 0.00 Bonuses: 0.00 Overtime: 0.00 Attendance Deduction: 50000.00 Net Salary: 0.00  
ID: 3    Name: Carla    Basic Salary: 70000.00 Allowances: 0.00 Deductions: 0.00 Bonuses: 0.00 Overtime: 0.00 Attendance Deduction: 70000.00 Net Salary: 0.00  
ID: 4    Name: David    Basic Salary: 75000.00 Allowances: 0.00 Deductions: 0.00 Bonuses: 0.00 Overtime: 0.00 Attendance Deduction: 75000.00 Net Salary: 0.00  
ID: 5    Name: Eve    Basic Salary: 65000.00 Allowances: 0.00 Deductions: 0.00 Bonuses: 0.00 Overtime: 0.00 Attendance Deduction: 65000.00 Net Salary: 0.00

Add Employee

Set Salary

Log Overtime

Set Bonus

Mark Attendance

Generate Payslip

Generate Report



## **CHAPTER 5**

### **CONCLUSION**

In conclusion, the Employee Payroll Management System (EPMS) project demonstrates how Java can be effectively utilized to automate and streamline the payroll processing tasks in an organization. By integrating key features such as employee management, salary calculations, attendance tracking, payroll report generation, and allowances/bonuses handling, the system offers an efficient solution for managing employee payroll operations. The use of Swing for the GUI enables a user-friendly interface, making the system accessible to non-technical users.

The project highlights several fundamental concepts in Java, including object-oriented programming (OOP) principles such as encapsulation, inheritance, and polymorphism, as well as practical use of collections and event-driven programming with Swing components. Additionally, by using action listeners and integrating data management through ArrayLists, the system becomes scalable for more employees and complex payroll features.

By following the design principles laid out in the project, businesses can save time and reduce errors associated with manual payroll calculations. Furthermore, the modular structure of the system ensures that additional features, such as reporting and payroll adjustments, can be easily integrated in the future.

The development of the EPMS also serves as an excellent example of how Java can be used in a practical, real-world application, reinforcing the importance of thorough testing, error handling, and user interface design. It is a reflection of how digital technologies can enhance productivity in administrative tasks, enabling better decision-making and fostering a transparent, efficient work environment.

## APPENDIX

### (Coding)

```
Import javax.swing.*; import java.awt.*;
Import java.awt.event.ActionEvent;
import java.awt.event.ActionListener; import java.util.ArrayList;
import java.util.List
class Employee {

    String name, employeeId;
    double basicSalary, allowances = 0, deductions = 0, bonuses = 0,
    overtimeHours = 0; int attendedDays = 0, totalWorkingDays = 30;

    public Employee(String name, String employeeId, double
        basicSalary) { this.name = name;
        this.employeeId = employeeId;
        this.basicSalary = basicSalary;
    }

    public double calculateNetSalary() {
        double dailySalary = basicSalary / totalWorkingDays;
        double attendanceDeduction = (totalWorkingDays - attendedDays)
        * dailySalary; double overtimePay = overtimeHours * (basicSalary
        / 160) * 1.5;
        return basicSalary + allowances + bonuses + overtimePay -
attendanceDeduction - deductions;
    }

    @Override
```

```

public String toString() {

    return String.format("ID: %-10s Name: %-20s Basic Salary: %.2f
Allowances: %.2f Deductions: %.2f Bonuses: %.2f Overtime: %.2f Attendance
Deduction: %.2f Net Salary: %.2f",
employeeId, name, basicSalary, allowances, deductions, bonuses, overtimeHours
    (totalWorkingDays - attendedDays) * (basicSalary / totalWorkingDays),
    calculateNetSalary());
}
}

```

```

public class EmployeePayrollSystemSwing extends
    JFrame { private static List<Employee> employees =
        new ArrayList<>(); private JTextArea outputArea;
        private JTextField nameField, idField, salaryField, daysField, overtimeField,
        bonusField, allowanceField, deductionField;

```

```

    public EmployeePayrollSystemSwing() {
        // Adding 5 sample employees
        employees.add(new Employee("Alice", "1", 60000));
        employees.add(new Employee("Bob", "2", 50000));
        employees.add(new Employee("Carla", "3", 70000));
        employees.add(new Employee("David", "4", 75000));
        employees.add(new Employee("Eve", "5", 65000));

        setTitle("Employee Payroll
System"); setSize(700, 600);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
    }
}

```

```

// Background Color
getContentPane().setBackground(Color.CYAN);

// Create panels for layout
JPanel inputPanel = new JPanel();
inputPanel.setLayout(new GridBagLayout()); // Use GridBagLayout for
better alignment GridBagConstraints gbc = new GridBagConstraints();
gbc.fill =
GridBagConstraints.HORIZONTAL;
gbc.insets = new Insets(5, 5, 5, 5);

// Labels and Text fieldsJLabel
nameLabel = new JLabel("Name:");
nameField = new JTextField(20);
JLabel idLabel = new JLabel("Employee
ID:"); idField = new JTextField(20);
JLabel salaryLabel = new
JLabel("Salary:"); salaryField = new
JTextField(20);

JLabel daysLabel = new JLabel("Attended Days (0-
30:"); daysField = new JTextField(5);
JLabel overtimeLabel = new JLabel("Overtime
Hours:"); overtimeField = new JTextField(5);
JLabel bonusLabel = new
JLabel("Bonus:"); bonusField = new
JTextField(5);
JLabel allowanceLabel = new JLabel("Allowances:");

```

```

allowanceField = new JTextField(5);
JLabel deductionLabel = new JLabel("Deductions:");
deductionField = new JTextField(5);

// Action Buttons
JButton addEmployeeButton = new JButton("Add
Employee"); JButton setSalaryButton = new
JButton("Set Salary");
JButton logOvertimeButton = new JButton("Log
Overtime"); JButton setBonusButton = new
JButton("Set Bonus");
JButton markAttendanceButton = new JButton("Mark
Attendance"); JButton generatePayslipButton = new
JButton("Generate Payslip"); JButton
generateReportButton = new JButton("Generate
Report");

// Setting action listeners for buttons
addEmployeeButton.addActionListener(new
ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e)

        { addEmployee();
        }
});

setSalaryButton.addActionListener(new

```

```
        ActionListener() { @Override
        public void
            actionPerformed(ActionEvent e)
            { setSalary();
            }
        });
```

```
logOvertimeButton.addActionListener(new
    ActionListener() { @Override
    public void actionPerformed(ActionEvent e)
        { logOvertime();
        }
    });
```

```
setBonusButton.addActionListener(new
    ActionListener() { @Override
    public void
        actionPerformed(ActionEvent e)
        { setBonus();
        }
    });
```

```
markAttendanceButton.addActionListener(new
    ActionListener() { @Override
    public void actionPerformed(ActionEvent e)
        { markAttendance();
        }
    });
generatePayslipButton.addActionListener(new
    ActionListener() { @Override
```

```

        public void actionPerformed(ActionEvent e)
        { generatePayslip();
        }
    });

generateReportButton.addActionListener(new
    ActionListener() { @Override
        public void actionPerformed(ActionEvent e)
        { generatePayrollReport();
        }
    });

// Adding components to input
panel gbc.gridx = 0;
gbc.gridy = 0;
inputPanel.add(nameLabel,
gbc); gbc.gridx = 1;
inputPanel.add(nameField,
gbc);

gbc.gridx = 0;
gbc.gridy = 1;
inputPanel.add(idLabel,
gbc); gbc.gridx = 1;
inputPanel.add(idField,
gbc);

gbc.gridx = 0;
gbc.gridy = 2;
inputPanel.add(salaryLabel,

```

```
gbc); gbc.gridx = 1;
inputPanel.add(salaryField,
gbc);

gbc.gridx = 0;
gbc.gridy = 3;
inputPanel.add(daysLabel, gbc);
gbc.gridx = 1;
inputPanel.add(daysField, gbc);

gbc.gridx = 0;
gbc.gridy = 4;
inputPanel.add(overtimeLabel,
gbc); gbc.gridx = 1;
inputPanel.add(overtimeField,
gbc);

gbc.gridx = 0;
gbc.gridy = 5;
inputPanel.add(bonusLabel,
gbc); gbc.gridx = 1;
inputPanel.add(bonusField,
gbc);

gbc.gridx = 0;
gbc.gridy = 6;
inputPanel.add(allowanceLabel,
gbc); gbc.gridx = 1;
inputPanel.add(allowanceField,
gbc);
```



```

gbc.gridx = 0;
gbc.gridy = 7;
inputPanel.add(deductionLabel,
gbc); gbc.gridx = 1;
inputPanel.add(deductionField,
gbc);

// Buttons panel
JPanel buttonsPanel = new JPanel(new GridLayout(3, 2, 10, 10));
buttonsPanel.add(addEmployeeButton); buttonsPanel.add(setSalaryButton);
buttonsPanel.add(logOvertimeButton); buttonsPanel.add(setBonusButton);
buttonsPanel.add(markAttendanceButton);
buttonsPanel.add(generatePayslipButton);
buttonsPanel.add(generateReportButton); // Text area for output
outputArea = new JTextArea(10, 50);
outputArea.setEditable(false);
outputArea.setBackground(Color.LIGHT_GRAY);
JScrollPane scrollPane = new
JScrollPane(outputArea);

// Layout the input panel and text area
add(inputPanel, BorderLayout.NORTH);
add(scrollPane,
BorderLayout.CENTER);
add(buttonsPanel,
BorderLayout.SOUTH);

setLocationRelativeTo(null);
}

private void addEmployee() {

```

```

String name =
nameField.getText(); String id
= idField.getText();
double salary =
Double.parseDouble(salaryField.getText());
employees.add(new Employee(name, id, salary));
outputArea.append("Employee added: " + name +
"\n");
}

```

```

private void setSalary()
{
    String id =
idField.getText();
Employee employee =
findById(id);
if
(employee != null) {
    double salary = Double.parseDouble(salaryField.getText());
    double allowance =
Double.parseDouble(allowanceField.getText());
    double deduction =
Double.parseDouble(deductionField.getText());
    employee.basicSalary = salary;
    employee.allowances =
allowance;
    employee.deductions =
deduction;
    outputArea.append("Salary set for " + employee.name + "\n");
}
}

```

```

private void logOvertime()
{
    String id =
    idField.getText();
    Employee employee =
    findEmployeeById(id); if
    (employee != null) {
        double overtime =
        Double.parseDouble(overtimeField.getText());
        employee.overtimeHours += overtime;
        outputArea.append("Overtime logged for " +
        employee.name + "\n");
    }
}

```

```

private void setBonus()
{
    String id =
    idField.getText();
    Employee employee =
    findEmployeeById(id); if
    (employee != null) {
        double bonus =
        Double.parseDouble(bonusField.getText());
        employee.bonuses = bonus;
        outputArea.append("Bonus set for " + employee.name + "\n");
    }
}

```

```

private void
markAttendance()
{
    String id =

```

```

idField.getText();
Employee employee =
findEmployeeById(id); if
(employee != null) {
    int days = Integer.parseInt(daysField.getText());
    employee.attendedDays = days;
    outputArea.append("Attendance marked for " + employee.name + "\n");
}
}

```

```

private void
generatePayslip() { String
id = idField.getText();

Employee employee =
findEmployeeById(id); if
(employee != null) {
    outputArea.append("Payslip for " + employee.name + ":\n" +
employee.toString() +
"\n\n");
}
}

```

```

private void generatePayrollReport()
{ outputArea.append("Payroll
Report:\n"); for (Employee
employee : employees) {
    outputArea.append(employee.toString() + "\n");
}
}

```

```

private Employee
    findEmployeeById(String id) { for
    (Employee e : employees) {
        if (e.employeeId.equals(id))
            { return e;
            }
        }
    }
    outputArea.append("Employee with ID " + id + " not
    found.\n"); return null;
}

public static void main(String[] args)
{ SwingUtilities.invokeLater(new Runnable() {
    @Override
    public void run() {
        // Show login screen
        first new
        LoginScreen();
    }
});
}
static class LoginScreen
{ LoginScreen() {
    JFrame loginFrame = new JFrame("Login");
    loginFrame.setSize(300, 200);
    loginFrame.setDefaultCloseOperation(JFrame.EXIT_ON_
    CLOSE);

    JPanel loginPanel = new JPanel();
    loginPanel.setLayout(new GridLayout(3, 2));
}
}

```

```

JLabel usernameLabel = new
JLabel("Username:"); JTextField
usernameField = new JTextField(20); JLabel
passwordLabel = new JLabel("Password:");
JPasswordField passwordField = new

JPasswordField(20); JButton loginButton = new

JButton("Login");

loginPanel.add(username
Label);
loginPanel.add(username
Field);
loginPanel.add(password
Label);
loginPanel.add(password
Field);
loginPanel.add(new JLabel()); // Empty cell
for alignment loginPanel.add(loginButton);

loginFrame.add(loginPanel);
loginFrame.setLocationRelativeTo(null);
loginFrame.setVisible(true);

loginButton.addActionListener(new
ActionListener() { @Override
public void actionPerformed(ActionEvent

```

```

e) { String username =
usernameField.getText();
String password = new String(passwordField.getPassword());

// Simple validation (you can improve this)

if ("admin".equals(username) && "password".equals(password))
    { loginFrame.dispose(); // Close login screen
      new EmployeePayrollSystemSwing().setVisible(true); // Show main
        application
    } else {
        JOptionPane.showMessageDialog(loginFrame, "Invalid username or
        password.");
    }
}
});
}
}
}

```

## REFERENCES:

### ☐ Books:

- "Java: The Complete Reference" by Herbert Schildt (9th Edition). McGraw-Hill Education.
- "Head First Java" by Kathy Sierra and Bert Bates. O'Reilly Media.
- "Core Java Volume I – Fundamentals" by Cay S. Horstmann. Prentice Hall.
- "Java Programming" by Joyce Farrell. Cengage Learning.

### ☐ Websites:

- [Oracle Java Documentation](#) – Official tutorials for Java programming concepts.
- GeeksforGeeks - Java Tutorials – A comprehensive source for learning Java and understanding its key concepts.
- W3Schools Java Tutorial – An easy-to-understand guide to Java programming and concepts.
- [Stack Overflow](#) – A helpful community for troubleshooting Java issues and finding solutions.

### ☐ Additional Resources:

- TutorialsPoint - Java Swing – A practical guide to building GUI applications with Java Swing.
- [Java2s.com](#) – Offers examples and tutorials for Java development across different topics.