EMPLOYEE PAYROLL MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

MAADHAV B (2303811710421086)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

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

SAMAYAPURAM – 621 112

NOVEMBER- 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **"EMPLOYEE PAYROLL MANAGEMENT SYSTEM "** is the bonafide work of **MAADHAV B (2303811710421086)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Dr.A.DELPHIN CAROLINA RANI, M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR

CGB1201-JAVA PROGRAMMING
Mrs.K.VALLI PRIYADHARSHINI, M.E.,(Ph.D.)
SUPERVISOR
ASSISTANTP ROFESSOR

SIGNATURE	**SIGNATURE**
Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,	Mrs.K.Valli Priyadharshini, M.E.,(Ph.D.,),
HEAD OF THE DEPARTMENT	**SUPERVISOR**
PROFESSOR	ASSISTANT PROFESSOR
Department of CSE	Department of CSE
K.Ramakrishnan College of Technology (Autonomous)	K.Ramakrishnan College of Technology (Autonomous)
Samayapuram–621112.	Samayapuram–621112.

Submitted for the viva-voce examination held on 03.12.2024

CGB1201-JAVA PROGRAMMING
Mr.MAHARMANNAN A, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

CGB1201-JAVA PROGRAMMING
Mrs.D. ARUNA PRIYA, M.E.,
EXTERNAL EXAMINER
ASSISTANT PROFESSOR
8104-DSEC, PERAMBALUR.

INTERNAL EXAMINER EXTERNAL EXAMINER

DECLARATION

I 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 fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING.**

.

Signature

MAADHAV B

Place: Samayapuram

Date: 03.12.2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for 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 to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mrs. K. VALLI PRIYADHARSHINI, M.E., (Ph.D.,),** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to 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.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

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 is a Java-based application built using the AWT Abstract Window Toolkit framework. It provides an intuitive graphical user interface for managing employee payroll data, including adding employee details, calculating net salaries, and generating payroll reports. The system features functionalities for adding employee records with details like ID, name, basic salary, deductions, and bonuses. It also allows users to search for employees by ID and generate a comprehensive payroll report for all employees. The application employs object-oriented principles, with a robust Employee class for encapsulating payroll logic and a Payroll Processor class for managing employee records. This system is designed for ease of use, with clear input fields, action buttons, and an output area to display results, making it an effective tool for small to medium-sized organizations to streamline payroll management processes.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The Employee Payroll Management System (EPMS) is designed to automate the payroll process for organizations. It calculates and manages employee salaries, deductions, bonuses, and generates payroll reports. EPMS aims to streamline payroll operations, ensure accuracy in salary computations, and provide a secure, user-friendly interface for HR administrators and employees.	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -3** **PO5 -3** **PO6 -3** **PO7 -3** **PO8 -3** **PO9 -3** **PO10 -3** **PO11-3** **PO12 -3**	**PSO1 -3** **PSO2 -3** **PSO3 -3**

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

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1Objective

The primary objective of this Java program is to create a functional Employee Payroll Management System using the AWT (Abstract Window Toolkit) framework for its graphical user interface (GUI). This system is designed to streamline the process of managing employee information and payroll computations. It allows users to input employee details, such as ID, name, basic salary, deductions, and bonuses, and efficiently calculates the net salary for each employee.The program provides functionality to add new employees to a centralized list, view a detailed payroll report for all employees, and search for specific employees using their ID. The payroll report includes each employee's ID, name, and calculated net salary, ensuring that all essential payroll details are readily accessible.

1.2Overview

This Java program implements an Employee Payroll Management System using AWT (Abstract Window Toolkit) for its graphical user interface (GUI). It consists of three main components: the Employee class, which represents individual employees with attributes like ID, name, basic salary, deductions, and bonuses, and includes a method to calculate net salary based on these values; the PayrollProcessor class, which manages a list of employees, providing functionalities to add employees, generate a payroll report summarizing each employee's details and net salary, and search for employees by their unique ID; and the EmployeePayrollSystemAWT class, which serves as the main GUI for user interaction.

1.3 Java Programming Concepts

- **Object-Oriented Programming (OOP)**

  The program is built using object-oriented principles, organizing functionality into distinct classes and objects. The Employee class encapsulates employee details and methods like salary calculation, while the PayrollProcessor class manages the collection of employees and their payroll data. The design promotes reusability, modularity, and clarity, aligning with OOP's core principles. Inheritance is demonstrated by extending the Frame class for GUI functionality, while method overriding is used to customize behavior such as the toString() method in the Employee class.

- **Collections Framework**

  The program leverages Java's ArrayList to store and manage employee records dynamically. The collection allows efficient addition, iteration, and searching of employee data. It simplifies handling a variable number of employees without needing fixed-size data structures, providing flexibility and scalability to the system.

- **Exception Handling**

  Robust exception handling is incorporated to ensure the system can handle invalid or unexpected input gracefully. For example, the program uses try-catch blocks to catch NumberFormatException when parsing user inputs from text fields. This ensures a seamless user experience and prevents the application from crashing due to incorrect data entry.

- **AWT (Abstract Window Toolkit)**

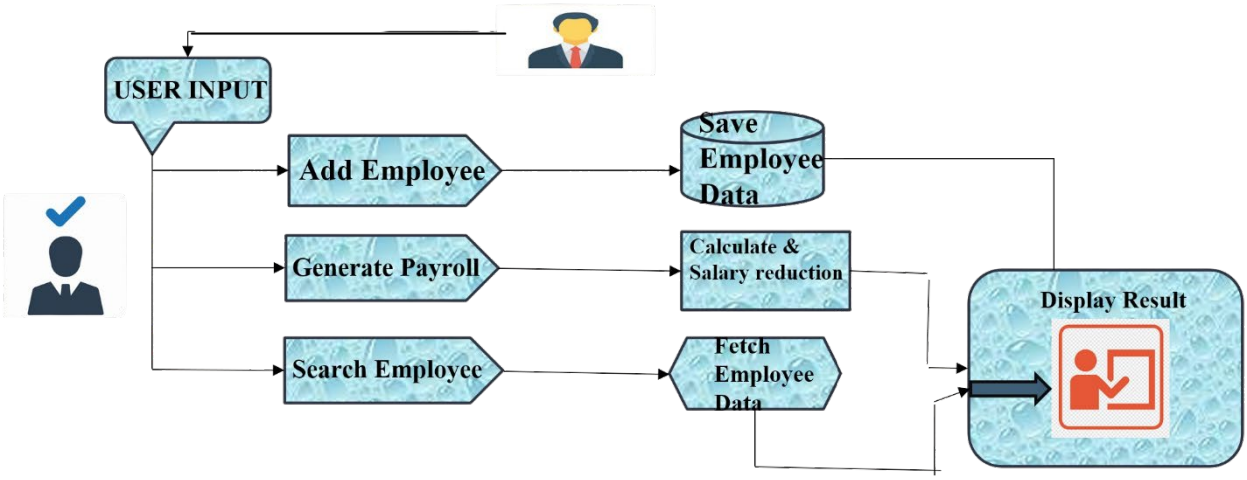
  The graphical user interface is created using AWT components such as TextField, Label, Button, and TextArea. These elements enable interactive input and output, providing users with a visual and intuitive way to interact with the system.

CHAPTER 2
PROJECT METHODOLOGY

2.1Proposed Work

The proposed work focuses on creating a system to streamline the process of managing employee records and payroll operations for small organizations. It aims to provide a user-friendly interface where users can easily input employee details such as identification numbers, names, and financial specifics, including salaries, deductions, and bonuses. The system will calculate net salaries automatically, generate detailed payroll reports, and allow users to search for specific employee details quickly. By simplifying these processes, the system seeks to enhance efficiency, reduce manual effort, and minimize errors in payroll management. It is designed to be intuitive and accessible, making it suitable for individuals with minimal technical expertise while meeting the organizational need for reliable record-keeping and financial accuracy.

2.2 Block Diagram



CHAPTER 3
MODULE DESCRIPTION

3.1 Employee Management Module

This module is responsible for managing employee information, including adding storing, and retrieving employee records. It ensures that details such as employee ID, name, basic salary, deductions, and bonuses are accurately captured and maintained for processing

3.2 Payroll Calculation Module

The payroll calculation module handles the computation of net salaries for employees. It calculates the net salary by taking into account the basic salary, deductions, and bonuses, ensuring precise financial data for payroll processing.

3.3 Report Generation Module

This module generates comprehensive payroll reports that summarize employee details, including ID, name, and net salary. The report provides a clear and concise overview of payroll information for all employees in the system.

3.4 GUI (Graphical User Interface) Module

The GUI module provides an interactive and user-friendly interface, enabling users to easily interact with the system.

CHAPTER 4
CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

In conclusion, this Employee Payroll Management System effectively demonstrates the integration of various Java concepts to create a functional, user-friendly application for managing employee records and payroll processing. By leveraging object-oriented programming principles, dynamic data handling with collections, robust error management, and an interactive graphical user interface, the system streamlines tasks such as adding employees, calculating net salaries, generating payroll reports, and searching for specific records.

4.2 FUTURE SCOPE

The future scope of this Employee Payroll Management System includes several enhancements to make it more robust, versatile, and scalable. Integration with databases like MySQL or MongoDB can enable persistent data storage, allowing employee records and payroll information to be saved and retrieved even after the application is closed. Adding advanced features such as automated tax calculations, generation of payslips in PDF format, and support for different payroll structures across organizations can improve its applicability.

APPENDIX A
(SOURCE CODE)

```
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;

class Employee {
    int id;
    String name;
    double basicSalary;
    double deductions;
    double bonuses;

    public Employee(int id, String name, double basicSalary, double deductions,
double bonuses) {
        this.id = id;
        this.name = name;
        this.basicSalary = basicSalary;
        this.deductions = deductions;
        this.bonuses = bonuses;
    }

    public double calculateNetSalary() {
        return basicSalary + bonuses - deductions;
    }
```

```
    @Override
    public String toString() {
        return "ID: " + id + ", Name: " + name + ", Net Salary: $" +
calculateNetSalary();
    }
}

class PayrollProcessor {
    ArrayList<Employee> employeeList = new ArrayList<>();

    public void addEmployee(Employee employee) {
        employeeList.add(employee);
    }

    public String generatePayroll() {
        StringBuilder payrollReport = new StringBuilder("--- Payroll Report ---\n");
        for (Employee emp : employeeList) {
            payrollReport.append(emp.toString()).append("\n");
        }
        return payrollReport.toString();
    }

    public Employee findEmployeeById(int id) {
        for (Employee emp : employeeList) {
            if (emp.id == id) {
                return emp;
            }
        }
        return null;
```

```
    }
}

public class EmployeePayrollSystemAWT extends Frame {
    PayrollProcessor payrollProcessor = new PayrollProcessor();

    TextField idField, nameField, basicSalaryField, deductionsField, bonusesField,
searchField;
    TextArea outputArea;

    public EmployeePayrollSystemAWT() {
        setLayout(new FlowLayout());

        Label title = new Label("Employee Payroll Management System");
        title.setFont(new Font("Arial", Font.BOLD, 16));
        add(title);

        add(new Label("Employee ID:"));
        idField = new TextField(10);
        add(idField);

        add(new Label("Employee Name:"));
        nameField = new TextField(20);
        add(nameField);

        add(new Label("Basic Salary:"));
        basicSalaryField = new TextField(10);
        add(basicSalaryField);
```

```
add(new Label("Deductions:"));
deductionsField = new TextField(10);
add(deductionsField);

add(new Label("Bonuses:"));
bonusesField = new TextField(10);
add(bonusesField);

Button addButton = new Button("Add Employee");
Button generateButton = new Button("Generate Payroll");
Button searchButton = new Button("Find Employee");
Button exitButton = new Button("Exit");

add(addButton);
add(generateButton);
add(searchButton);
add(exitButton);

add(new Label("Search by ID:"));
searchField = new TextField(10);
add(searchField);

outputArea = new TextArea(10, 50);
outputArea.setEditable(false);
add(outputArea);

addButton.addActionListener(e -> addEmployee());
generateButton.addActionListener(e -> generatePayroll());
searchButton.addActionListener(e -> findEmployee());
```

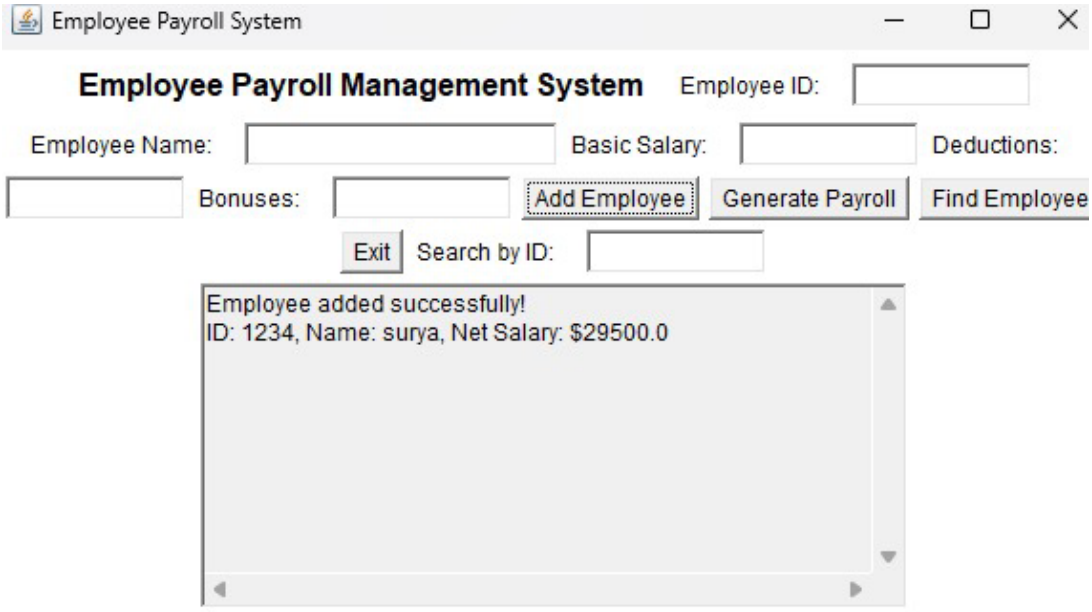
```
        exitButton.addActionListener(e -> System.exit(0));

        setTitle("Employee Payroll System");
        setSize(600, 400);
        setVisible(true);

        // Close Button
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                System.exit(0);
            }
        });
    }

    private void addEmployee() {
        try {
            int id = Integer.parseInt(idField.getText());
            String name = nameField.getText();
            double basicSalary = Double.parseDouble(basicSalaryField.getText());
            double deductions = Double.parseDouble(deductionsField.getText());
            double bonuses = Double.parseDouble(bonusesField.getText());

            Employee newEmployee = new Employee(id, name, basicSalary,
deductions, bonuses);
            payrollProcessor.addEmployee(newEmployee);

            outputArea.setText("Employee added successfully!\n" +
newEmployee.toString());
            clearFields();
```

```
        } catch (NumberFormatException ex) {
            outputArea.setText("Error: Invalid input. Please enter numeric values for ID,
salary, deductions, and bonuses.");
        }
    }

    private void generatePayroll() {
        String payrollReport = payrollProcessor.generatePayroll();
        outputArea.setText(payrollReport.isEmpty() ? "No employees found." :
payrollReport);
    }

    private void findEmployee() {
        try {
            int searchId = Integer.parseInt(searchField.getText());
            Employee foundEmployee = payrollProcessor.findEmployeeById(searchId);
            if (foundEmployee != null) {
                outputArea.setText("Employee Found:\n" + foundEmployee.toString());
            } else {
                outputArea.setText("Employee not found.");
            }
        } catch (NumberFormatException ex) {
            outputArea.setText("Error: Invalid ID. Please enter a numeric value.");
        }
    }

    private void clearFields() {
        idField.setText("");
        nameField.setText("");
```

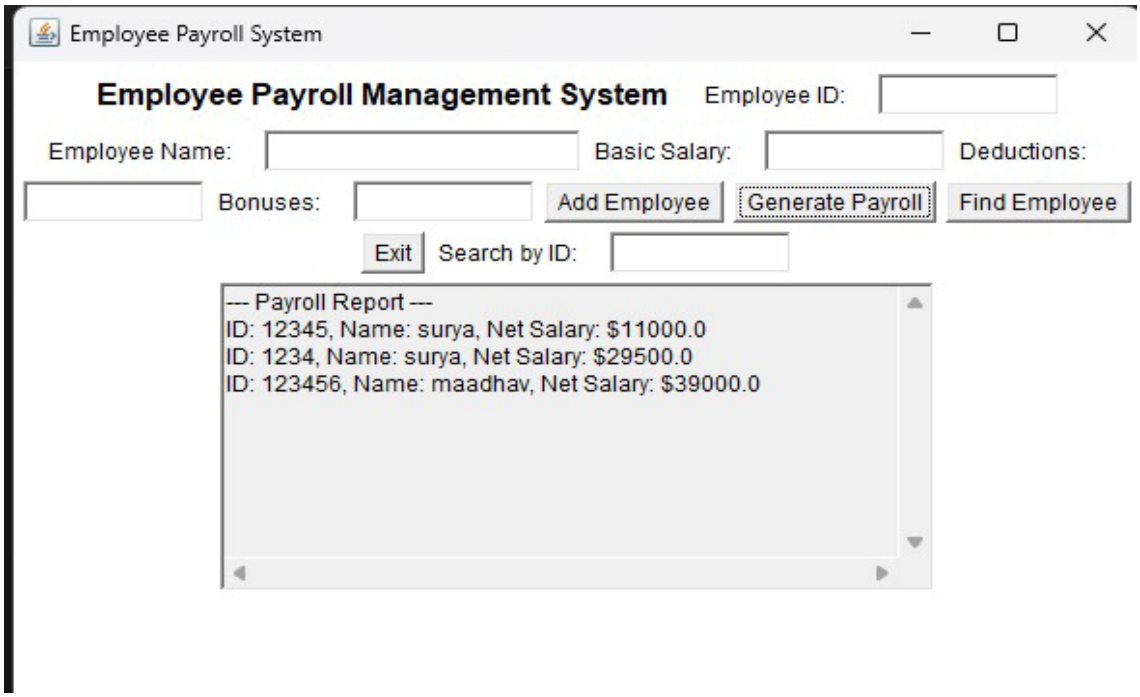
```
        basicSalaryField.setText("");
        deductionsField.setText("");
        bonusesField.setText("");
    }

    public static void main(String[] args) {
        new EmployeePayrollSystemAWT();
    }
}
```

APPENDIX B

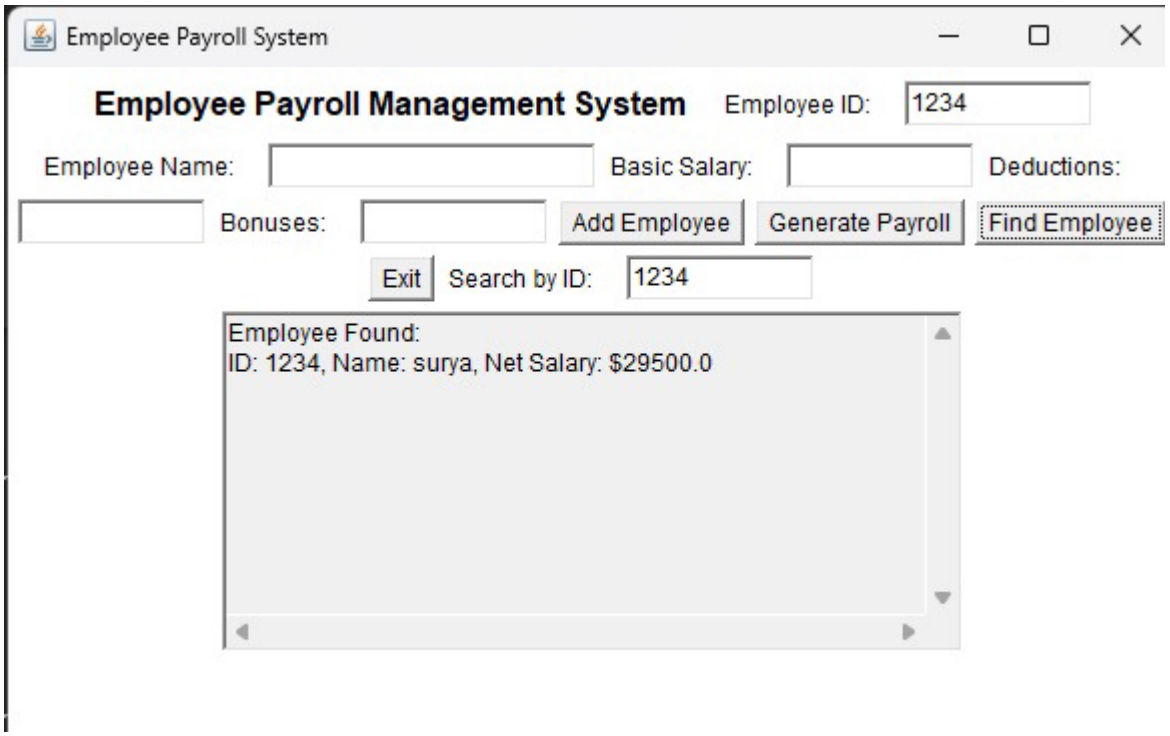
ADD EMPLOYEE



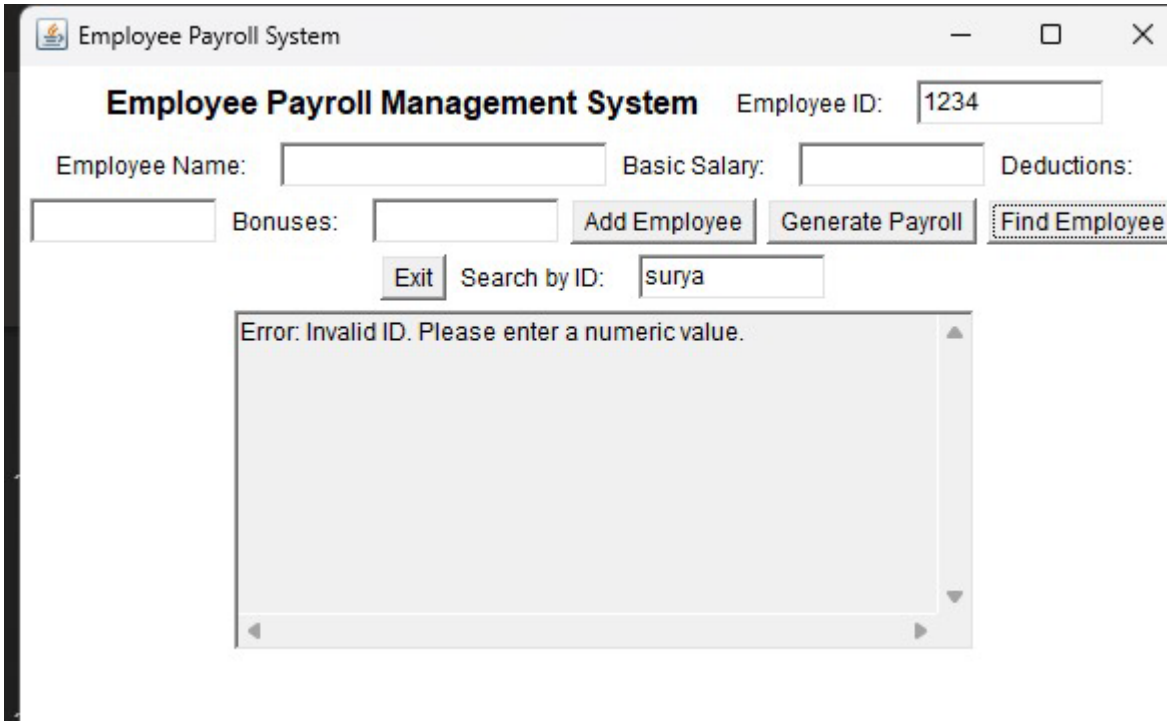
GENERATE PAYROLL



FIND EMPLOYEE



INVALID INPUT



REFERENCES

1."Java Programming: A Beginner's Guide" by Herbert Schildt.

2."Data Structures and Algorithms in Java" by Robert Lafore.

3.Oracle Java Documentation: https://docs.oracle.com/javase/

4.Stack Overflow: Java-related discussions on HashMaps, Java classes, and loops.