

RAJALAKSHMI ENGINEERING COLLEGE
RAJALAKSHMI NAGAR, THANDALAM-602 105



CS23333 OOPS Using Java

Laboratory Record Note Book

Name : ANIRUDH R

Year / Branch / Section : 2ND YEAR \ CSE

University Register No. : 2116240701039

College Roll No. : 240701039

Semester : III

Academic Year : 2025 - 2026



**RAJALAKSHMI ENGINEERING
COLLEGE**
An Autonomous Institution

BONAFIDE CERTIFICATE

Name:ANIRUDH R.....

Academic Year: 2025-2026... **Semester:****III**..... **Branch:**CSE.....

Register No.

2116240701039

*Certified that this is the bonafide record of work done by the above student in
the.....Laboratory
during the academic year 2025- 2026*

Signature of Faculty in-charge

Submitted for the Practical Examination held on.....

INDEX

EX.NO	DATE	NAME OF THE EXPERIMENT	GITHUB QR
1		I/O, Data Types, Operators	
2		Control Structures	
3		Arrays	
4		Strings	
5		Classes & Objects	
6		Inheritance	
7		Interface	
8		Exceptions	
9		Collections	
10		Collections	
11		Project	
12		Lambda	

ABSTRACT

The **Salary Management System** is a software application designed to automate and simplify the process of managing employee salary records within an organization. The system ensures accuracy, efficiency, and security in handling payroll data by integrating a **Java-based front end** with a **SQL database back end**.

The front end, developed using **Java**, provides an interactive and user-friendly interface for administrators and HR personnel to manage employee details, salary structures, deductions, bonuses, and attendance-related inputs. The back end, built on **Structured Query Language (SQL)**, securely stores and retrieves employee information, payroll details, and transaction histories while maintaining data integrity and consistency.

The system automates salary calculations based on predefined parameters such as basic pay, allowances, and deductions, thus minimizing human errors. It also generates reports and payslips, ensuring transparency and easy record-keeping. Additionally, the system allows quick updates and retrieval of data, supporting efficient decision-making and reducing manual workload.

Overall, the Salary Management System enhances organizational productivity by offering a reliable, scalable, and secure solution for payroll management, bridging the gap between HR operations and data-driven management through the seamless integration of **Java and SQL technologies**.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	4
1	INTRODUCTION	6
1.1	Introduction	
1.2	Scope of the Work	
1.3	Problem Statement	
1.4	Aim and Objectives of the project	
2	SYSTEM REQUIREMENTS	7
3	MODULE DESCRIPTION	8
4	SYSTEM DESIGN	9
5	IMPLEMENTATION	10
6	SCREENSHOTS	14
7	CONCLUSION AND FUTURE ENHANCEMENT	18
8	REFERENCES	19

CHAPTER 1

INTRODUCTION

1.1 Introduction

The **Salary Management System** is a software application developed to automate and manage all salary-related operations within an organization. In traditional payroll systems, salary calculations, record maintenance, and report generation are often done manually, which leads to errors, data inconsistency, and time inefficiency. To overcome these challenges, the Salary Management System provides a computerized solution that simplifies the entire payroll process.

1.2 Scope of the Work

The scope of this project includes the development of a complete salary management solution for small to medium-sized organizations. The system automates employee data handling, salary computation, and report generation. It also allows for easy modification of salary details, viewing of payment history, and generation of payslips.

1.3 Problem Statement

In many organizations, salary management is still handled manually using spreadsheets or paper-based systems. This manual process is time-consuming, error-prone, and lacks security and consistency. Errors in salary calculations, difficulties in maintaining records, and delays in report generation often lead to employee dissatisfaction and reduced productivity.

Therefore, there is a need for an automated system that can handle all salary-related processes efficiently, accurately, and securely — ensuring smooth and transparent payroll management.

1.4 Aim and Objectives of the Project

The specific objectives of the project are:

- To develop a computerized system for managing employee salary records.
- To automate salary calculations based on employee details, allowances, and deductions.
- To store and retrieve employee data securely using an SQL database.
- To generate payslips and reports quickly and accurately.
- To reduce manual effort, errors, and paperwork in the payroll process.
- To improve efficiency and transparency in salary management operations.

CHAPTER 2

SYSTEM REQUIREMENTS

Hardware Requirements:

- Processor: Intel i3 or higher
- RAM: 4GB or above
- Hard Disk: 500MB free space

Software Requirements:

- OS: Windows 10 or above
- IDE: IntelliJ IDEA / Eclipse
- Frontend: Java Swing
- Backend: MySQL
- Connector: MySQL JDBC Connector (.jar)

CHAPTER 3

MODULE DESCRIPTION

The Salary Management System is divided into several functional modules to ensure smooth operation, easy maintenance, and clear separation of tasks. Each module performs a specific function and interacts with other modules through a shared database. The modular design improves system efficiency, scalability, and reliability.

3.1 INTRODUCTION

The **Salary Management System** is divided into several functional modules to ensure smooth operation, easy maintenance, and clear separation of tasks. Each module performs a specific function and interacts with other modules through a shared database. The modular design improves system efficiency, scalability, and reliability.

3.2 LIST OF MODULES

The main modules of the Salary Management System are:

- 1. Login Module**
- 2. Employee Management Module**
- 3. Salary Calculation Module**
- 4. Deduction and Allowance Module**
- 5. Report Generation Module**
- 6. Database Management Module**

3.3 MODULE DESCRIPTION

1. Login Module

This module provides secure access to the system. Only authorized users such as administrators or HR staff can log in using valid credentials. It ensures data security and prevents unauthorized access.

2. Employee Management Module

This module handles all employee-related information, including adding, updating, viewing, and deleting employee records. It stores personal and job details in the SQL database for easy retrieval.

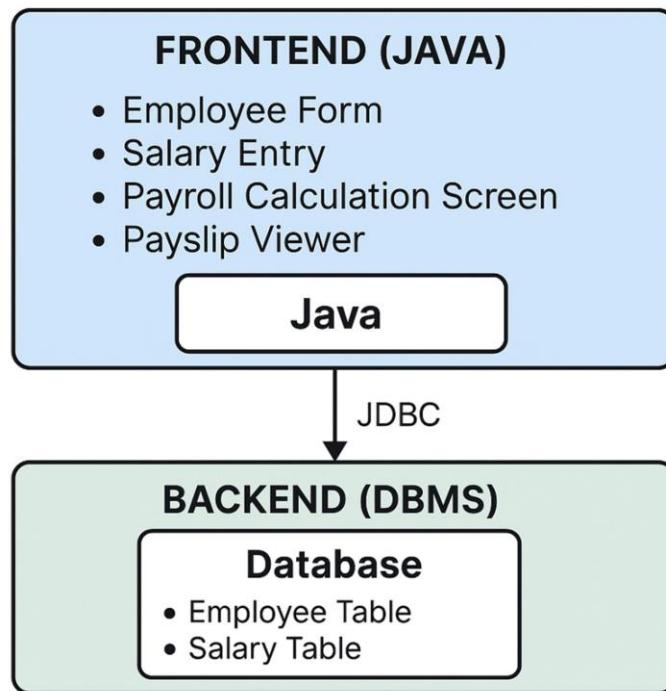
CHAPTER 4

SYSTEM DESIGN

The Salary Management System uses a Java-based frontend to handle user interactions like salary entry and payslip viewing.

It connects to a DBMS backend via JDBC to store and manage employee and salary data.

This design ensures efficient data processing, secure storage, and smooth payroll management.

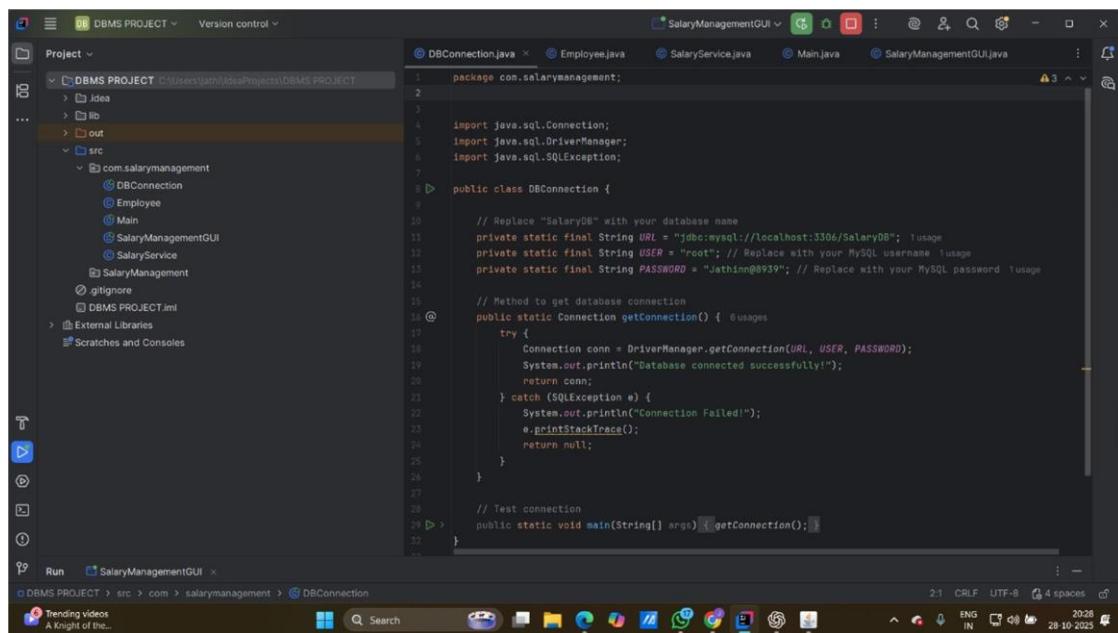


CHAPTER 5:

IMPLEMENTATION

The implementation of this code focuses on connecting the Java application to a MySQL database. It defines a method to establish the connection using JDBC by specifying the database URL, username, and password.

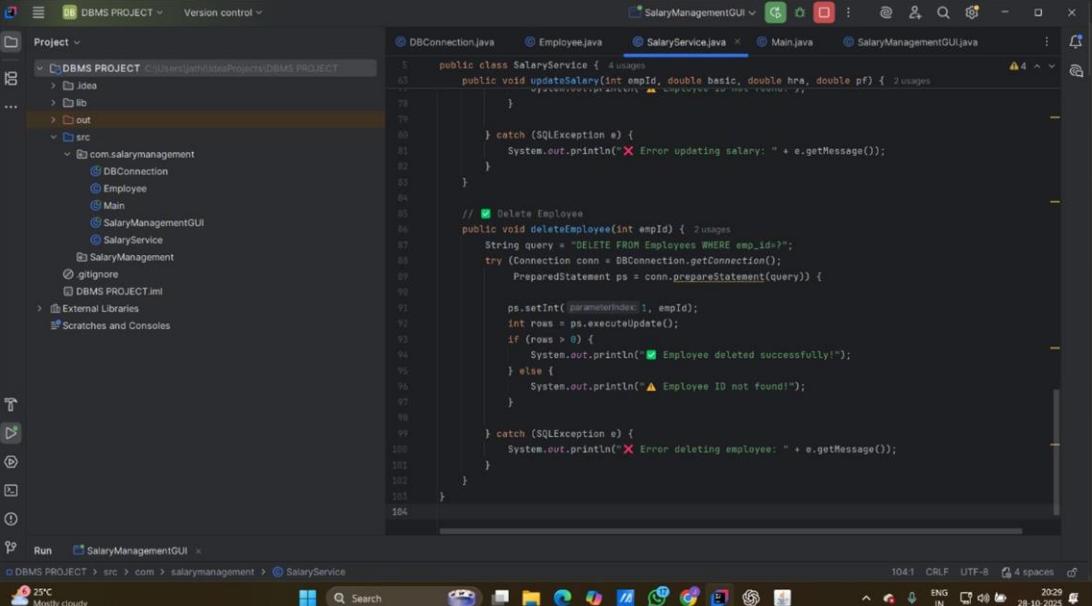
DB CONNECTION CODE:



The screenshot shows the IntelliJ IDEA interface with the project structure on the left and the code editor on the right. The code editor displays the `DBConnection.java` file. The code is as follows:

```
1 package com.salarymanagement;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 public class DBConnection {
8
9
10    // Replace "SalaryDB" with your database name
11    private static final String URL = "jdbc:mysql://localhost:3306/SalaryDB"; Usage
12    private static final String USER = "root"; // Replace with your MySQL username usage
13    private static final String PASSWORD = "Jathinn@939"; // Replace with your MySQL password usage
14
15    // Method to get database connection
16    public static Connection getConnection() { Usages
17        try {
18            Connection conn = DriverManager.getConnection(URL, USER, PASSWORD);
19            System.out.println("Database connected successfully!");
20            return conn;
21        } catch (SQLException e) {
22            System.out.println("Connection Failed!");
23            e.printStackTrace();
24            return null;
25        }
26    }
27
28    // Test connection
29    public static void main(String[] args) {getConnection();}
30 }
```

SALARY SERVICE CODE:

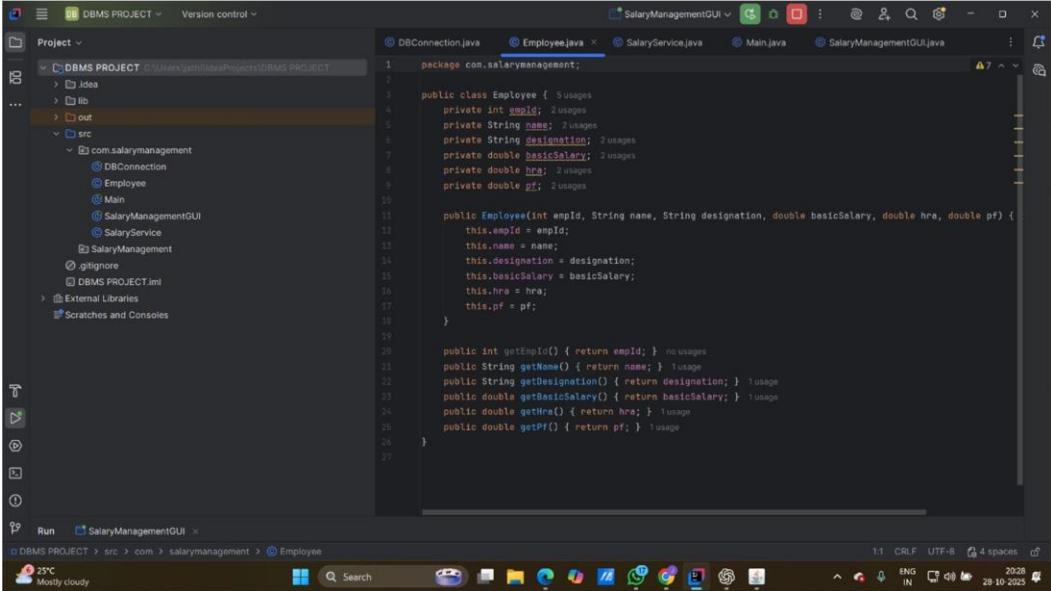


The screenshot shows the IntelliJ IDEA interface with the project 'DBMS PROJECT' open. The 'SalaryService.java' file is the active editor. The code implements a service layer for managing employee salaries. It includes methods for updating an employee's salary and deleting an employee by ID. The code uses JDBC to interact with a database.

```
public class SalaryService {
    public void updateSalary(int empId, double basic, double hra, double pf) {
        try (Connection conn = DBConnection.getConnection();
             PreparedStatement ps = conn.prepareStatement("UPDATE Employees SET basic=?, hra=?, pf=? WHERE emp_id=?")) {
            ps.setDouble(1, basic);
            ps.setDouble(2, hra);
            ps.setDouble(3, pf);
            ps.setInt(4, empId);
            int rows = ps.executeUpdate();
            if (rows > 0) {
                System.out.println("Employee updated successfully!");
            } else {
                System.out.println("Employee ID not found!");
            }
        } catch (SQLException e) {
            System.out.println("Error updating salary: " + e.getMessage());
        }
    }

    // ✅ Delete Employee
    public void deleteEmployee(int empId) {
        String query = "DELETE FROM Employees WHERE emp_id=?";
        try (Connection conn = DBConnection.getConnection();
             PreparedStatement ps = conn.prepareStatement(query)) {
            ps.setInt(1, empId);
            int rows = ps.executeUpdate();
            if (rows > 0) {
                System.out.println("Employee deleted successfully!");
            } else {
                System.out.println("Employee ID not found!");
            }
        } catch (SQLException e) {
            System.out.println("Error deleting employee: " + e.getMessage());
        }
    }
}
```

EMPLOYEE CODE:



The screenshot shows the IntelliJ IDEA interface with the Employee.java file open in the editor. The code defines a class Employee with private fields empId, name, designation, basicSalary, hrs, and pf. It includes a constructor, getters, and setters for these fields.

```
1 package com.salarymanagement;
2
3 public class Employee {
4     private int empId;
5     private String name;
6     private String designation;
7     private double basicSalary;
8     private double hrs;
9     private double pf;
10
11    public Employee(int empId, String name, String designation, double basicSalary, double hrs, double pf) {
12        this.empId = empId;
13        this.name = name;
14        this.designation = designation;
15        this.basicSalary = basicSalary;
16        this.hrs = hrs;
17        this.pf = pf;
18    }
19
20    public int getEmpId() { return empId; }
21    public String getName() { return name; }
22    public String getDesignation() { return designation; }
23    public double getBasicSalary() { return basicSalary; }
24    public double getHrs() { return hrs; }
25    public double getPf() { return pf; }
26}
27
```

CHAPTER 6

SCREENSHOTS

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The left sidebar displays the Navigator with Schemas (salarydb), Tables (employee), Views, Stored Procedures, Functions, and sys. The main area contains a SQL editor window titled 'Query 1' with the following code:

```
1 * CREATE DATABASE SalaryDB;
2 * USE SalaryDB;
3 * CREATE TABLE Employees (
4     emp_id INT PRIMARY KEY AUTO_INCREMENT,
5     name VARCHAR(50) NOT NULL,
6     designation VARCHAR(50),
7     basic_salary DECIMAL(10,2),
8     hra DECIMAL(10,2),
9     pf DECIMAL(10,2)
10 );
11 * INSERT INTO employees (id, name, designation, basic, hra, pf, net_salary) VALUES
12     (1, 'John Doe', 'Manager', 50000.0, 10000.0, 5000.0, 55000.0),
13     (2, 'Jane Smith', 'Developer', 40000.0, 8000.0, 4000.0, 44000.0),
14     (3, 'Hirthic', 'Sports Manager', 150000.0, 20000.0, 10000.0, 150000.0),
15     (4, 'Dathin', 'Lawyer', 500000.0, 100000.0, 30000.0, 560000.0),
16     (5, 'Giri', 'Software Engineer', 500000.0, 150000.0, 65000.0, 585000.0),
17     (6, 'John Doe', 'Manager', 50000.0, 10000.0, 5000.0, 55000.0),
18     (7, 'Jane Smith', 'Developer', 40000.0, 8000.0, 4000.0, 44000.0),
19     (8, 'Alice Johnson', 'Tester', 30000.0, 6000.0, 3000.0, 33000.0);
20
21
```

The right side of the interface has a 'SQL Additions' panel with a note about automatic context help being disabled. The bottom status bar shows the date and time as 28-10-2025.

Salary Management System

ID	Name	Designation	Basic	HRA	PF	Net Salary
1	John Doe	Manager	50000.0	10000.0	5000.0	55000.0
2	Jane Smith	Developer	40000.0	8000.0	4000.0	44000.0
3	Hirthic	Sports Manager	150000.0	20000.0	11000.0	159000.0
4	Jathin	Lawyer	500000.0	100000.0	39000.0	561000.0
5	Giri	Software Enginner	500000.0	150000.0	65000.0	585000.0
6	John Doe	Manager	50000.0	10000.0	5000.0	55000.0
7	Jane Smith	Developer	40000.0	8000.0	4000.0	44000.0
8	Alice Johnson	Tester	30000.0	6000.0	3000.0	33000.0
9	Kathir Vikaas	Doctor	250000.0	85000.0	36000.0	299000.0

ID: Name: Designation:
 Basic: HRA: PF:

[Add Employee](#) [View Employees](#) [Update Salary](#) [Delete Employee](#) [Clear Fields](#)

Salary Management System

ID	Name	Designation	Basic	HRA	PF	Net Salary
1	John Doe	Manager	50000.0	10000.0	5000.0	55000.0
2	Jane Smith	Developer	40000.0	8000.0	4000.0	44000.0
3	Hirthic	Sports Manager	150000.0	20000.0	11000.0	159000.0
4	Jathin	Lawyer	500000.0	100000.0	39000.0	561000.0
5	Giri	Software Enginner	500000.0	150000.0	65000.0	585000.0
6	John Doe	Manager	50000.0	10000.0	5000.0	55000.0
7	Jane Smith	Developer	40000.0	8000.0	4000.0	44000.0
8	Alice Johnson	Tester	30000.0	6000.0	3000.0	33000.0

ID:	9	Name:	Kathir Vikas	Designation:	Doctor
Basic:	250000	HRA:	85000	PF:	36000
Add Employee		View Employees	Update Salary	Delete Employee	Clear Fields

ID:	9	Name:	Kathir Vikaas	Designation:	Doctor
Basic:	250000	HRA:	85000	PF:	36000
Add Employee View Employees Update Salary Delete Employee Clear Fields					

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

Conclusion:

The Salary Management System simplifies and automates the process of managing employee salary details. It ensures accurate calculations, secure data handling, and efficient record management through the integration of Java for the frontend and a DBMS for the backend. The system reduces manual effort, minimizes errors, and improves overall payroll efficiency within an organization.

Future Enhancements:

- Integrate biometric or attendance tracking systems for automated salary computation.
- Add online access for employees to view payslips and salary history.
- Include tax, loan, and bonus management modules.
- Implement data encryption and user role-based access for improved security.
- Develop a mobile or web-based version for remote access and real-time updates.

CHAPTER 8

REFERENCES

1. Java: The Complete Reference – Herbert Schildt
2. MySQL Documentation – <https://dev.mysql.com/doc/>
3. Oracle Java Tutorials – <https://docs.oracle.com/javase/tutorial/>
4. Online Java and JDBC resources.