Employee Payroll System — Project Report

Project Title

Employee Payroll System

Group / Author

Shivam Tech (repository: Shivam-tech-Employee-Payroll-System)

Abstract

This project implements an Employee Payroll System to automate employee management, attendance tracking, salary calculation, and payslip generation. The system reduces manual payroll processing time, improves accuracy, and provides reporting capabilities for management. This report describes the problem, system objectives, analysis, design, implementation, testing, and future enhancements.

Table of Contents

---

1. Introduction

Payroll is a critical function in any organization. Manual payroll processing is time-consuming and error-prone. The Employee Payroll System automates payroll operations, ensuring consistent salary calculations, tax and deduction handling, and providing printable payslips and reports.

2. Problem Statement

Manual payroll management leads to: - Calculation errors (overtime, deductions, taxes) - Delays in payslip distribution - Difficulty producing accurate reports for management and auditing - High administrative overhead

3. Objectives

- Automate employee records management (add, edit, delete employee details)
- Track attendance / working hours
- Compute salaries automatically including basic pay, allowances, deductions, taxes
- Generate and print payslips
- Provide summary and detailed payroll reports
- Maintain secure access and audit logs

4. Scope

- Employee master data (personal, contact, job and salary details)
- Attendance interface (manual entry or import)
- Salary processing module with configurable components
- Payslip generation and export (PDF/print)
- Administrative reports (monthly payroll summary, department wise cost)
- Role-based access (Admin, HR, Accountant)

5. Literature Review (brief)

Payroll systems typically provide modular design separating data storage, business logic (salary calculations), and presentation. Standard approaches incorporate relational databases (MySQL/PostgreSQL), server-side application logic (Java/PHP/.NET/Python/Node), and a web or desktop UI.

6. System Analysis

Functional Requirements

- FR1: Manage employee data (CRUD)
- FR2: Record attendance and leaves
- FR3: Configure salary components (basic pay, HRA, allowances, tax rates)
- FR4: Process payroll for a pay period
- FR5: Generate payslips and payroll reports
- FR6: Backup and restore payroll data

Non-functional Requirements

- NFR1: Reliability and accuracy for financial calculations
- NFR2: Security: role-based access, encrypted storage for sensitive data
- NFR3: Usability: clear UI for administrators and payroll staff
- NFR4: Portability: deployable on common OS/servers

Primary Use Cases

- UC1: Admin logs in and creates an employee record
- UC2: HR records attendance for employees
- UC3: Accountant runs payroll for the month
- UC4: System generates and prints payslips for employees
- UC5: Manager views payroll summary reports

7. System Design

Architecture Overview

(Replace with exact architecture after analyzing repo) - Typical 3-tier architecture: - Presentation Layer: Web UI (HTML/CSS/JS) or Desktop GUI Business Logic Layer: Salary calculation modules, validation - Data Layer: Relational database (MySQL/SQLite/Postgres)

Module Descriptions

- Employee Module: Manage employee profile and salary components
- Attendance Module: Capture attendance and leave
- Payroll Module: Calculate gross/net pay, taxes, deductions, and generate payslips
- Reports Module: Generate monthly, departmental, and yearly reports
- Authentication Module: User login and role management
- Utilities: Backup/restore, data import/export

Database Design (sample)

Core tables (example schemas — adapt from actual DB in zip): - employees - id (PK) - employee_code - first_name - last_name - dob - address - department_id (FK) -

designation - date_of_joining - basic_salary - bank_account tax_id - attendance - id (PK) - employee_id (FK) - date - in_time - out_time hours_worked - status (present/absent/leave) - payroll - id (PK) - employee_id (FK) - pay_period_start - pay_period_end - gross_pay - total_deductions net_pay - pay_date - payroll_components (optional) - id - name - type (earning/deduction) - formula_or_value

(Include ER diagram in final report when extracting from repo or DB schema)

UI Design

- Dashboard: Payroll summary
- Employee list and profile pages
- Attendance entry/import screen
- Payroll processing wizard
- Payslip viewer and PDF export

8. Implementation

Technologies Used (placeholders — confirm from zip)

- Backend: (e.g., Java / PHP / Python / .NET / Node.js)
- Frontend: HTML/CSS/JavaScript or desktop GUI framework
- Database: MySQL / SQLite / PostgreSQL
- Reporting: PDF generation library (e.g., iText/FPDF/ReportLab)

(I will replace the technologies and list the actual package names and versions after inspecting the zip contents.)

Key Classes / Files (example placeholders)

- src/controllers/EmployeeController (handles CRUD)
- src/services/PayrollService (salary computation)
- src/models/Employee, Attendance, Payroll
- templates/payslip_template.html or GUI forms

Data Flow

- HR enters attendance -> Attendance records saved
- Accountant triggers payroll -> PayrollService aggregates attendance, applies salary components and deductions -> Persist payroll record -> Generate payslip PDFs

9. Testing

Test Strategy

- Unit tests for salary computation logic (basic pay, allowances, tax)

- Integration tests for end-to-end payroll processing
- Manual testing of UI flows
- Validation testing for data inputs

Sample Test Cases

- TC1: Create employee with basic salary 10000, verify gross and net salary calculation
- TC2: Mark 2 days leave and verify deduction applied in payroll
- TC3: Generate payslip and verify PDF content matches calculated amounts

(Provide test results table after running actual tests from the codebase.)

10. Deployment

- Server requirements: 2 CPU, 2GB RAM (example)
- Steps:
    1. Install runtime (e.g., Java/PHP/Python)
    2. Create database and run provided SQL scripts
    3. Configure app to database connection
    4. Start application service
    5. Access at provided URL or launch desktop binary

(Exact commands and scripts will be added after reviewing repository.)

11. Security and Data Privacy

- Use role-based access control
- Secure password storage (hashed + salted)
- Protect database backups and restrict access to payroll data
- Audit logging for salary changes and payroll runs

12. Conclusion

The Employee Payroll System automates core payroll tasks, reduces manual effort, and improves accuracy. With configurable salary components and robust reporting, it aids HR and accounting teams in timely payroll processing.

13. Future Enhancements

- Integrate with biometric attendance systems or third-party attendance providers
- Support multiple pay frequencies and multi-currency payroll
- Advanced tax and compliance modules for multiple jurisdictions
- Employee self-service portal for payslips and leave requests
- Automated bank transfer/NEFT integratio