

HR Management & Payroll System

System Design, Implementation Approach, and Challenges Report

Group 67: Ben Morrison - 101572409

1. Overview

The HR Management & Payroll System is a Java-based application designed to handle essential HR operations such as employee record management, department tracking, and payroll processing. Built with JavaFX for the user interface and Java Serialization for data persistence, the system is structured to support modular development, data encapsulation, and scalability for future enhancements like authentication and reporting.

2. System Architecture

2.1 Architectural Pattern

The application follows a modular layered architecture comprising three key layers:

Layer	Responsibility	Key Components
Presentation Layer	Handles GUI interactions and user inputs.	Main.java (JavaFX)
Business Logic Layer	Implements core functionality, payroll calculations, and data management.	DataManager.java, PayrollCalculator.java
Data Model Layer	Represents domain objects and manages their relationships.	Employee.java, Department.java, Payroll.java

Each layer interacts through clearly defined interfaces, minimizing coupling and promoting maintainability.

2.2 Project Directory Structure

```
group-project/
```

```
├── src/
```

```
|   ├── Main.java # JavaFX main application entry point
```

```
| |— model/
| | |— Employee.java # Employee entity
| | |— Department.java # Department entity
| | |— Payroll.java # Payroll entity
| |— service/
| |— DataManager.java # Handles persistence and CRUD operations
| |— PayrollCalculator.java # Handles all payroll computations
|— data/ # Runtime-generated serialized files
| |— employees.dat
| |— departments.dat
| |— payrolls.dat
|— instructions.pdf # Original project brief
```

This structure ensures clear separation of concerns and supports iterative feature expansion.

3. System Components

3.1 Data Model Layer

Employee.java

Represents individual employee records.

Attributes include ID, name, department, employment type, and status.

Implements Serializable for data persistence.

Supports multiple employment types (Full-Time, Part-Time, Contract, Intern).

Department.java

Tracks departmental data and employee associations.

Allows for manager assignment and dynamic employee listing.

Payroll.java

Encapsulates payroll information for each pay period.

Handles gross pay, deductions, and net pay computation automatically.

3.2 Business Logic Layer

DataManager.java

Implements a Singleton pattern to ensure centralized data management.

Performs CRUD operations for employees, departments, and payroll records.

Manages serialization/deserialization for persistent data storage.

PayrollCalculator.java

Contains the payroll computation engine.

Supports both hourly and salaried employees.

Calculates:

Regular and overtime pay.

Federal (15%) and state (5%) taxes.

Social Security (6.2%) and Medicare (1.45%) deductions.

Generates year-to-date summaries for financial tracking.

3.3 Presentation Layer

Main.java

Launches the JavaFX application and initializes the main user interface.

Contains a menu-driven layout with navigation for Employees, Departments, Payroll, and Reports.

Placeholder views for yet-to-be-implemented features ensure forward compatibility.

4. Implementation Approach

Phase 1 – Data Model Setup

Define and test serializable model classes (Employee, Department, Payroll).

Validate object relationships (e.g., Employee-Department linkage).

Phase 2 – Data Persistence

Implement DataManager for centralized data handling.

Introduce automatic data saving on exit and reloading on startup.

Phase 3 – Payroll Calculation Engine

Develop modular PayrollCalculator to ensure isolated and testable computation logic.

Abstract tax and deduction rates for future configuration.

Phase 4 – GUI Development (Ongoing)

Build modular scenes in JavaFX for each core function.

Implement menu-driven navigation and placeholders for upcoming UIs.

Phase 5 – Reporting and Advanced Features (Planned)

Generate CSV/text-based reports.

Integrate role-based authentication and attendance modules.

5. Data Flow

User Input (via JavaFX GUI) →

DataManager (handles CRUD and persistence) →

Model Objects (Employee, Department, Payroll) →

PayrollCalculator (applies business rules and computations) →

Updated Data Serialized (saved in /data/ directory).

This flow ensures data integrity and transparency between operations.

6. Identified Challenges and Potential Risks

Area	Challenge	Potential Impact	Mitigation Strategy
Serialization	Future schema changes may break backward compatibility.	Loss or corruption of saved data.	Implement version control or migrate to a lightweight DB (e.g., SQLite).
GUI Development	Complex TableView and dialog management for CRUD operations.	UI inconsistency and event handling issues.	Modularize UI controllers; use FXML for separation.
Payroll Accuracy	Fixed tax rates are hard-coded.	Limits flexibility for real-world use.	Move tax rates to configuration file or database.
Concurrency	Simultaneous data access (if multi-threaded UI).	Possible data race conditions.	Synchronize access in DataManager.
Scalability	Java serialization limits concurrent usage and large dataset handling.	System slowdown with large employee datasets.	Future-proof by planning migration to SQL/ORM (e.g., Hibernate).

Reporting Generating formatted reports can be complex. Incomplete feature delivery. Start with CSV/text export; add chart visualization later.

Next Steps

Priority 1 – Employee Management UI

Implement CRUD operations in GUI, with search/filter capabilities and form validation.

Priority 2 – Department Management

Enable employee assignment to departments and department hierarchy management.

Priority 3 – Payroll Processing Interface

Build interface to input hours, overtime, and bonuses.

Priority 4 – Reporting

Add export functionality for payroll and employee reports.

Priority 5 – Advanced Enhancements

Include authentication, role-based permissions, and graphical analytics via JavaFX charts.

5. Conclusion

The HR Management & Payroll System lays a solid foundation for a complete HR operations platform. Its modular architecture, consistent data handling, and clear expansion roadmap make it a maintainable and extensible solution for small to mid-scale HR departments.

Future work should emphasize transitioning from serialized data to a relational database for better scalability, adding authentication for data security, and enhancing user experience with interactive reporting dashboards.