

1. Project Description

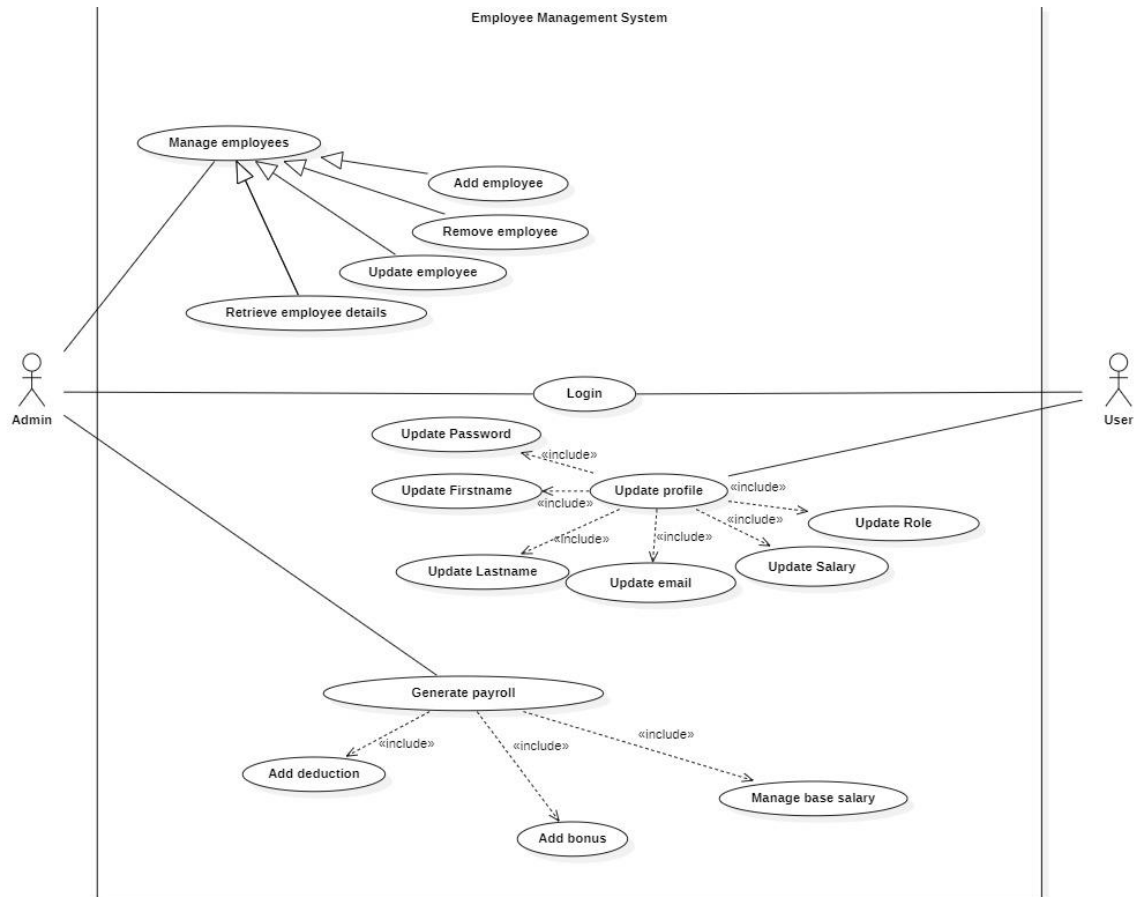
Employee Management System is a software designed to help organisations manage their employee records and track their payrolls. The system provides the ability to add, edit, and delete employee records. It also provides an interactive dashboard to view employee information. The software helps organisations save time and increase efficiency by reducing errors and inaccuracies, and facilitating easy data access. The user-friendly interface and customisable features make this software an ideal solution for organisations of all sizes and industries, helping to support their growth and success. Any changes or updates made to the system are stored in a master log file.

Architecture: Spring MVC

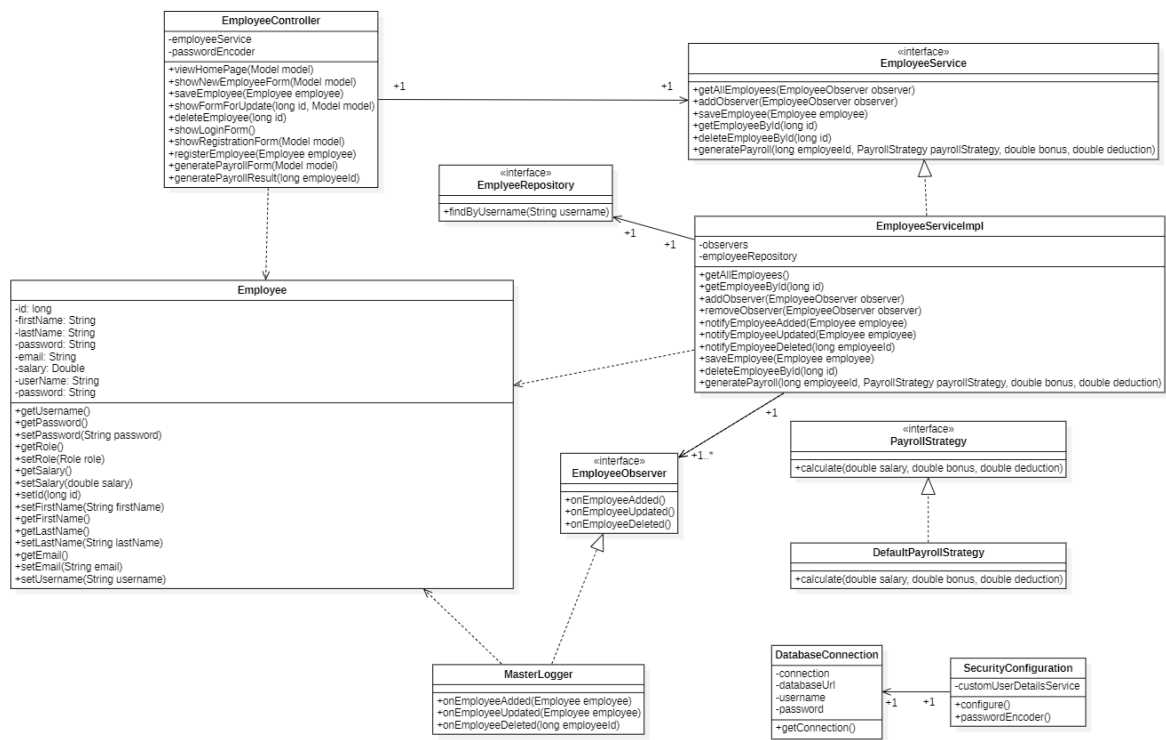
Framework: Spring Boot

2. Analysis and Design Models

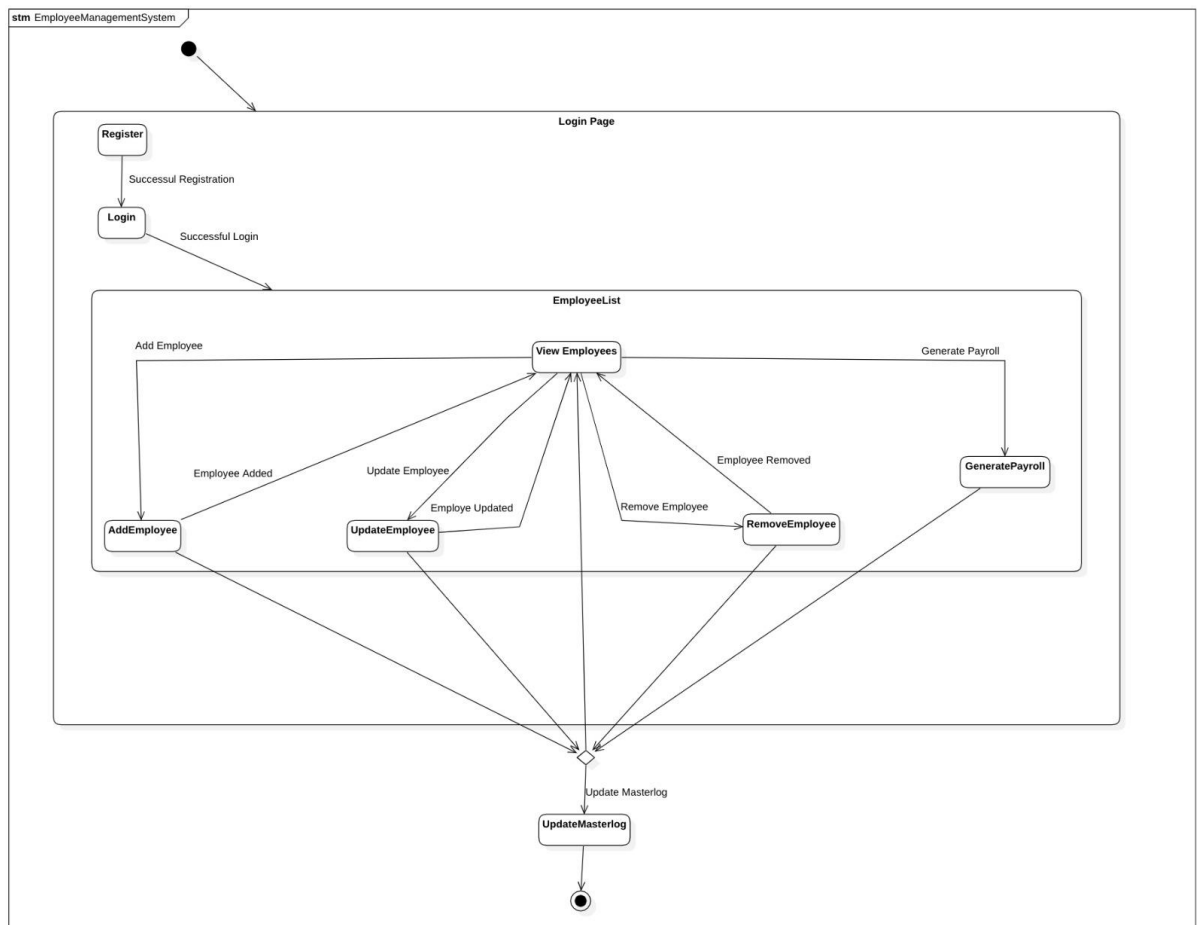
USE CASE DIAGRAM



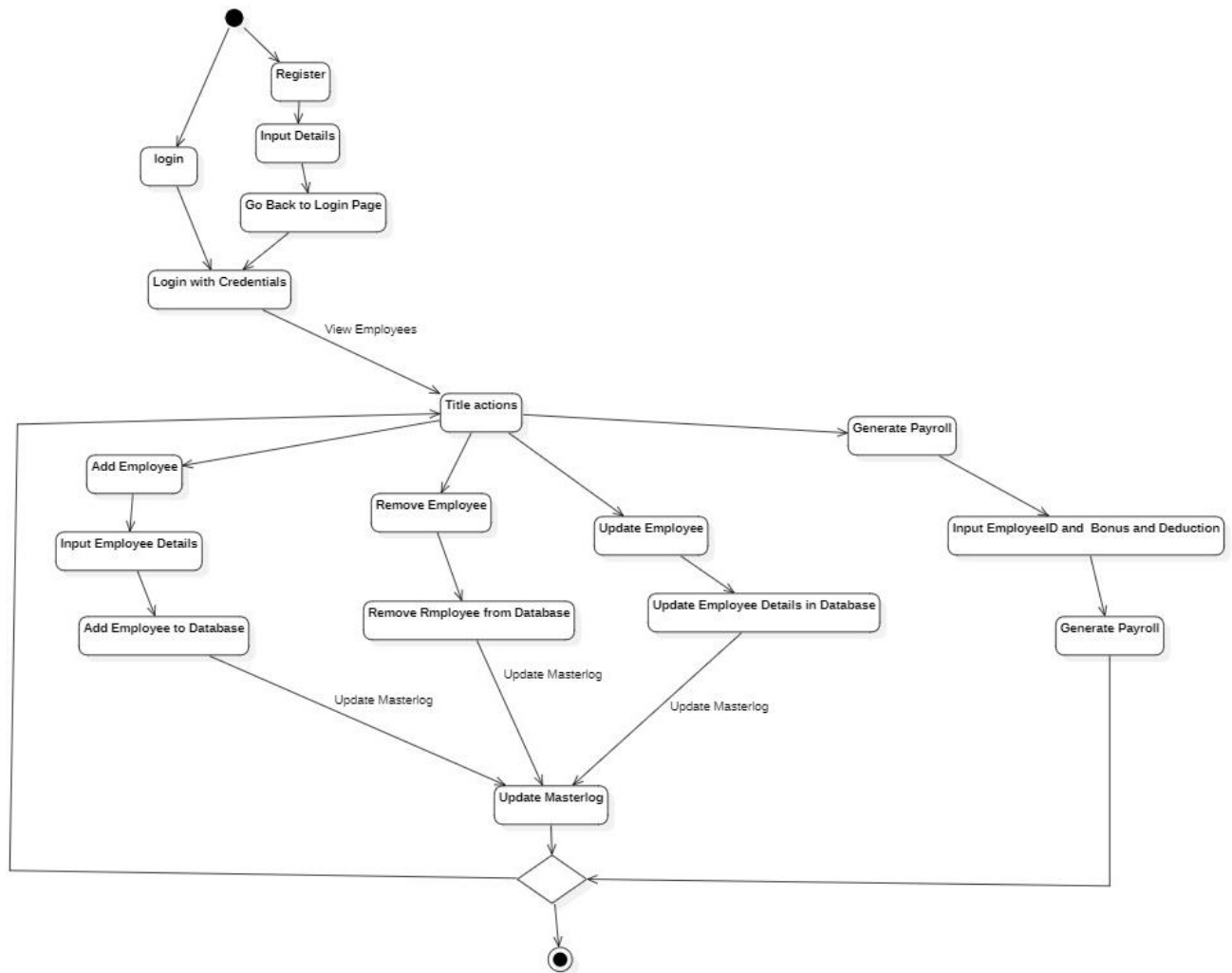
CLASS DIAGRAM



STATE DIAGRAM



ACTIVITY DIAGRAM



3. Architecture Patterns, Design Principles and Design Patterns

Architecture patterns: Spring MVC

Design patterns:

Singleton Design Pattern:

The singleton pattern is used in this code to ensure that only one instance of the DatabaseConnection class is created and shared throughout the application. This implementation of the singleton pattern ensures that the application only creates and uses a single database connection, which can be beneficial in terms of resource management and performance.

Observer Pattern:

The Observer Pattern is a behavioural design pattern that allows an object (the subject) to maintain a list of its dependents (observers) and notify them automatically of any state changes, usually by calling one of their methods.

Whenever there is a change in an employee's state, the subject would call the appropriate method on each of its observers to notify them of the change.

Strategy Pattern:

The Strategy design pattern is used to abstract the payroll calculation logic using an interface named PayrollStrategy. This enables us to use different algorithms (strategies) interchangeably, making it easy to switch between them or extend the system. By using the Strategy pattern in this way,

you can easily add new payroll calculation strategies (e.g., for different tax systems or employee types) without changing the code

Design principles:

Single Responsibility Principle:

The Single Responsibility Principle (SRP) is demonstrated in this code through the Employee class, which represents an employee entity in the application. This class has a single responsibility: to model the employee data and its properties. The class provides getter and setter methods for each property, allowing other parts of the application to interact with the employee data in a controlled manner. These methods do not contain any business logic or validation, keeping the class focused on its single responsibility of modelling the employee data.

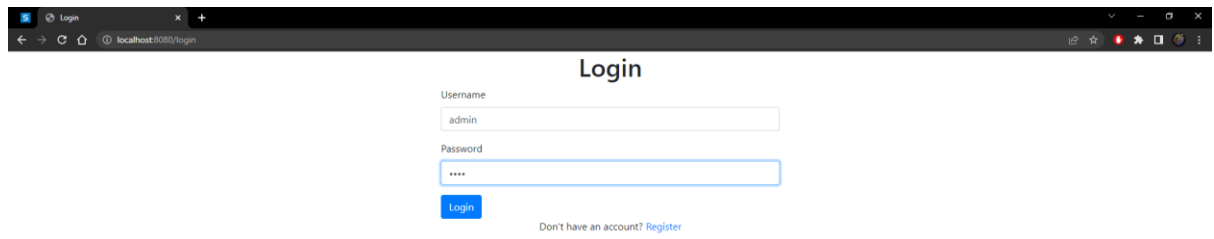
Open-Close Principle:

The Open-Closed Principle (OCP) is one of the SOLID design principles, stating that software entities should be open for extension but closed for modification, promoting more maintainable and flexible code. The Observer Pattern follows the OCP by allowing you to add new types of observers without changing the code of the subject that manages the observers. The subject only needs to maintain a list of observers and notify them when an event occurs. The observers implement a specific interface that defines the methods to be called upon receiving the notifications. The same can be applied to Payroll Strategy.

Liskov Substitution Principle:

It states that objects of a superclass should be able to be replaced with objects of a subclass without affecting the correctness of the program. EmployeeService is an interface that defines the contract for the employee service, which includes methods for adding/removing observers, managing employees, and generating payroll. EmployeeServiceImpl is a concrete class that implements the EmployeeService interface, providing the actual implementation of the methods. By designing the system with an interface (EmployeeService) and a concrete implementation (EmployeeServiceImpl), LSP is being applied. This design allows for flexibility in the future if additional implementations of the EmployeeService interface are needed.

4. Screenshots with input values populated and output shown



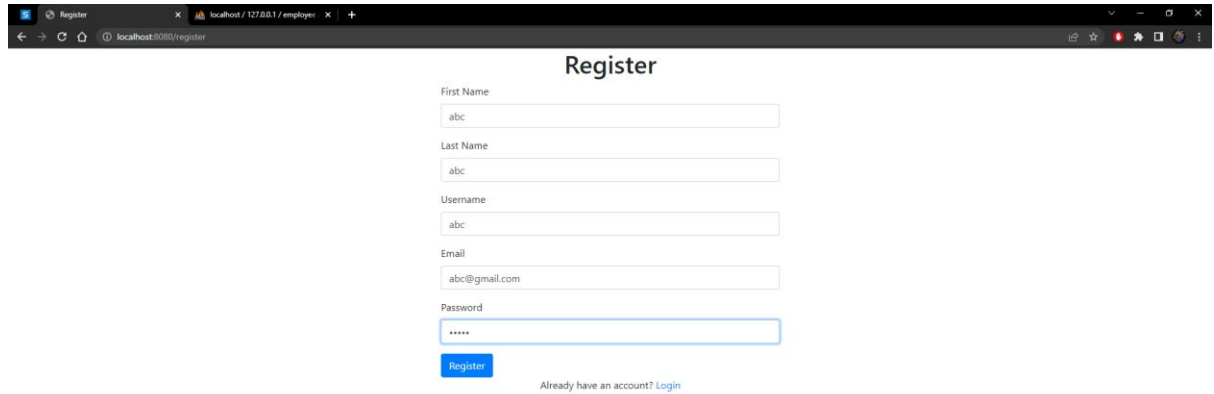
Username

admin

Password

Login

Don't have an account? [Register](#)



First Name

abc

Last Name

abc

Username

abc

Email

abc@gmail.com

Password

Register

Already have an account? [Login](#)



Employee List

[Add Employee](#) [Generate Payroll](#)

Employee First Name	Employee Last Name	Employee Email	Actions
abc	abc	abc@gmail.com	Update Delete
tt	tt	tt@gmail.com	Update Delete
e	e	e@gmail.com	Update Delete
Tushar	Bhat	admin@mail.com	Update Delete
Rethwik	Sai	abcd@gmail.com	Update Delete
Mike	Ross	miker	Update Delete

Employee Management System



Employee Management System

Add Employee

Employee Type: USER

[Save Employee](#)

[Back to Employee List](#)

Generate Payroll

localhost / 127.0.0.1 / employee

localhost:8080/generate-payroll

Generate Payroll

Employee ID

Bonus

Deduction

[Generate Payroll](#)

Payroll Result

localhost / 127.0.0.1 / employee

localhost:8080/generate-payroll

Payroll Result

Payroll Amount: 5000.0

[Generate Another Payroll](#) [Back to Home](#)

Employee Management System

Update Employee

Mike

Ross

miker

miker

1500.0

Employee Type: USER

Update Employee

[Back to Employee List](#)

phpMyAdmin

localhost / 127.0.0.1 / employee

Database: employee - Table: employees

Showing rows 0 - 5 (5 total. Query took 0.0011 seconds)

SELECT * FROM `employees`

☐ Profiling ☐ Edit inline ☐ Explain SQL ☐ Create PHP code ☐ Refresh

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

	id	first_name	last_name	email	password	role	username	salary
<input type="checkbox"/>	9	abc	abc	abc@gmail.com	\$2a\$10\$aK04g09EDsdtrZiHESAJe eA2scVN5uOSOpCRu6cV4...	ROLE_USER	abc	0.1
<input type="checkbox"/>	11	tt	tt	tt@gmail.com	\$2a\$10\$nhj0z5 h1RDKVMTqfRE.Jvsq4 my Fe7BWWJUNBA...	ROLE_USER	tt	0.1
<input type="checkbox"/>	17	e	e	e@gmail.com	\$2a\$10\$9To0XzVbURDjshLUpFedAUsp0wsgVhyQVMTqOg...	ROLE_USER	e	1
<input type="checkbox"/>	26	Tushar	Bhat	admin@mail.com	\$2a\$10\$53eb0OSnuOcrh09hB0uOnvYsqjxKGaJaNhcdieWTWF...	ROLE_USER	admin	1000
<input type="checkbox"/>	30	Rethwik	Sai	abcd@gmail.com	NULL	ROLE_ADMIN	reth	6000
<input type="checkbox"/>	31	Mike	Ross	miker	NULL	ROLE_USER	miker	1500

☐ Check all | With selected | ☐ Edit | ☐ Copy | ☐ Delete | ☐ Export

Query results operations

☐ Print | ☐ Copy to clipboard | ☐ Export | ☐ Display chart | ☐ Create view

☐ Bookmark this SQL query

Label: ☐ Let every user access this bookmark

Console

Press Ctrl+Enter to execute query

> SELECT * FROM `employees`

>