EMPLOYEE MANAGEMENT SYSTEM

Intoduction: The **Employee Management System** is a comprehensive web-based application designed to simplify and automate the management of employee-related tasks within an organization. In today's fast-paced work environment, efficiently managing employee data, attendance, tasks, and performance is crucial for organizational success.This project leverages the power of **Full Stack Java Development**, utilizing **Spring Boot** for the backend to handle business logic and data processing, and **React.js** or **Bootstrap** for the frontend to create an intuitive and responsive user interface. The system is designed to enhance productivity, ensure data security, and provide real-time insights through robust reporting and analytics features.

The primary objective of this project is to provide a centralized platform that seamlessly integrates all aspects of employee management, thereby reducing manual effort, minimizing errors, and improving overall operational efficiency.

Objective: To design and implement a web-based application that efficient manages employee records, including registration, attendance, task assignments, and performance evaluations.

Key Features:

- User Authentication:Secure login and registration system for admins and employees
- Attendence Management:Daily check-in/check-out system with attendance
- Performance Evaluation : Feedback and performance metrics for employees.
- Reports and Analytics: Generate reports on attendance, tasks, and performance

Technology Stack

Backend Tech	Frontend Tech
Spring Boot3	React JS 18+
Spring Data JPA (Hibernate 6)	Vite JS
MySQL Database	Bootstrap CSS
Intellij IDEA	JavaScript
Postman Client	NPM
	VS Code IDE
	Axios

Requirement 1 – Build Frontend React App for Employee Management Module

Add Employee

Get Employee

Get All Employee

Update Employee

Delete Employee

Requirement 1 – Build Frontend React App for Employee Management Module

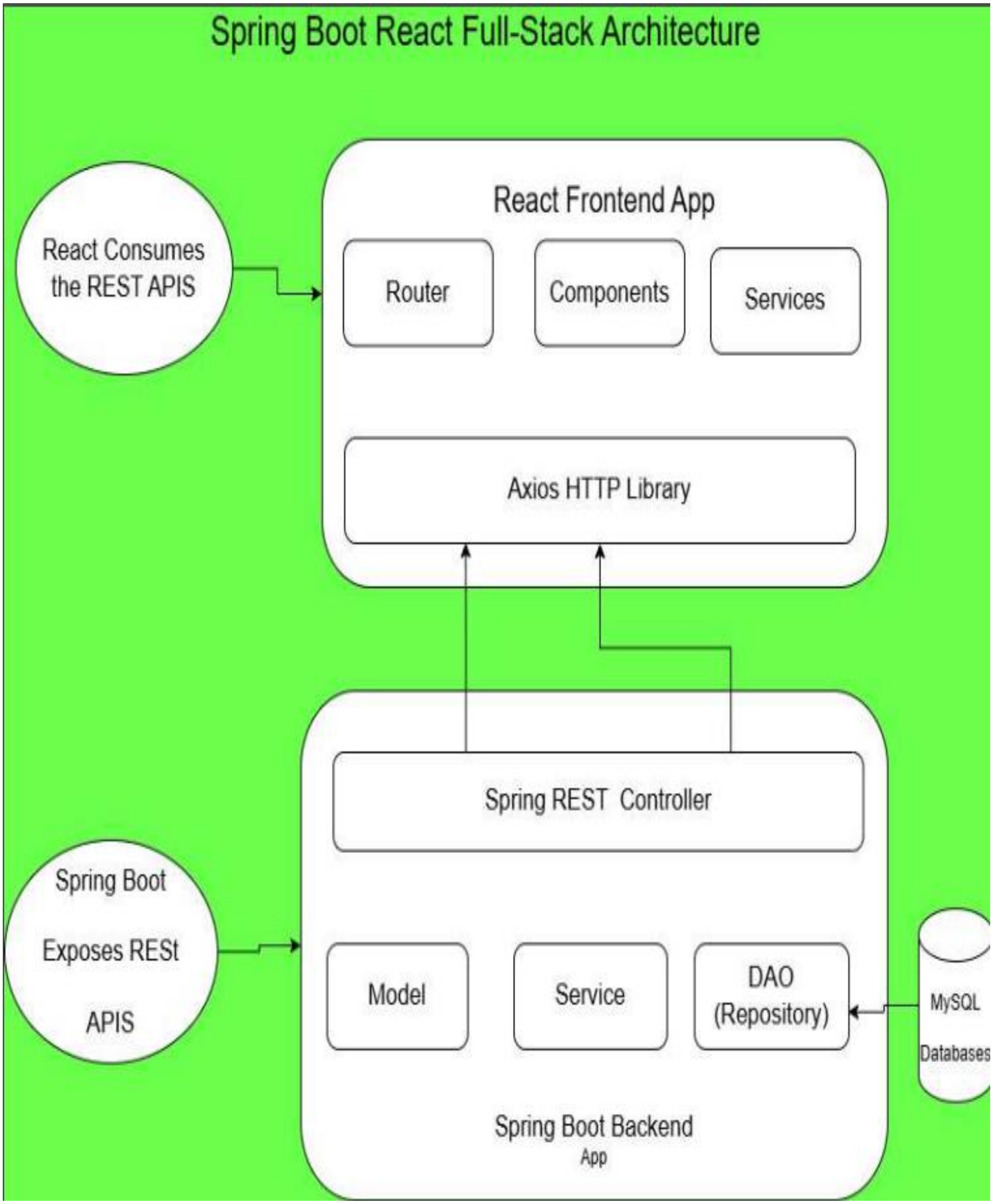
Use should able perform below CRUD operations

Add New Employee

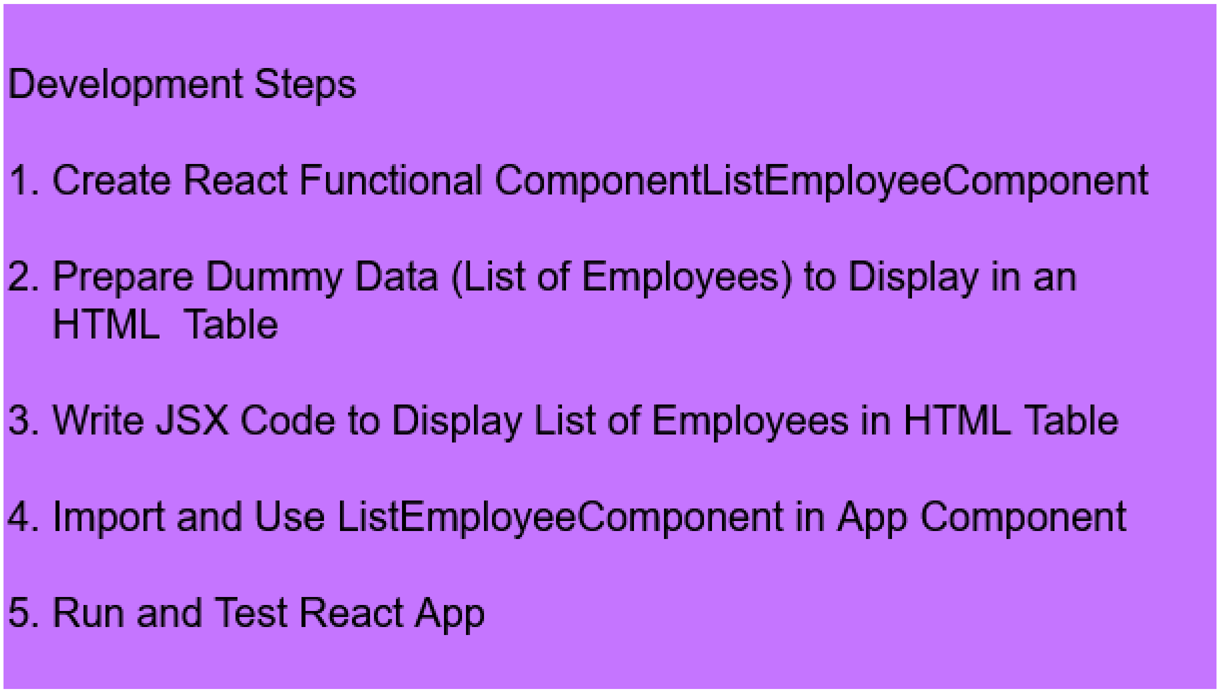
List All Employee

Update Existing Employee

Delete Existing Employee


Spring Boot React Full-Stack Architecture
React Consumes the REST APIS
React Frontend App
Router
Components
Services
Axios HTTP Library
Spring REST Controller
Spring Boot Exposes RESt APIS
Model
Service
DAO (Repository)
MySQL Databases
Spring Boot Backend App

BUILD EMS PROJECT STEP BY STEP



Development Steps

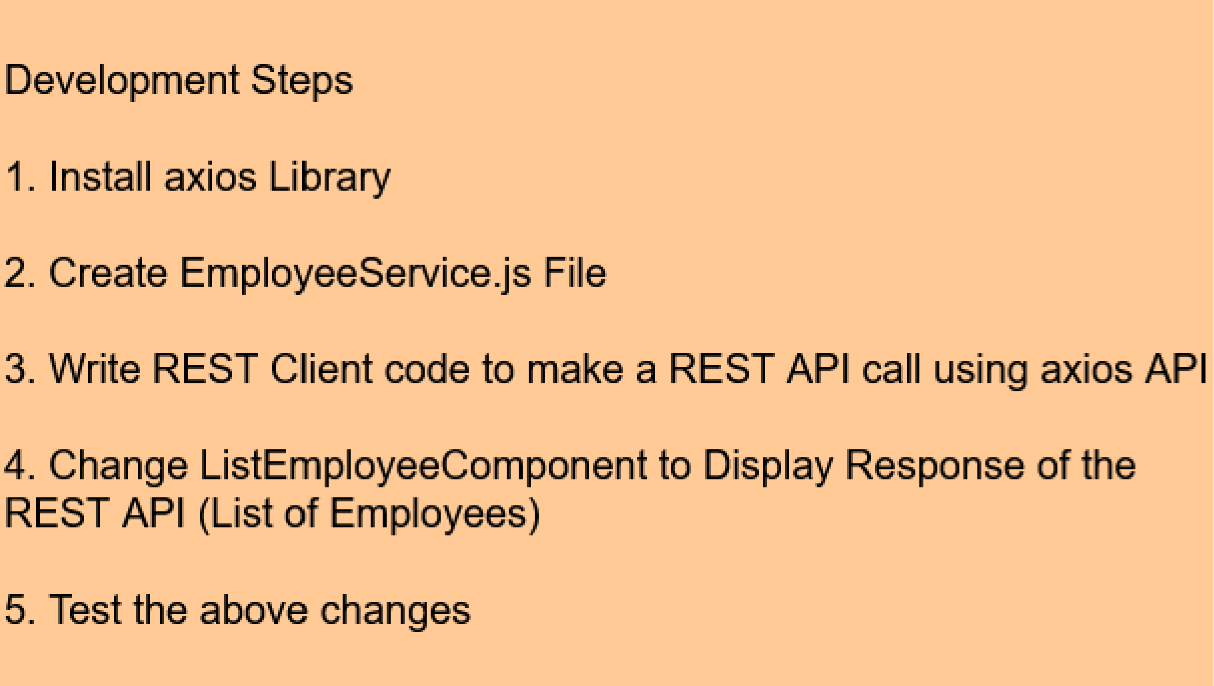
1. Create React Functional ComponentListEmployeeComponent

2. Prepare Dummy Data (List of Employees) to Display in an HTML Table

3. Write JSX Code to Display List of Employees in HTML Table

4. Import and Use ListEmployeeComponent in App Component

5. Run and Test React App



Development Steps

1. Install axios Library

2. Create EmployeeService.js File

3. Write REST Client code to make a REST API call using axios API

4. Change ListEmployeeComponent to Display Response of the REST API (List of Employees)

5. Test the above changes

Development Steps

1. Create HeaderComponent (functional component)

2. Import and Use HeaderComponent in App Component

3. Create FooterComponent (functional component)

4. Import and Use FooterComponent in App Component

Development Steps

1. Install react-router-dom library using NPM

2. Configure Routing in App Component

3. Configure Route for ListEmployeeComponent

4. Test Route for ListEmployeeComponent

Development Steps

1. Create React Functional Component - EmployeeComponent

2. Add "App Employee" buttom in ListEmployeeComponent

3. Configure Route for EmployeeComponent

4. Test the above changes

Development Steps

1. Define state variables (firstName, lastName and email) in EmployeeComponent using useState Hook.

2. Design Add Employee Form using HTML and Bootstrap

3. Create JavaScript Function to handle onClick Event (Form submit)

4. Test React App (print form data in browser console)

Development Steps

1. In EmployeeService, write a code to call Add Employee REST API using axios

2. Change EmployeeComponent to call EmployeeService method

3. Navigate to List Employees Page After Form Submission Done

4. Test above changes

Development Steps

1. Use the useState hook to intialize state variable that will hold validation errors

2. Write a validation function that checks the form data and returns validation errors

3. Validate Form on Submission

4. Display Validation Errors

5. Test above changes

Development Steps

1. Add Update button to list employees page

2. Add Route for Update Employee in App component

3. Change Page Title Dynamically (EmployeeComponent supports both Add and Update)

5. Test above changes

Development Steps

1. In EmployeeService, write a code to call Get Employee REST API unsing axios

2. Use useEffect hook to populate the employee data in the form for update

3. Test above

Development Steps

1. In EmployeeService, write a code to call Update Employee REST API unsing axios

2. Change EmployeeComponent.saveOrUpdateEmployee() method to perform both add and update employee operation

3. Test above changes

Development Steps

1. In EmployeeService, write a code to call Update Employee REST API unsing axios

2. Change EmployeeComponent.saveOrUpdateEmployee() method to perform both add and update employee operation

3. Test above changes

Roles and Responsibilities:

- Designed and implemented RESTful APIs using **Spring Boot**.
- Developed a responsive UI using **React.js** and **Bootstrap**.
- Managed database schemas and queries using **JPA/Hibernate** with **MySQL**
- Ensured secure authentication and authorization using **Spring Security**.
- Collaborated in an Agile environment with regular code reviews and version control using **Git.**

Final Conclusion:

The **Employee Management System** successfully streamlines and automates the management of employee data, attendance, task assignments, and performance evaluations. By leveraging a robust tech stack of **Spring Boot** for the backend and **React.js** or **Bootstrap** for the frontend, the system delivers a responsive and user-friendly interface.

The project not only enhances operational efficiency but also ensures secure data handling and easy scalability for future enhancements. It demonstrates effective implementation of **Full Stack Java Development** practices, showcasing proficiency in building maintainable, high-performance web applications.

Overall, the project achieves its objective of simplifying HR processes, reducing manual effort, and providing actionable insights through comprehensive reports and analytics, making it a valuable tool for any organization.

- Develop by : Sumant Kumar