**Staff Management System**

[Objective: 2](#_Toc153533858)

[Abstract: 2](#_Toc153533859)

[Users of Staff Management System 3](#_Toc153533860)

[Functionality available to the User 3](#_Toc153533861)

[Modules in the Staff Management System 3](#_Toc153533862)

[Create Employee Module: 3](#_Toc153533863)

[List Employee Module: 3](#_Toc153533864)

[Update Employee Module: 3](#_Toc153533865)

[Search Employee Module: 4](#_Toc153533866)

[View Employee Module: 4](#_Toc153533867)

[Delete Employee Module: 4](#_Toc153533868)

[NFRs & SLAs: 4](#_Toc153533869)

[Testing: 4](#_Toc153533870)

[TDD/BDD Approach: 5](#_Toc153533871)

[Atomic Design in React 5](#_Toc153533872)

[Other Considerations: 6](#_Toc153533873)

[Staff Management System Architecture 7](#_Toc153533874)

[Staff Management System Process Flow 8](#_Toc153533875)

[Prerequisites 8](#_Toc153533876)

[Staff Management system Layers 8](#_Toc153533877)

[Tools and Technologies Used 8](#_Toc153533878)

[Server-side tools and technologies used 8](#_Toc153533879)

[Client-side tools and technologies used 9](#_Toc153533880)

[Creating the Backend Application using Spring Boot 9](#_Toc153533881)

[Project Structure of React Front End 10](#_Toc153533882)

[Screenshots of Staff Management System 11](#_Toc153533883)

[Console Menu: 11](#_Toc153533884)

[ReactJS Application Landing Page 11](#_Toc153533885)

[Create Employee Module: 12](#_Toc153533886)

[CLI Application 12](#_Toc153533887)

[ReactJS Application 12](#_Toc153533888)

[List Employee Module: 12](#_Toc153533889)

[CLI Application 12](#_Toc153533890)

[ReactJS Application 13](#_Toc153533891)

[Update Employee Module: 13](#_Toc153533892)

[CLI Application 13](#_Toc153533893)

[ReactJS Application 13](#_Toc153533894)

[Search Employee Module: 14](#_Toc153533895)

[CLI Application 14](#_Toc153533896)

[ReactJS Application 14](#_Toc153533897)

[View Employee Module: 15](#_Toc153533898)

[CLI Application 15](#_Toc153533899)

[ReactJS Application 15](#_Toc153533900)

[Delete Employee Module: 15](#_Toc153533901)

[CLI Application 15](#_Toc153533902)

[ReactJS Application 16](#_Toc153533903)

[References: 16](#_Toc153533904)

# Objective:

Develop an end-to-end full-stack web application that is a **Staff Management System** using ReactJS, Spring Boot, Spring REST and Spring Data JPA.

Implement CI-CD with Jenkins.

Dockerize the front and back-end applications and push to a container registry.

Pull the images and deploy Spring Boot Back End and ReactJS Front End applications.

# Abstract:

Staff Management System Project is an online software application that provides easy, accurate and fast access to the staffing system.

This Application will initially provide a Menu-Driven Console Interface to a User. The User can perform functions like Create Employee, List Employee, Update Employee, Search for an Employee, View Employee and Delete Employee.

Over the course of five Sprints, various features and functionalities will be introduced like Hibernate ORM replacing JDBC, introduction of REST APIs, transformation to a Spring Boot project, CI/CD with Jenkins, Performance Testing with JMeter till finally it will be deployed as a Full Stack Application with a ReactJS Front End and a Spring Boot Back End. Development will follow the TDD approach. NFRs will be emphasized throughout and the system will be expected to generate Unit and BDD-Cucumber test reports in every Sprint. Importantly, the UML diagrams detailing the architecture of the project needs to be updated in every Sprint.

The following facilities provided by the Online Staffing software make it more easy, accessible and effective.

* The HR manager (User) can view employee details such as first name, last name, email ID etc. from anywhere via internet.
* Online Staff Management System is designed to give facilities to create, update, list, search, view and delete employees.
* The landing page of the web application should display the complete list of employees in the organization.

The features have been kept few and simple because the goal of this Contextual Problem statement is to allow trainees to focus on NFRs/SLAs and additionally gain experience in DevOps tools, Performance and Unit Testing as well as writing customer acceptance tests with Cucumber.

# Users of Staff Management System

There is one user in the Staff Management System:

* HR Manager: With Full Access to the system.

## Functionality available to the User

**These are the features available to a user of the system:**

* Create Employee
* List Employees
* Update Employee
* Search for an Employee
* Delete Employee
* View Employee

# Modules in the Staff Management System

## Create Employee Module:

This module provides all the functionality related to creating an employee in the system. Employee details input by the HR Manager or user of the system will be stored in the appropriate database tables after validation in the front as well as back-end.

## List Employees Module:

The main purpose for developing this module is to display all the registered employees.

## Update Employee Module:

The main purpose of this module is providing all the functionality related to updating an employee’s details in the system. Employee details input by the HR Manager or user of the system will be stored in the appropriate database tables after validation in the front as well as back-end.

## Search Employee Module:

The main purpose of this module is providing all the functionality related to searching for a particular employee’s details in the system.

## View Employee Module:

Employee records will be retrieved from the database and displayed as formatted output in the console or browser depending on the functionalities described in the Sprint.

## Delete Employee Module:

The main objective of this module is to manage employee exits from the organization.

# NFRs & SLAs:

* The system must support 1-page/5-page requests per second
* The system will need to support 150 transactions per second.
* Concurrent user session – 200
* Page view times < 3s
* Responsive web design - supported view port 767px (Mobile and Desktop)
* CI&CD Pipeline(s) to be deployed via Jenkins | Docker.
* Different employees should not be able to access each other’s data.
* Secure coding standards to be followed.
* User friendly error pages should be displayed to the end users
* Validation of user input to be carried out in both the front as well as the back-end.
* Automate Performance Testing with JMeter
* **Only one session per user** at a time or **no concurrent session per user**. If the user tries to open a new session, then either an alert is shown or his previous session is closed.

# Testing:

* Smoke and Sanity Testing
* Unit Testing and Regression
* System Testing
* [JaCoCo](https://www.eclemma.org/jacoco/) - Code coverage > 80%
* Performance Test with JMeter
* Unit Test Reports with JUnit and Spring Boot Testing Framework
* Error and Exception handling
* Test Data preparation, management and data cleanup should be in place

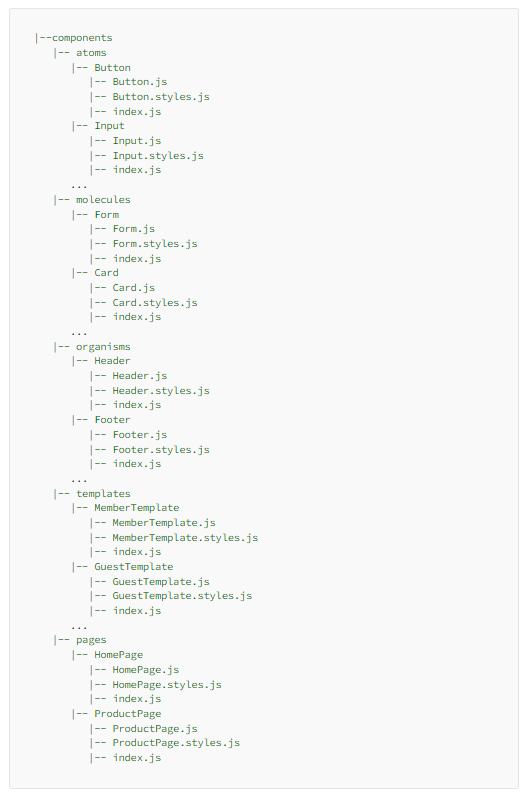
# TDD/BDD Approach:

* TDD with Junit to be followed for Java application development.
* TDD with Jest to be followed for ReactJS application development.
* Behavior Driven Development (BDD) to utilize the Cucumber framework.

# Atomic Design in React

Participants should have a working knowledge of the five distinct levels of atomic design — atoms > molecules > organisms > templates > pages — and how it maps to a React application’s component-based architecture.

This is an example of a project hierarchy that uses the principles of Atomic Design while separating the logic and styles of the components:



Below Hands-On Assignments are mandatory:

Week-5 Day-3

|  |
| --- |
| Lab 25 - Build a React Application with Material UI following Atomic Design — Part 1 |
| Lab 25 - Build a React Application with Material UI following Atomic Design — Part 2 |
|  |

# Other Considerations:

* System should support upload of raw data in CSV format.
* At least 50,000 employee records need to be stored
* The below link allows you to download sample csv files ranging from 100 records to 5000000 records: -

<https://eforexcel.com/wp/downloads-16-sample-csv-files-data-sets-for-testing/>

* Other links that may be helpful for obtaining raw data: -

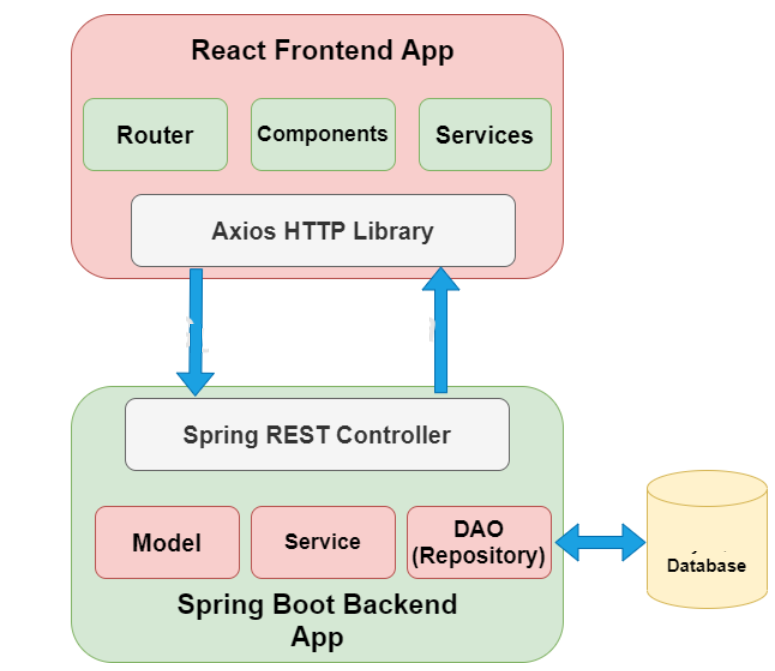
<https://gist.github.com/kevin336/acbb2271e66c10a5b73aacf82ca82784>

<https://sample-videos.com/download-sample-csv.php>

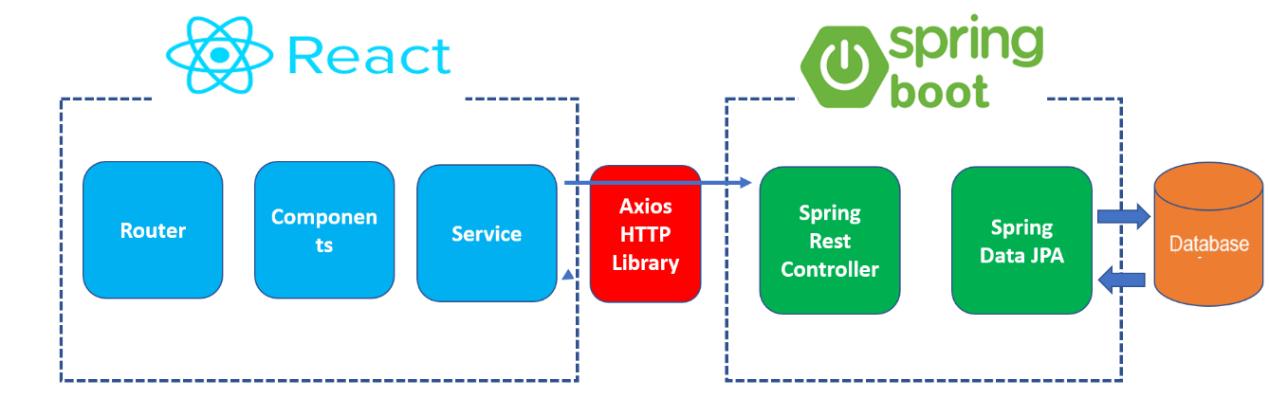
<https://wsform.com/knowledgebase/sample-csv-files/>

* If no one csv suits your requirement, you can compose your datasets from multiple CSVs.
* You may also import only selected columns from a CSV into tables in the database.

# Staff Management System Architecture



# Staff Management System Process Flow



# Prerequisites

* Basic familiarity with HTML & CSS
* Basic knowledge of JavaScript and programming
* Spring Boot Basics
* ReactJS basics
* Node.js and npm installed globally

# Staff Management system Layers

We will build two projects and an EIS tier:

1. **sprint boot-backend (server)** – To Produce the REST APIs
2. **react-frontend (client)** – To Consume the REST APIs
3. **database layer** - PostgreSQL

# Tools and Technologies Used

## Server-side tools and technologies used

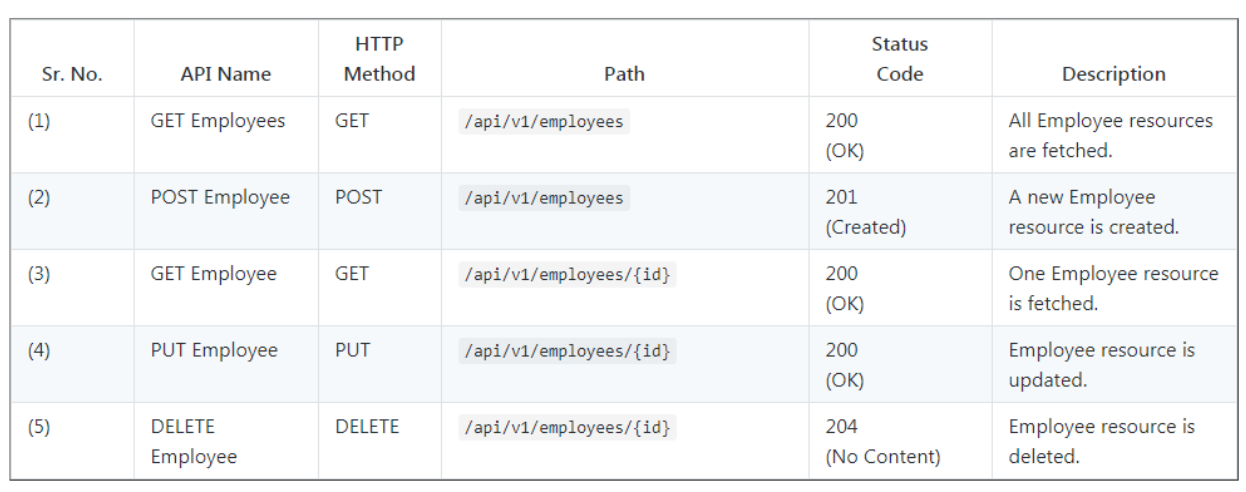
* Spring Boot 2 +
* Spring Data JPA (Hibernate)
* Maven 3.2 +
* JDK 11+
* Embedded Tomcat 8.5+
* PostgreSQL Database

## Client-side tools and technologies used

* React
* Modern JavaScript (ES6)
* NodeJS and NPM
* VS Code IDE
* Create React App CLI
* Bootstrap 5 and Axios HTTP Library

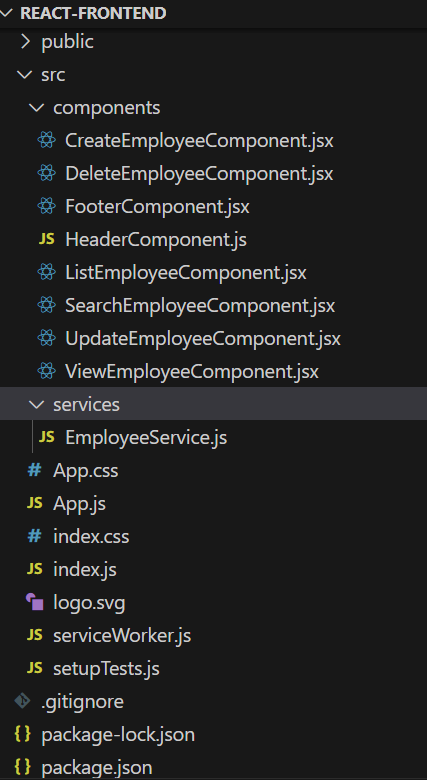
# Creating the Backend Application using Spring Boot

Following are five REST APIs (Controller handler methods), that will be developed for Employee resource.



# Project Structure of React Front End

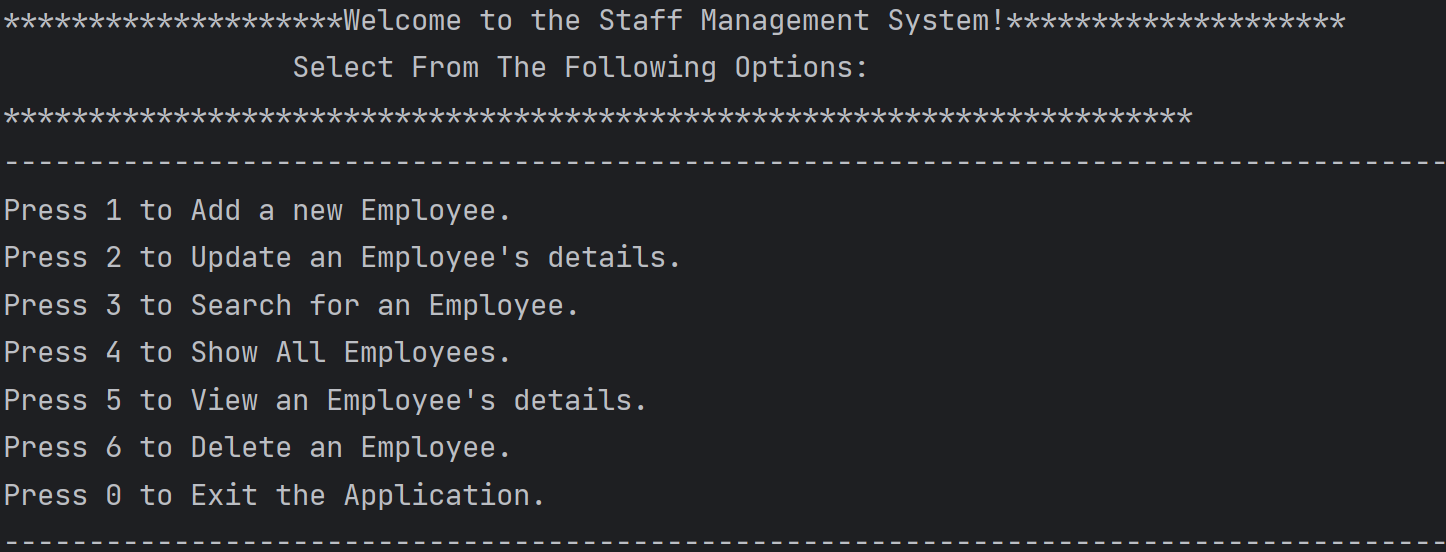
Below is a suggested Project Structure for the Front-end application:



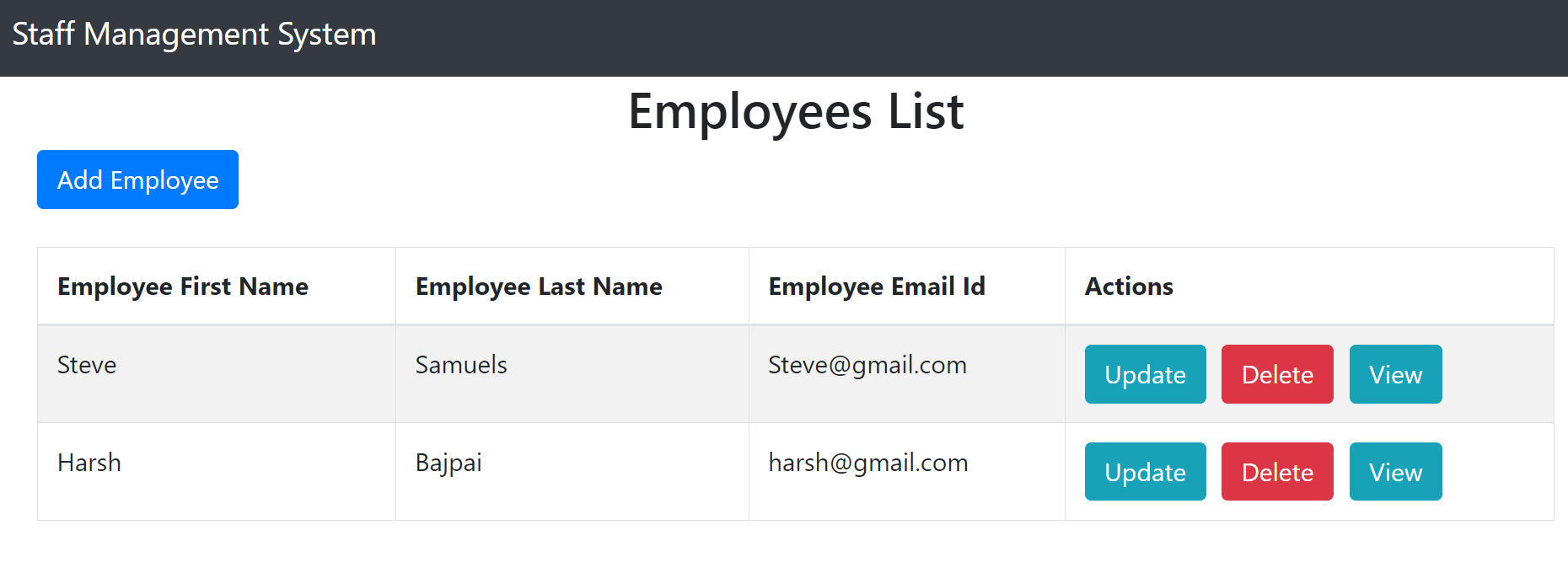
# Screenshots of Staff Management System

\*\* Participants are free to use their own layout and styling but essential details must be displayed.

## Console Menu:

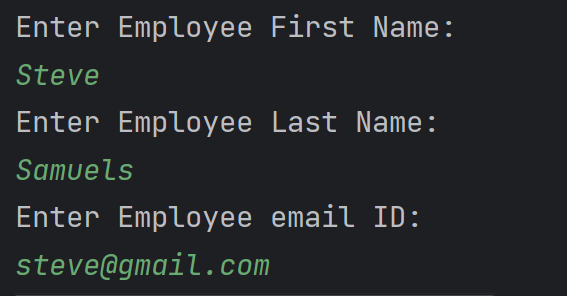


## ReactJS Application Landing Page

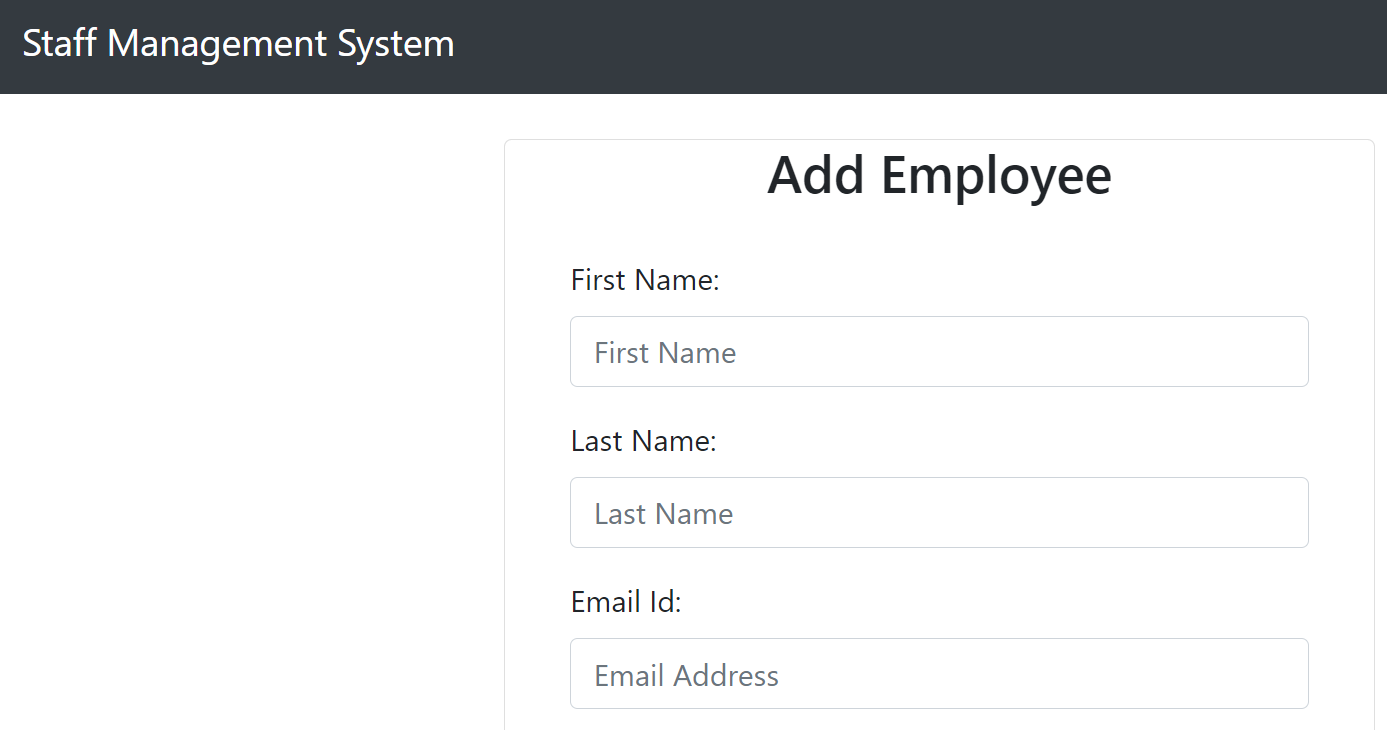


## Create Employee Module:

### CLI Application

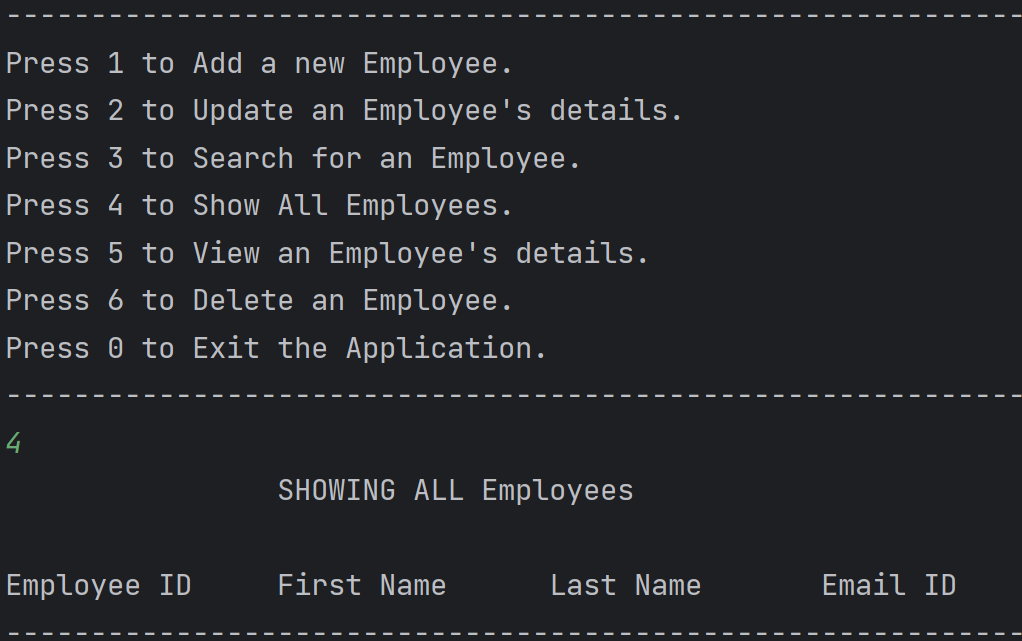
****

### ReactJS Application

****

## List Employee Module:

### CLI Application

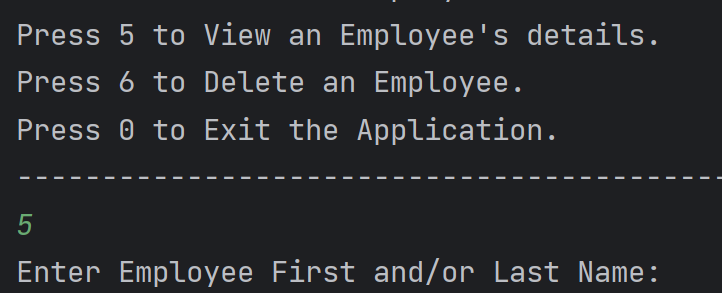
****

### ReactJS Application

Displayed on the landing page.

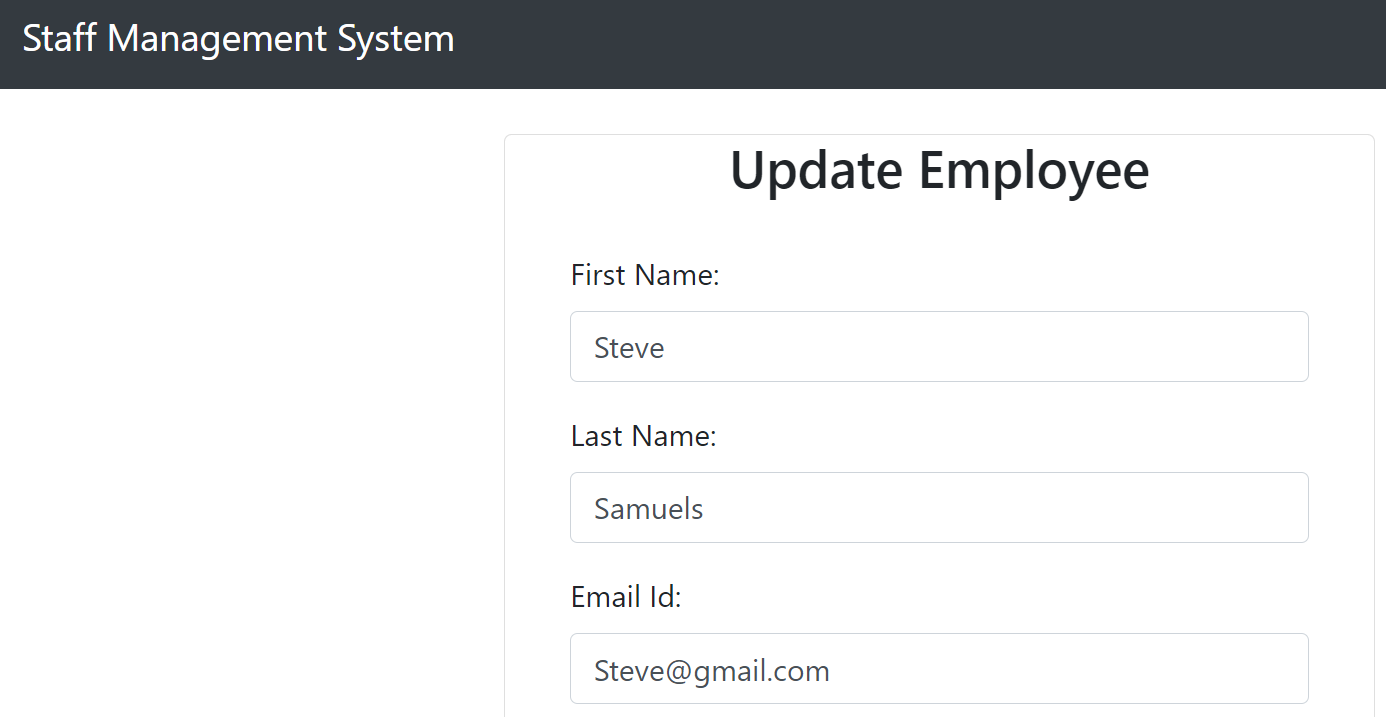
## Update Employee Module:

### CLI Application



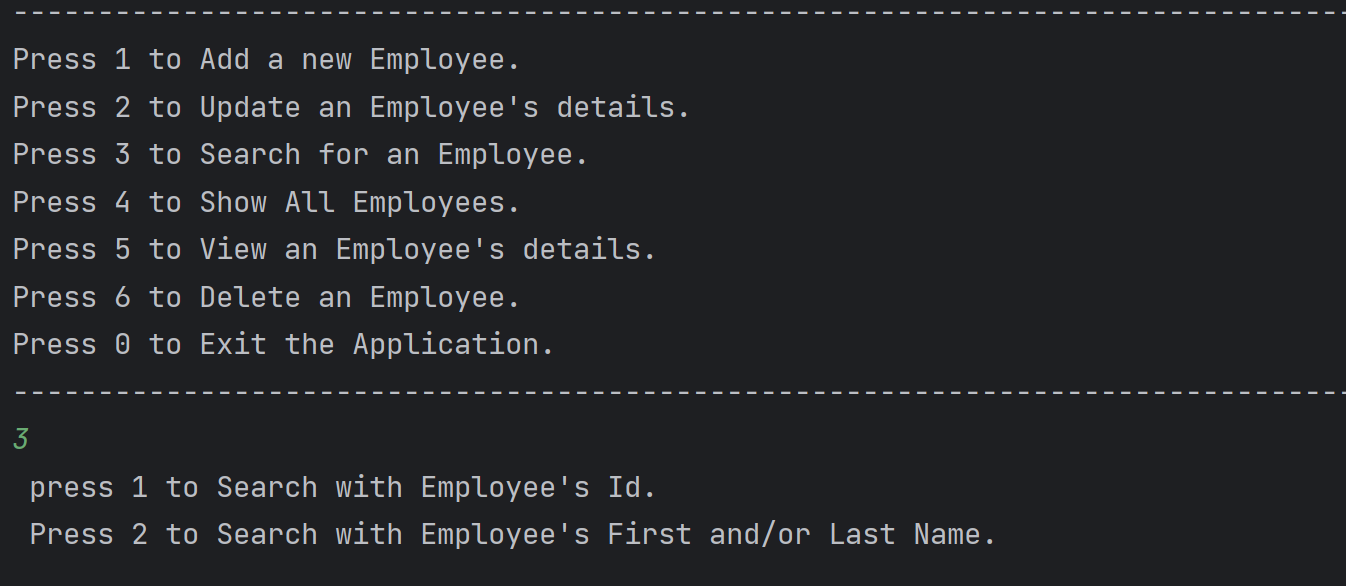
**Employee details can be retrieved as above with further options to update any field excluding the Employee ID.**

### ReactJS Application

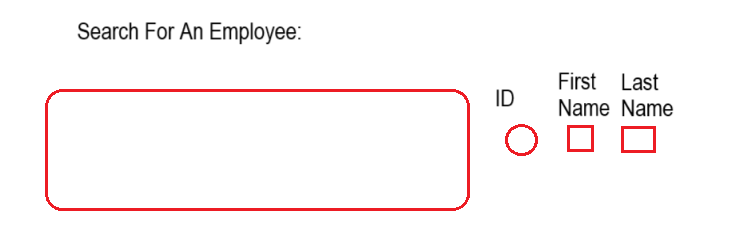


## Search Employee Module:

### CLI Application

****

### ReactJS Application

****

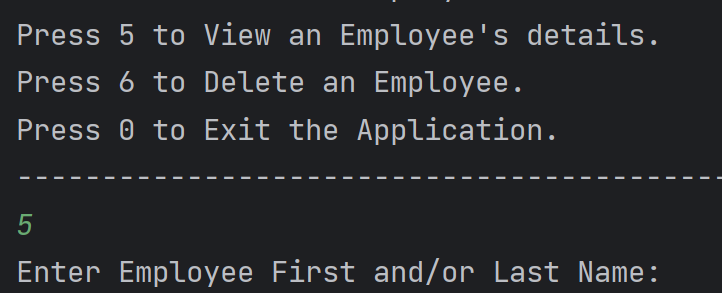
**User has the option of searching for an employee by the ID in which case he/she will select the radio button**

**Or**

**By First and/or Last Name in which case both or one of the checkboxes will be selected.**

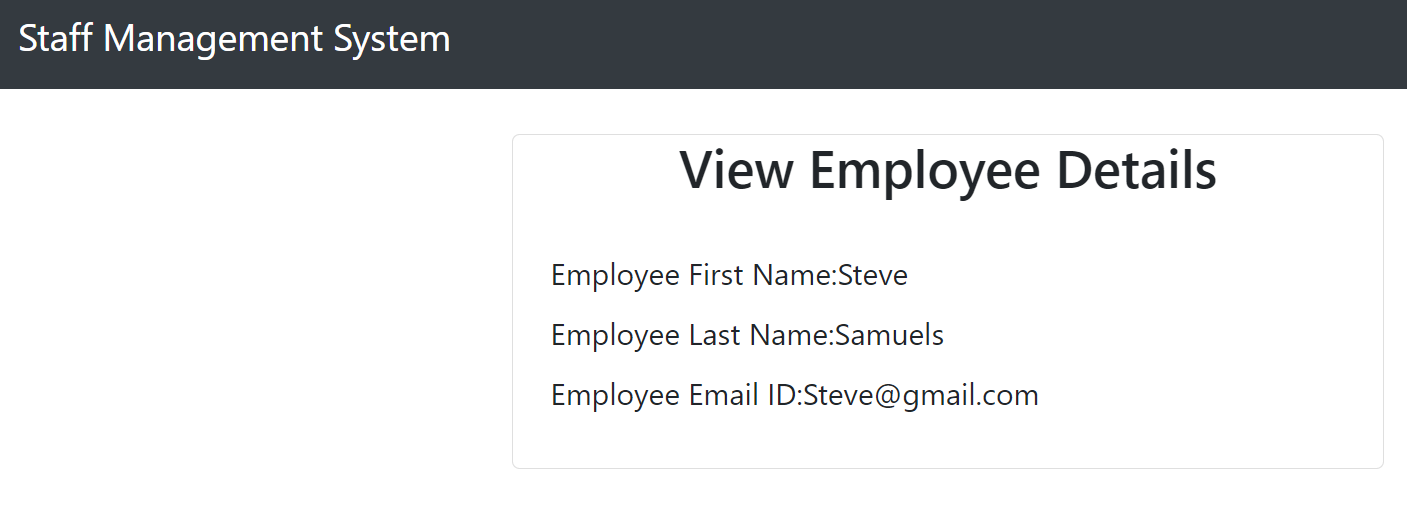
## View Employee Module:

### CLI Application



**Employee details can be retrieved as above as read-only.**

### ReactJS Application

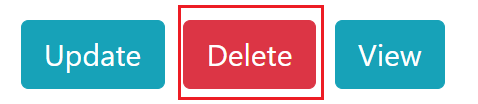
****

## Delete Employee Module:

### CLI Application

**Employee details can be retrieved as in View Employee module. Confirmation message that the employee details have been deleted successfully should be presented on screen.**

### ReactJS Application



Employee Record should be removed from the List of Employees on the web application and from the back-end.

## References:

1. ASDE\_Training\_USER\_STORIES - 2024.xlsx: – Goals and Tasks for each of five Sprints in the Learning Phase.
2. ASDE 2024 - Day-wise ToC.xlsx: TOC corresponding to each of five Sprints in the Learning Phase.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*