

Frontend UI Assignment

Objective

Develop a responsive and interactive **Employee Directory Web Interface** using **HTML, CSS, JavaScript, and Freemarker templates**. The goal is to assess your understanding of modern front-end development principles and your ability to build clean, modular, and user-friendly interfaces **without relying heavily on external APIs**.

Requirements

User Interface:

1. Dashboard Page
 - Display a list/grid of employees with the following details:
Employee ID, First Name, Last Name, Email, Department, and Role.
 - Use a **Freemarker template** to render the list (mock data will be passed to the template).
 - Each employee card/row should have an "Edit" and "Delete" button.



For more information visit www.ajackus.com

Employee Directory

Filter

Sort: --Select-- Show: 10

Add Employee

Alice Smith
Email: **alice@example.com**
Department: **HR**
Role: **Manager**
Edit Delete

Bob Johnson
Email: **bob@example.com**
Department: **IT**
Role: **Developer**
Edit Delete

Charlie Lee
Email: **charlie@example.com**
Department: **Finance**
Role: **Analyst**
Edit Delete

© 2025 Employee Directory App. All rights reserved.

2. Add/Edit Form Page

- Create a **form (HTML)** to add or edit an employee.
- This form should be styled and validated using CSS and JavaScript.
- Fields: **First Name**, **Last Name**, **Email**, **Department**, **Role**.
- Use simple client-side JavaScript to handle form submission, validation, and updating the data locally (in memory, no backend required).

Add Employee

First name

Last name

Email

Department

Role

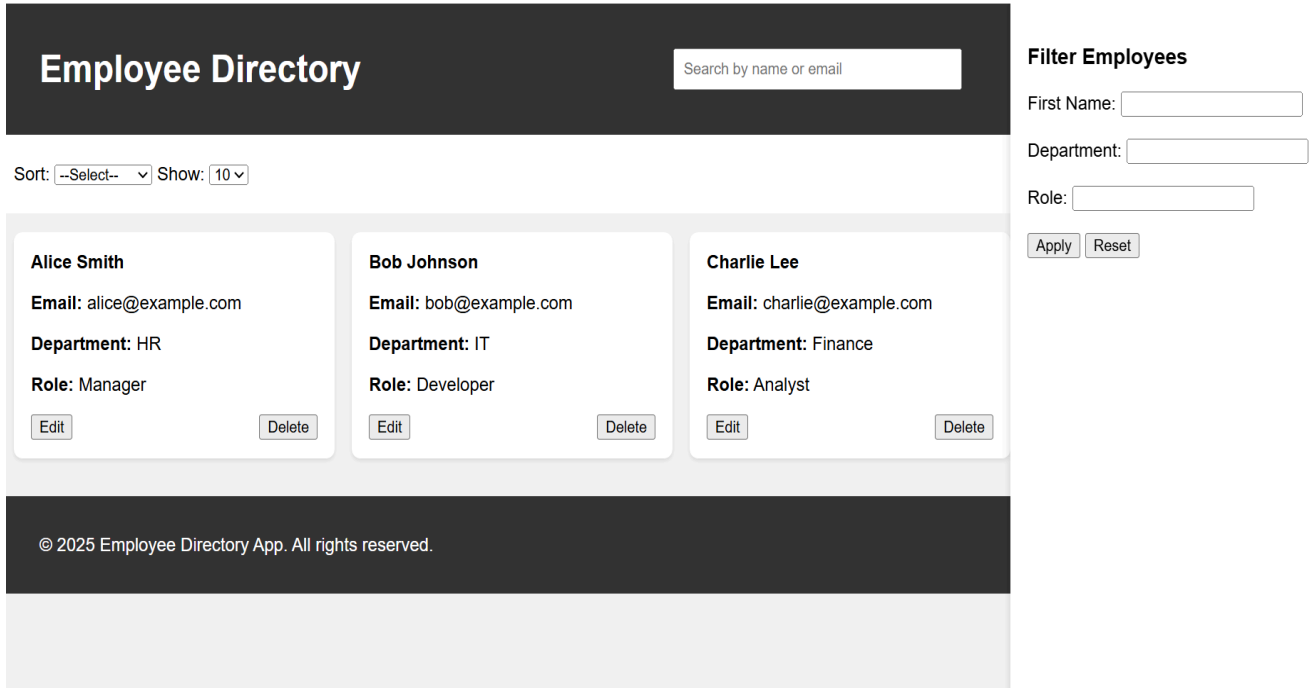
Cancel

Add

3. Filter/Sort/Search

- Add a **filter popup or sidebar** that allows filtering employees by:
 - First Name
 - Department
 - Role
- Add a search bar at the top to search employees by **name or email**.

- Add sorting capability by **First Name** and **Department**.



The mockup shows an 'Employee Directory' application. It features a dark header with the title and a search bar. Below the header, there are sorting and display controls. The main content area displays three employee cards, each with fields for name, email, department, and role, along with 'Edit' and 'Delete' buttons. A sidebar on the right provides filters for First Name, Department, and Role, with 'Apply' and 'Reset' buttons. A footer contains copyright information.

Employee Directory Search by name or email

Sort: --Select-- Show: 10

Alice Smith
Email: alice@example.com
Department: HR
Role: Manager
Edit Delete

Bob Johnson
Email: bob@example.com
Department: IT
Role: Developer
Edit Delete

Charlie Lee
Email: charlie@example.com
Department: Finance
Role: Analyst
Edit Delete

Filter Employees
First Name:
Department:
Role:
Apply Reset

© 2025 Employee Directory App. All rights reserved.

4. Pagination or Infinite Scroll
 - Implement pagination (options for 10, 25, 50, 100) or infinite scrolling using JavaScript.
5. Responsive Design
 - Ensure the app is responsive and looks good on desktop, tablet, and mobile.

Functionality

- Data Handling:
Use a local JavaScript array to simulate employee data (can be loaded via `<#assign>` tag in Freemarker from a mock JSON object).
- Form Validation:
Validate fields like email format, required fields, etc., using **JavaScript**.
- No Backend Interaction Required:
You do not need to connect to any backend API or perform real network calls.

Error Handling & Validations:

- Display validation errors clearly (e.g., invalid email, required fields).
- If the user tries to delete an employee without selecting one or editing without saving, handle these interactions gracefully.

Guidelines:

- Focus on writing clean HTML/CSS/JS.
- Use **vanilla JavaScript** (optional: minimal use of libraries if really needed).
- Use **Freemarker** where dynamic HTML rendering is required.
- Try to modularize your CSS and JS.
- Add comments in your code where necessary to explain logic.

Submission:

- Upload your project to a **public GitHub repo**.
- Include a **README** file with:
 - Setup and run instructions.
 - Overview of your project structure.
 - Screenshots (optional but recommended).
 - Reflection: Challenges faced and what you'd improve if given more time.

Timeline:

- The faster turnaround will be awarded bonus points.

Evaluation Criteria:

- HTML/CSS code quality and responsiveness.
- Use of JavaScript for DOM manipulation and interactivity.
- Freemarker integration and clean templating.
- User Experience (UX) & Interface Design (UI).
- Code structure, readability, and validation handling.