

# EMPLOYEE MANAGEMENT SYSTEM – MERN STACK

---

## **Introduction: -**

The Employee Management System is a full-stack web application designed to manage employee data efficiently. It provides a user-friendly interface for performing CRUD (Create, Read, Update, Delete) operations on employee records. The project is built using the MERN stack, which includes MongoDB, Express.js, React.js, and Node.js.

## **Features: -**

- **CRUD Operations:** Create, read, update, and delete employee records.
- **User-Friendly Interface:** A responsive and dynamic UI for managing employee data.
- **Secure Data Handling:** Authentication and authorization implemented to protect sensitive data.
- **RESTful APIs:** Backend APIs for managing data operations securely and efficiently.
- **Scalable Architecture:** Utilizes MongoDB for scalable and robust data storage.

## **Technologies Used: -**

- **MongoDB:** NoSQL database for storing employee data.
- **Express.js:** Backend framework for building RESTful APIs.
- **React.js:** Frontend library for building dynamic user interfaces.
- **Node.js:** JavaScript runtime for backend development.
- **Nodemon:** Development tool for automatically restarting the server on code changes.

## **Setup and Installation: -**

### **Prerequisites:**

- Node.js (v14.x or higher)
- MongoDB (local or cloud-based)
- Git

## **Installation Steps: -**

### **1. Clone the Repository:**

git clone [https://github.com/Kaysanshaikh/FutureIntern\\_FSD\\_02](https://github.com/Kaysanshaikh/FutureIntern_FSD_02)

cd employee-management-system

### **2. Backend Setup:**

- Navigate to the backend directory:

cd server

- Install dependencies:

npm install

Create a .env file and configure your environment variables (e.g., MongoDB URI, Port):

MONGO\_URI=your\_mongodb\_uri  
PORT=3000

- Start the backend server:

npm start

### **3. Frontend Setup:**

- Navigate to the frontend directory:

cd client

- Install dependencies:

npm install

- Start the frontend development server:

npm start

### **4. Access the Application:**

- The application should now be running on <http://localhost:3000>

## Screenshots: -

C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19045.4780]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Desktop\FutureIntern\Employee Management System\server>npm run dev

> crud@1.0.0 dev
> nodemon index.js

[nodemon] 3.1.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
server started in port 3100
Connected successfully
```

Windows PowerShell

```
Compiled with warnings.

[eslint]
src\components\LeftNav\LeftNav.jsx
  Line 20:6: React Hook useEffect has a missing dependency: 'getEmployeeById'. Either include it or remove the dependency array react-hooks/exhaustive-deps
  Line 26:9: img elements must have an alt prop, either with meaningful text, or an empty string for decorative images jsx-a11y/alt-text

Search for the keywords to learn more about each warning.
To ignore, add // eslint-disable-next-line to the line before.

WARNING in [eslint]
src\components\LeftNav\LeftNav.jsx
  Line 20:6: React Hook useEffect has a missing dependency: 'getEmployeeById'. Either include it or remove the dependency array react-hooks/exhaustive-deps
  Line 26:9: img elements must have an alt prop, either with meaningful text, or an empty string for decorative images jsx-a11y/alt-text

webpack compiled with 1 warning
```

## Code: -

JS App.js ×

client > src > JS App.js

```
1 import { useState } from 'react';
2 import './App.css';
3 import LeftNav from './components/LeftNav/LeftNav';
4 import MainSection from './components/MainSection/MainSection';
5 import TopNav from './components/TopNav/TopNav';
6
7 function App() {
8   const [employeeId, setEmployeeId] = useState('')
9   console.log(employeeId)
10  return (
11    <div className="App">
12      <TopNav/>
13      <LeftNav employeeId={employeeId}/>
14      <MainSection setEmployeeId={setEmployeeId}/>
15    </div>
16  );
17 }
18
19 export default App;
20
```

JS App.test.js ×

client > src > JS App.test.js > ...

```
1 import { render, screen } from '@testing-library/react';
2 import App from './App';
3
4 test('renders learn react link', () => {
5   render(<App />);
6   const linkElement = screen.getByText(/learn react/i);
7   expect(linkElement).toBeInTheDocument();
8 });
9
```

JS axiosServices.js X

client > src > JS axiosServices.js > ...

```
1  import axios from 'axios';
2
3  const baseURL = `http://localhost:3100`
4
5  export const axiosGet = (url) =>{
6    return axios.get(`${baseURL}${url}`, {
7      headers:{
8        "Content-Type": "application/json"
9      }
10   })
11 }
12 export const axiosPost = (url, data) =>{
13   return axios.post(`${baseURL}${url}`, data, {
14     headers:{
15       "Content-Type": "application/json"
16     }
17   })
18 }
19 export const axiosDelete = (url) =>{
20   return axios.delete(`${baseURL}${url}`, {
21     headers:{
22       "Content-Type": "application/json"
23     }
24   })
25 }
26 export const axiosPut = (url, data) =>{
27   return axios.put(`${baseURL}${url}`, data, {
28     headers:{
29       "Content-Type": "application/json"
30     }
31   })
32 }
```

# index.css X

client > src > # index.css > ...

```
1  @import url('https://fonts.googleapis.com/css2?family=Inter:wght@200;300;400;500;600;700;800;900&display=swap');
2
3  *{
4    padding: 0;
5    margin: 0;
6  }
7
8  body {
9    margin: 0;
10   font-family: 'Inter', sans-serif;
11   background-color: #F7F6FB;
12   width: 100%;
13   height: 100vh;
14 }
15
16 code {
17   font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
18   monospace;
19 }
20
```

JS index.js X

client > src > JS index.js > ...

```
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3 import './index.css';
4 import App from './App';
5 import reportWebVitals from './reportWebVitals';
6
7 const root = ReactDOM.createRoot(document.getElementById('root'));
8 root.render(
9   <React.StrictMode>
10     <App />
11   </React.StrictMode>
12 );
13
14 // If you want to start measuring performance in your app, pass a function
15 // to log results (for example: reportWebVitals(console.log))
16 // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
17 reportWebVitals();
18
```

logo.svg X

client > src > logo.svg

```
1 <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 841.9 595.3"><g fill="#61DAFB"><path d="M666.3 296.5c0-32.5-40.7-63.3-103.1-82.4 14.4-63.6 8-114.2-20.2-130.4-6.5-3.8-14.1-5.6-22.4-5.6v22.3c4
```

JS reportWebVitals.js X

client > src > JS reportWebVitals.js > ...

```
1 const reportWebVitals = onPerfEntry => {
2   if (onPerfEntry && onPerfEntry instanceof Function) {
3     import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {
4       getCLS(onPerfEntry);
5       getFID(onPerfEntry);
6       getFCP(onPerfEntry);
7       getLCP(onPerfEntry);
8       getTTFB(onPerfEntry);
9     });
10  }
11 };
12
13 export default reportWebVitals;
14
```

JS setupTests.js X

client > src > JS setupTests.js

```
1 // jest-dom adds custom jest matchers for asserting on DOM nodes.
2 // allows you to do things like:
3 // expect(element).toHaveTextContent(/react/i)
4 // learn more: https://github.com/testing-library/jest-dom
5 import '@testing-library/jest-dom';
6
```

<> index.html ×

client > public > <> index.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="utf-8" />
5      <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
6      <meta name="viewport" content="width=device-width, initial-scale=1" />
7      <meta name="theme-color" content="#000000" />
8      <meta
9        name="description"
10       content="Web site created using create-react-app"
11     />
12     <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
13     <!--
14       manifest.json provides metadata used when your web app is installed on a
15       user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
16     -->
17     <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
18     <!--
19       Notice the use of %PUBLIC_URL% in the tags above.
20       It will be replaced with the URL of the `public` folder during the build.
21       Only files inside the `public` folder can be referenced from the HTML.
22
23       Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
24       work correctly both with client-side routing and a non-root public URL.
25       Learn how to configure a non-root public URL by running `npm run build`.
26     -->
27     <title>React App</title>
28   </head>
29   <body>
30     <noscript>You need to enable JavaScript to run this app.</noscript>
31     <div id="root"></div>
32     <!--
33       This HTML file is a template.
34       If you open it directly in the browser, you will see an empty page.
35
36       You can add webfonts, meta tags, or analytics to this file.
37       The build step will place the bundled scripts into the <body> tag.
38
39       To begin the development, run `npm start` or `yarn start`.
40       To create a production bundle, use `npm run build` or `yarn build`.
41     -->
42   </body>
43 </html>
44
```

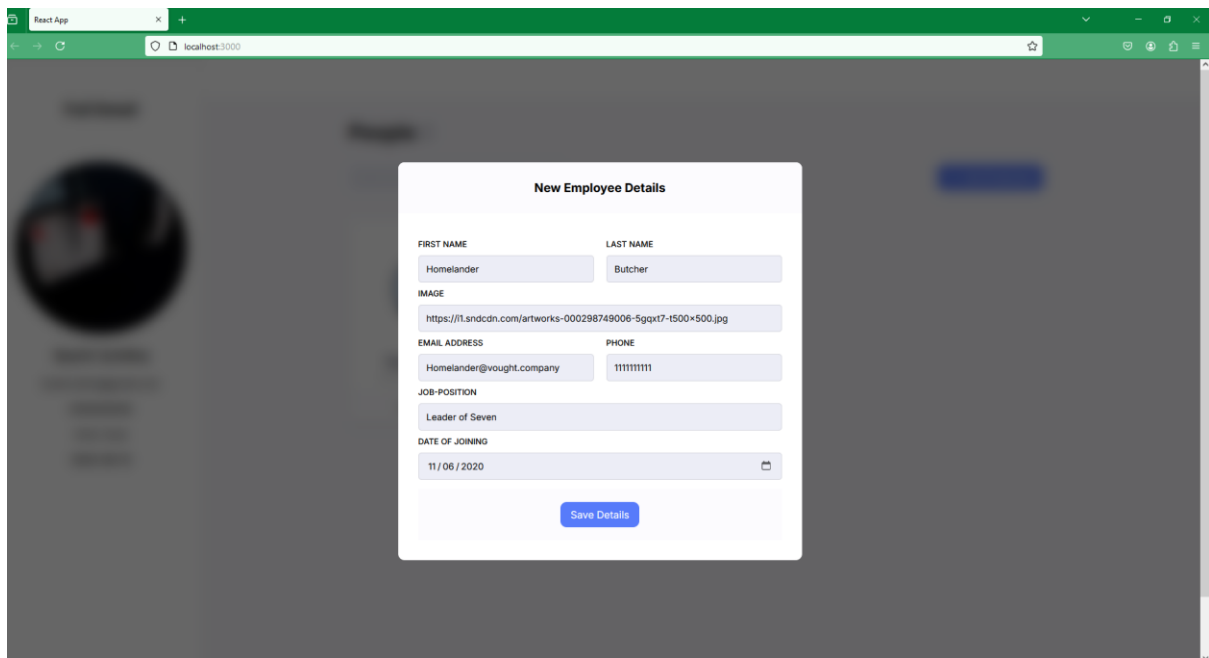
{ } manifest.json ×

client > public > { } manifest.json > ...

```
1  {
2    "short_name": "React App",
3    "name": "Create React App Sample",
4    "icons": [
5      {
6        "src": "favicon.ico",
7        "sizes": "64x64 32x32 24x24 16x16",
8        "type": "image/x-icon"
9      },
10     {
11       "src": "logo192.png",
12       "type": "image/png",
13       "sizes": "192x192"
14     },
15     {
16       "src": "logo512.png",
17       "type": "image/png",
18       "sizes": "512x512"
19     }
20   ],
21   "start_url": ".",
22   "display": "standalone",
23   "theme_color": "#000000",
24   "background_color": "#ffffff"
25 }
26
```

## Output: -

### Create:



**New Employee Details**

FIRST NAME: Homelander

LAST NAME: Butcher

IMAGE: <https://i1.sndcdn.com/artworks-000298749006-5gxt7-t500x500.jpg>

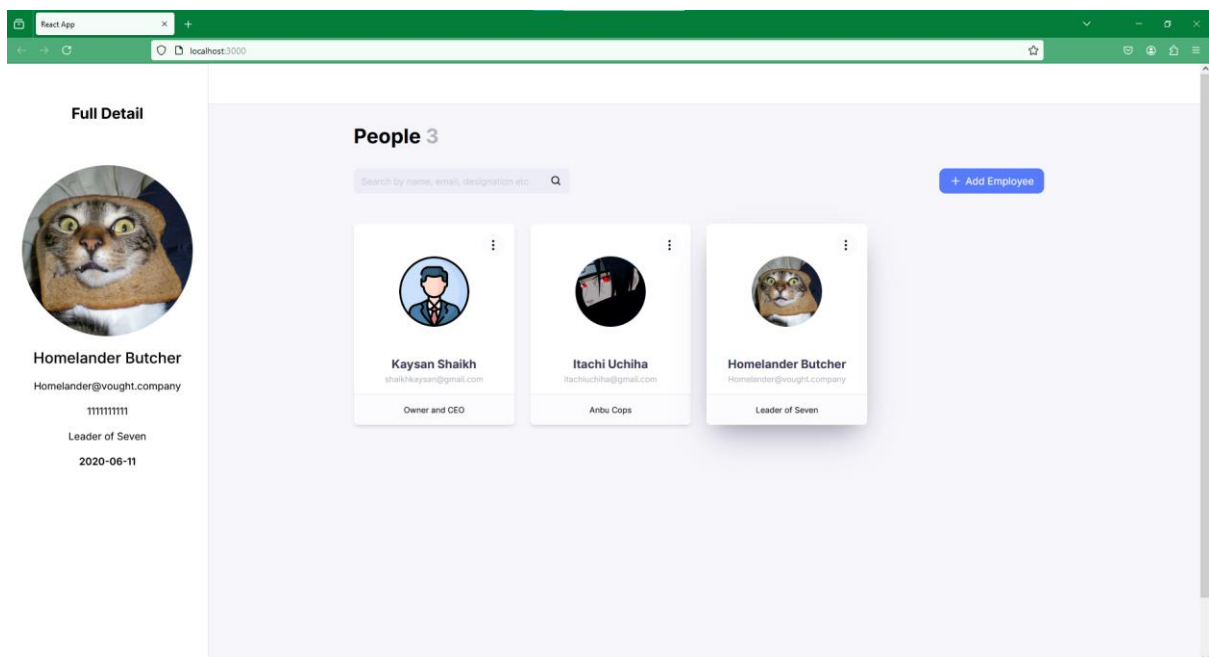
EMAIL ADDRESS: Homelander@vought.company

PHONE: 1111111111


JOB-POSITION: Leader of Seven

DATE OF JOINING: 11/06/2020

Save Details



**Full Detail**




**Homelander Butcher**  
Homelander@vought.company  
1111111111  
Leader of Seven  
2020-06-11


**People 3**

Search by name, email, designation etc.


[+ Add Employee](#)



**Kaysan Shaikh**  
shaikhsayang@gmail.com  
Owner and CEO



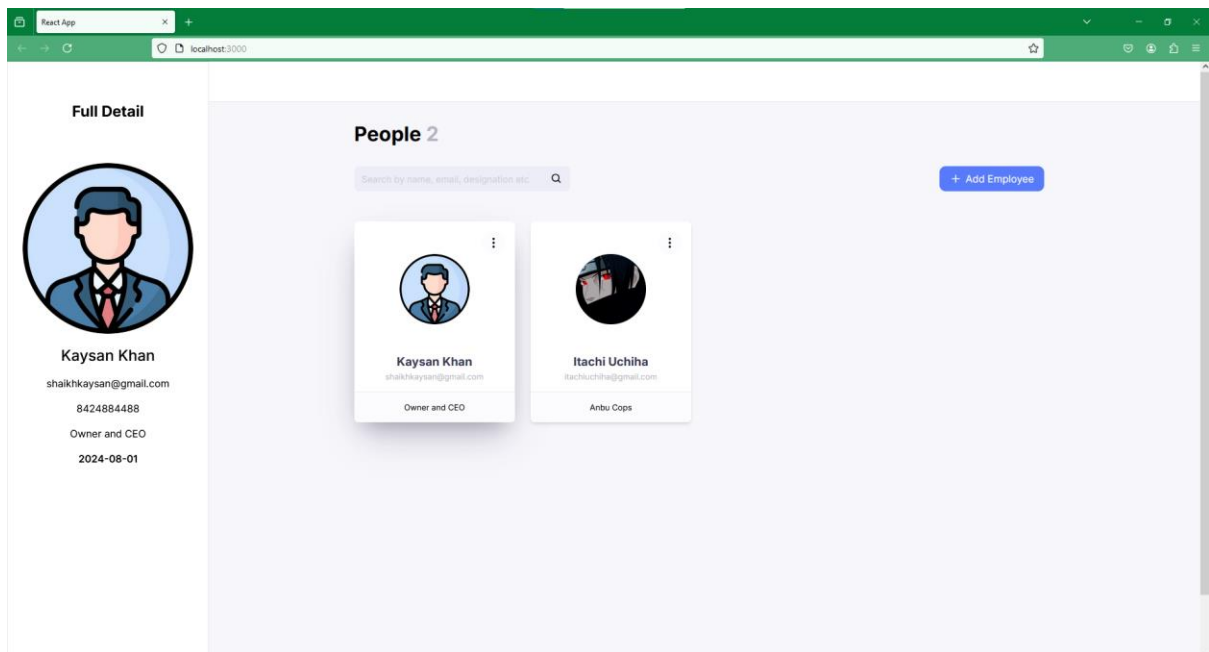
**Itachi Uchiha**  
itachuchiha@gmail.com  
Anbu Cops



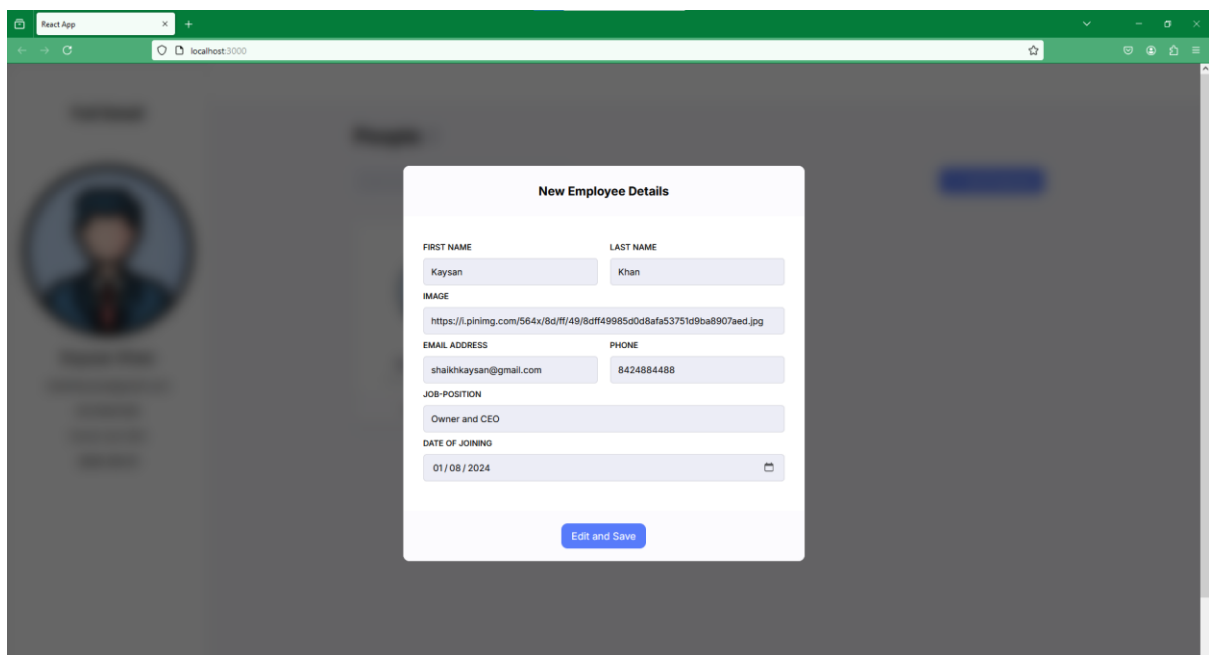
**Homelander Butcher**  
Homelander@vought.company  
Leader of Seven



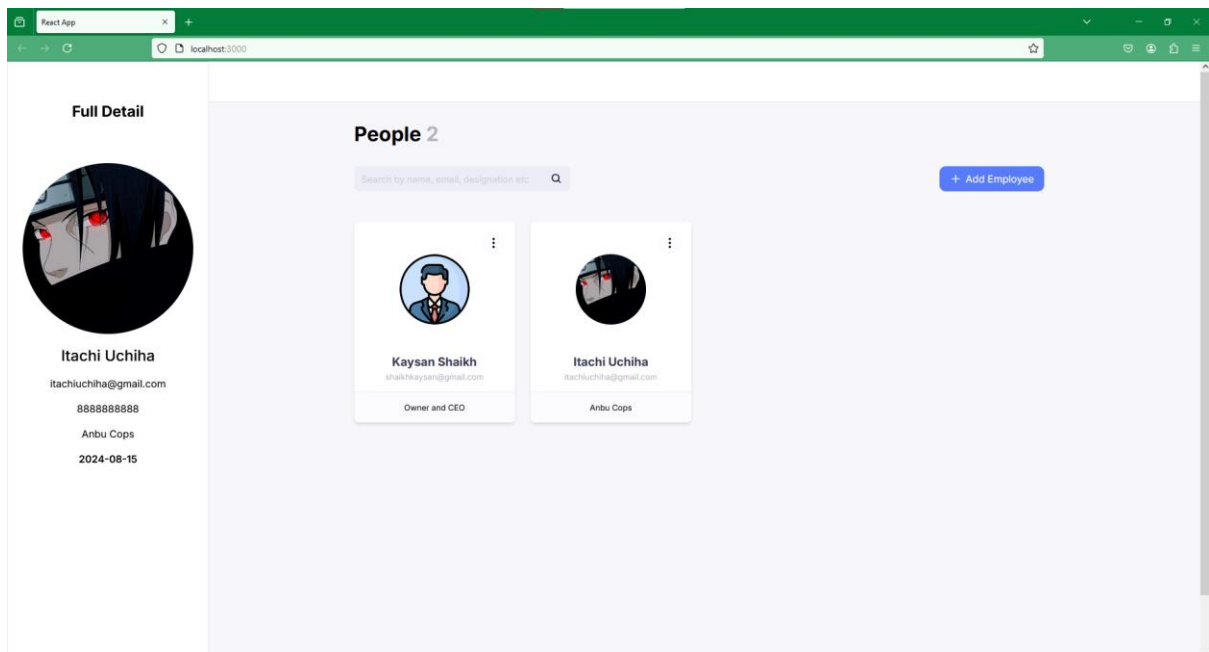
## Delete:



## Update:



## Read:



## API Endpoints: -

- **GET /api/employees:** Retrieve all employees.
- **GET /api/employees/**  
: Retrieve a specific employee by ID.
- **POST /api/employees:** Create a new employee.
- **PUT /api/employees/**  
: Update an existing employee by ID.
- **DELETE /api/employees/**  
: Delete an employee by ID.

## Usage: -

1. **Create Employee:** Navigate to the "Add Employee" page, fill in the required details, and submit the form.
2. **View Employees:** Access the "Employee List" page to view all registered employees.
3. **Update Employee:** On the "Employee List" page, select an employee to edit their details.
4. **Delete Employee:** Remove an employee by clicking the delete button on their record in the "Employee List" page.

### **Security Considerations: -**

- **Authentication:** Secure access to sensitive routes and operations.
- **Authorization:** Implemented role-based access control to ensure only authorized users can perform certain actions.
- **Error Handling:** Graceful handling of errors to avoid exposing sensitive information.

### **Future Enhancements: -**

- **Pagination and Sorting:** Implement pagination and sorting for large datasets.
- **Advanced Role Management:** Extend role-based access control to more granular permissions.

### **Contributing: -**

- Contributions are welcome! Please fork the repository and submit a pull request with your changes.