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/Kaysanshaikh-FutureIntern_FSD_02.git
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
     PORT=3100
     ```

   - 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

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.