# Employee Management System (EMS)

**Github Repo :-** https://github.com/Dilipkumar41/EmployeeManagementSystem-Sprint

## Overview

The Employee Management System (EMS) is a **full-stack web application** designed to streamline HR, Manager, and Employee workflows.

It provides **role-based access control (RBAC)** for HR, Managers, and Employees to manage employee data, leave requests, departments, and user roles.

## 🛠️ Tech Stack

- **Frontend:** React, Redux, Vite, TailwindCSS

- **Backend:** .NET Core Web API (C#), Entity Framework Core

- **Database:** SQL Server (SSMS)

- **Authentication:** JWT-based Authentication & Role-Based Access Control (RBAC)

### HR Features

- Manage employees (**CRUD operations**).

- Manage departments (**CRUD operations**).

- Assign and update roles (**HR, Manager, Employee**).

- Approve / Reject leave requests.

- View reports (Employees, Departments, Leave history).

- Manage users (create, delete, update roles).

- View their own **profile and leave history**.

### Manager Features

- View **My Profile**.

- View and manage **department employees**.

- Approve / Reject **team leave requests**.

- Apply and track **own leaves**.

### 👨‍🔧 Employee Features

- View **My Profile**.

- Apply for **leave requests**.

- Track leave request status.

---

## 🗂️ Project Structure

### Frontend (`/frontend/src`)

```
src/
├── api/            # API clients (axios wrappers)
├── app/            # Redux store & rootReducer
├── components/     # Shared UI components
├── features/       # Feature-based slices & pages
│   ├── auth/       # Authentication (login/register, role guards)
│   ├── employees/  # Employees CRUD
│   ├── departments/  # Departments CRUD
│   ├── leaves/     # Leaves management
│   ├── me/         # Logged-in user profile
│   ├── roles/      # Role management
│   └── users/      # User management
├── layouts/        # Layouts (Main, Auth)
├── pages/          # Dashboard, Login, Register, etc.
├── routes/         # AppRoutes for navigation
├── styles/         # CSS files
├── App.jsx         # Main React App
├── main.jsx        # Entry point
```

### Backend (`/backend/EmployeeManagementSystem`)

```
EmployeeManagementSystem/
├── Controllers/      # API Controllers
├── Data/             # DbContext
├── DTOs/             # Data Transfer Objects
├── Models/           # Database Models
├── Repositories/     # Repository Pattern
├── Services/         # Business Logic Services
├── Program.cs        # App Startup
├── appsettings.json  # Configurations
```

### Database (`EMS.sql`)

- Roles
- Users
- Departments
- Employees
- Leaves

---

## ⚡ API Endpoints

### Authentication

- `POST /api/auth/register` → Register a new user
- `POST /api/auth/login` → Login and get JWT token

### Users

- `GET /api/users` → Get all users
- `GET /api/users/{id}` → Get user by ID
- `PUT /api/users/{id}/role` → Update user role

- `DELETE /api/users/{id}` → Delete user

### Employees
- `POST /api/employees` → Create new employee
- `GET /api/employees` → Get all employees
- `GET /api/employees/{id}` → Get employee by ID
- `PUT /api/employees/{id}` → Update employee
- `DELETE /api/employees/{id}` → Delete employee

### Departments
- `POST /api/departments` → Create new department
- `GET /api/departments` → Get all departments
- `PUT /api/departments/{id}` → Update department
- `DELETE /api/departments/{id}` → Delete department

### Leaves
- `POST /api/leaves` → Apply leave
- `GET /api/leaves` → Get all leaves
- `PUT /api/leaves/{id}/status` → Approve/Reject leave
- `DELETE /api/leaves/{id}` → Delete leave

---

## ⚙️ Setup & Installation

### Backend
1. Open the project in **Visual Studio / Rider**.
2. Update `appsettings.json` with your SQL Server connection string.
3. Run migrations:
   ```
   dotnet ef database update
   ```

```

4. Run the API:

```
dotnet run
```

5. Test in Swagger: `https://localhost:7061/swagger`

### Frontend

1. Navigate to `frontend/`.

2. Install dependencies:

```
npm install
```

3. Start development server:

```
npm run dev
```

4. Open: `http://localhost:5173`

---

## 📊 Dummy Users for Testing

| Role | Email | Password |
|-----------|----------------------|------------|
| HR | hr@company.com | 123456 |
| Manager | manager@company.com | 123456 |
| Employee | employee@company.com | 123456 |
```