Employee Management System

Project Overview

The Employee Management System is a Spring Boot REST API application that manages employee records. It provides CRUD (Create, Read, Update, Delete) operations for handling employee data and integrates with a relational database using Spring Data JPA.

Features

- Add, view, update, and delete employees
- RESTful API architecture
- Exception handling
- Tested using Postman
- Easily extensible with authentication and frontend UI

Technologies Used

- Java 17+
- Spring Boot
- Spring Data JPA
- MySQL / H2
- Maven
- Postman

Project Structure

```
src/
├── main/
│   ├── java/com/ems/
│   │   ├── controller/EmployeeController.java
│   │   ├── exception/ResourceNotFoundException.java
│   │   ├── model/Employee.java
│   │   ├── repository/EmployeeRepository.java
│   │   └── EmployeeManagementSystemApplication.java
│   └── resources/application.properties
└── test/java/com/ems/EmployeeManagementSystemApplicationTests.java
```

How to Run the Project

1. Import the project into IntelliJ or Eclipse.
2. Update `application.properties` with your MySQL credentials.
3. Run `EmployeeManagementSystemApplication.java` as a Spring Boot app.
4. Use Postman to test API endpoints.

API Endpoints

GET /api/v1/employees → List all employees
GET /api/v1/employees/{id} → Get employee by ID
POST /api/v1/employees → Add a new employee
PUT /api/v1/employees/{id} → Update employee by ID
DELETE /api/v1/employees/{id} → Delete employee by ID

Sample Request (POST)

```
{
 "firstName": "John",
 "lastName": "Doe",
 "emailId": "john.doe@example.com"
}
```

Authentication

Currently not implemented. Can be added using Spring Security with JWT or basic authentication.

Postman Testing

Use Postman to test the endpoints with raw JSON requests and observe responses. Make sure the server is running locally on `http://localhost:8080`.