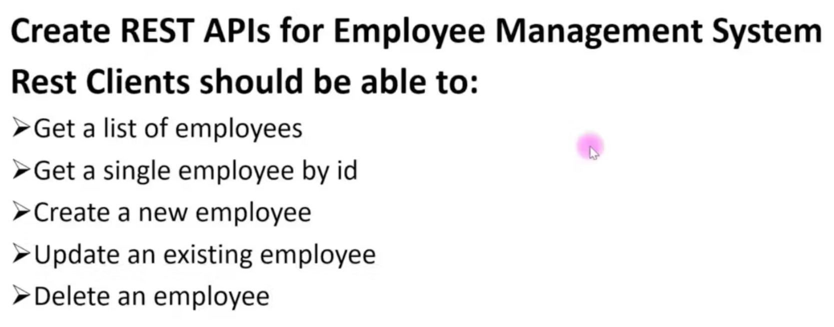
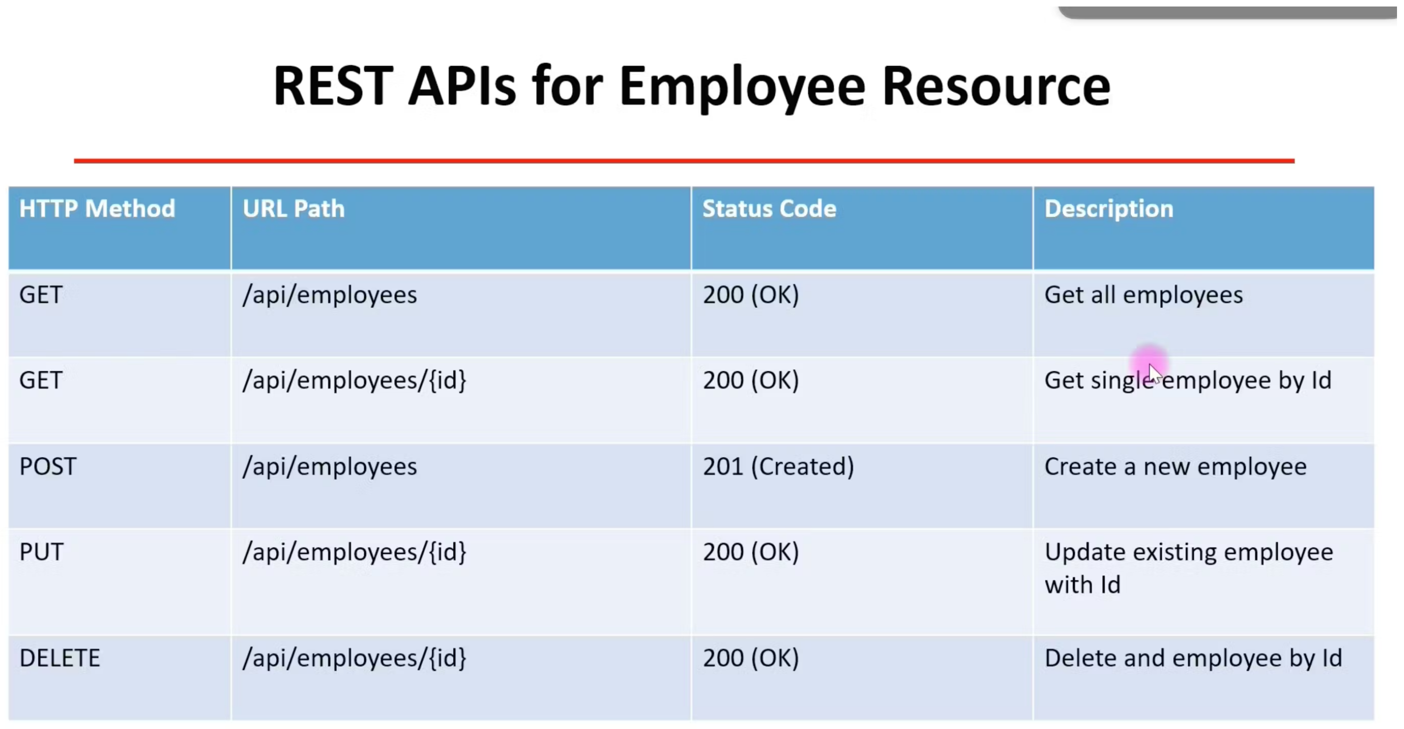
Spring Boot

Client Requirements API


Create REST APIs for Employee Management System
Rest Clients should be able to:

- ➢Get a list of employees
- ➢Get a single employee by id
- ➢Create a new employee
- ➢Update an existing employee
- ➢Delete an employee

Step 1: Tabulate what API calls are necessary GET, POST, PUT, DELETE.

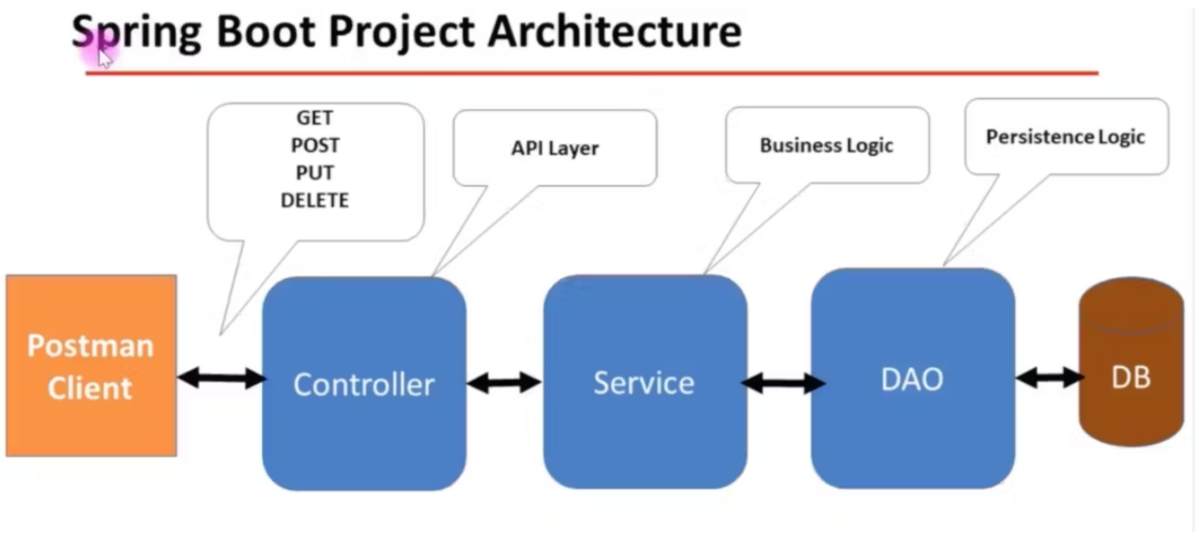

REST APIs for Employee Resource

HTTP Method	URL Path	Status Code	Description
GET	/api/employees	200 (OK)	Get all employees
GET	/api/employees/{id}	200 (OK)	Get single employee by Id
POST	/api/employees	201 (Created)	Create a new employee
PUT	/api/employees/{id}	200 (OK)	Update existing employee with Id
DELETE	/api/employees/{id}	200 (OK)	Delete and employee by Id

The spring arch has

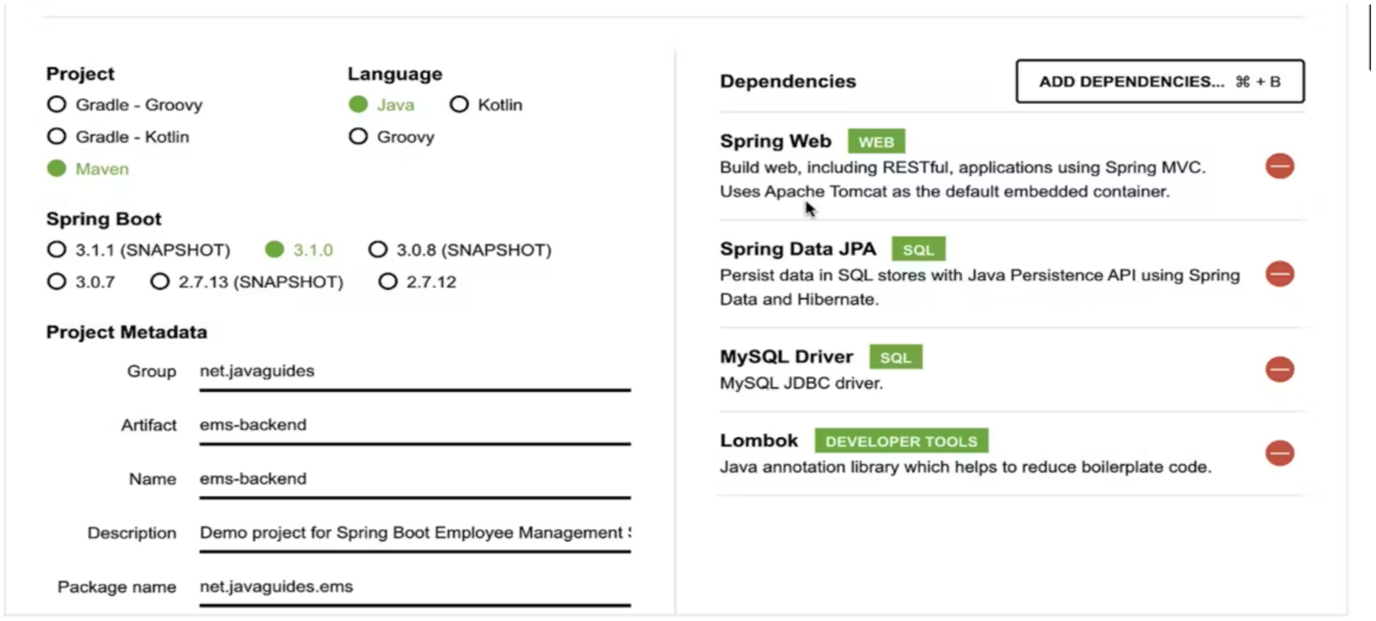
1. Controller
2. Service
3. Database Manager Layers

Step 1.1: Design the Project Arch to understand the flow



Step 2: Reference of Dependencies while creating a spring boot project.

Step 2.1: visit start.spring.io and select following options



Project
○ Gradle - Groovy
○ Gradle - Kotlin
● Maven

Language
● Java ○ Kotlin
○ Groovy

Spring Boot
○ 3.1.1 (SNAPSHOT) ● 3.1.0 ○ 3.0.8 (SNAPSHOT)
○ 3.0.7 ○ 2.7.13 (SNAPSHOT) ○ 2.7.12

Project Metadata

Group	net.javaguides
Artifact	ems-backend
Name	ems-backend
Description	Demo project for Spring Boot Employee Management !
Package name	net.javaguides.ems

Dependencies ADD DEPENDENCIES... ⌘ + B

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

MySQL Driver SQL
MySQL JDBC driver.

Lombok DEVELOPER TOOLS
Java annotation library which helps to reduce boilerplate code.

Create a zip file and open in IDE.

Step: 2.2 Now configure DB

Step:2.2.1 Create DB in Workbench

Step:2.2.2 go to src/main/resources/application.properties and add

`spring.database.url=jdbc:mysql://localhost:3306/ems`

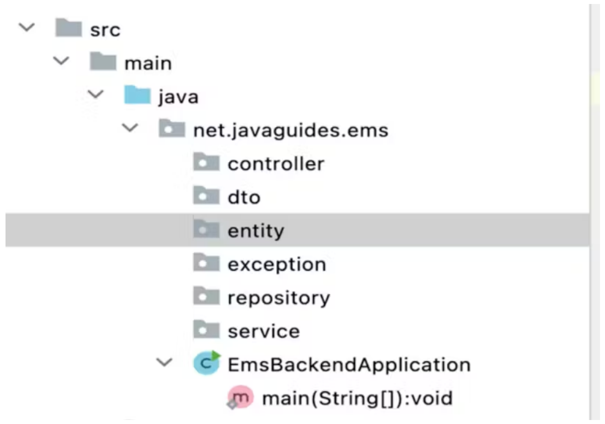
`spring.database.username=root`

`spring.database.password=pass`

`Now, Hibernate: spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect` uses this Dialect for selected DB

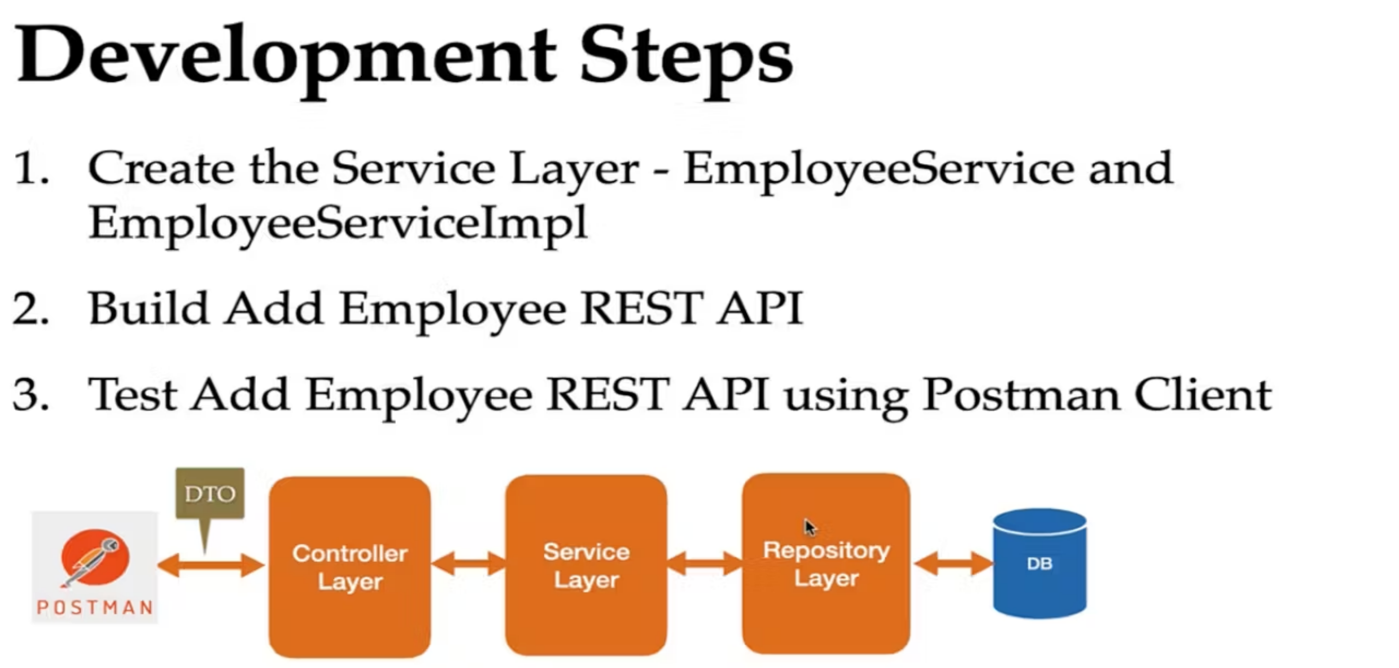
`spring.jpa.hibernate.ddl-auto=update` → create tables or update Tables depending JPA entities

Step: 2.3 Configure Package for whole project in default package manager



```
src
  main
    java
      net.javaguides.ems
        controller
        dto
        entity
        exception
        repository
        service
        EmsBackendApplication
          main(String[]):void
```

Step : 3 Connecting to DB and writing REST API's

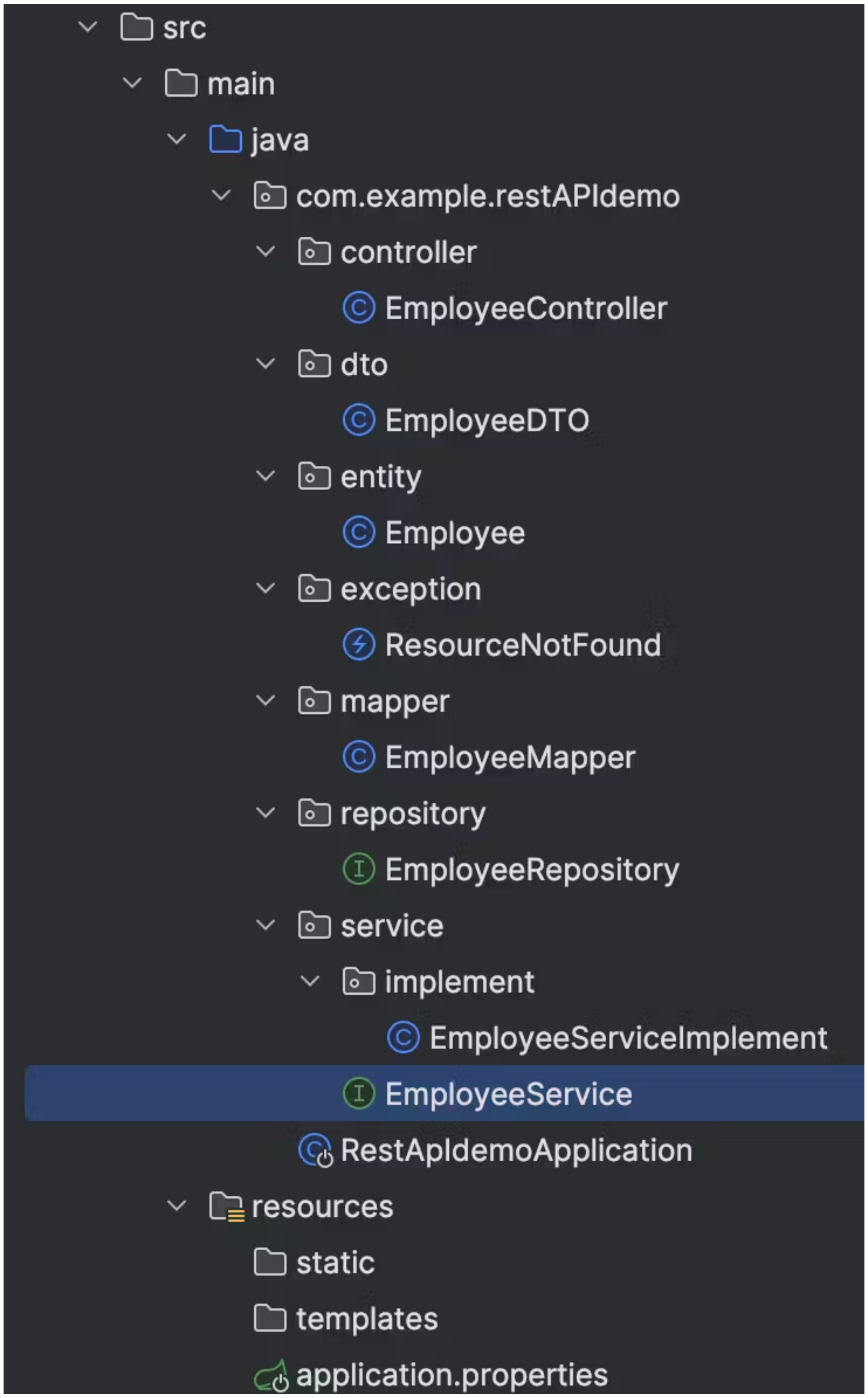


Development Steps

1. Create the Service Layer - EmployeeService and EmployeeServiceImpl
2. Build Add Employee REST API
3. Test Add Employee REST API using Postman Client

POSTMAN
DTO
Controller Layer
Service Layer
Repository Layer
DB

Step: 4 Finally after successful Connection the file structure should be as below.



further EmployeeController class will connect to REST API which is connected to EmployeeService interface which is implemented in EmployeeServiceImplement.