**Document on User Project**

Set up the Spring Boot project along with hibernate/JPA with the help of Maven, in this project There are 5 levels of stages like

1)Create a User

2)update the user

3)delete the user

4) get single userDetails by using user id

5)get all the users based on pagination

**1)Create a USER**

* 1. create a new controller class called NewUserController and in that class write a method for creating a user. In this method write logic to get the new user details like userName, UserAge and UserSalary from the postman or by using swagger in the form of json/XML by using URL by using POST request
  2. Now from the NewUserController the UserDetails should navigate to NewUSerService/DAO where user details should be saved in the Database (MySQL) by using JPA/Hibernate
  3. Write an Entity class called UserEntity which should map with table in the database, in the Entity class (UserEntity) it should generate an ID by the Hibernate which should insert ID as into the Database and return the status code as Success if user get inserted in the database table

**2)Update the User**

2.1) create a new Controller class called UpdateUserController in this class write a method to update the UserDetails.

In this method get user id (path variable) and Updated user details (Request body) from the postman or by using swagger

in the form of XML/json and first check whether the user is available or not by using user id by sending the id to the UserFindService/DAO class and method which is used to find the user

a) if the user is not available throw an exception saying user is not available

b) if the user is available then sent the Updated details to UserUpdateService/DAO class, userUpdate method by using the user

details in the database and send the success status code to Swagger or Postman along with the updated user details

**3)Delete a user**

3.1) create a new Controller class called DeleteUserController in this class write a method to delete the UserDetails.

In this method get user id (path variable) and Updated user details (Request body) from the postman or by using swagger

in the form of XML/json and first check whether the user is available or not by using user id by sending the id to the

UserFindService/DAO class and method which is used to find the user

a) if the user is not available throw an exception saying user is not available

b) if the user is available then send the id to UserDeleteService/DAO class use Delete method by using the user id delete the

User from the user table and send the success status code to Swagger or postman

**4)get single userDetails by using user id**

4.1) create a new Controller class called GetUserController in this class write a method to get the user details.

In this method get user id (path variable) from the postman or by using Swagger in the form of XML/json and check whether the user is available or not by using user id by sending the id to the

UserFindService/DAO class and method which is used to find the user

a) if the user is not available throw an exception saying user is not available

b) if the user details are available then get the UserDetails from the Database server and then send the user details to the Controller and UI (Swagger or postman) int the form of JSON/XML and success response

**5)get multiple users based on pagination**

5.1) As the GetUserController class is available in this class write 2nd method to get the userdetails bases on number of rows (no of users),

page no and order by (based on name or id or salary can be given in the form of string) from postman or by using Swagger int form of XML or json.

Send the details to UserFindService/DAO class

a) if the Users are not available in database throw an exception with the status code

b) if the Users are available send the details in the form of JSON/XML and success response

**Test case**

Write a test case by using Junit, RestAssured and Mockito

Scenario 1)

write a testcase to get User from DB by using Mockito any integer as id and check with user details as id, age, name, salary as mock and by using RestAssured check which accepts json object and get method URL, which matches the status as 200 and return body should be equal to object

Scenario 2)

Write a test case to create a saved user and test user so that the mock object has the save user object that should match with the test user which is given if it is true then return success status code

Create a REST API for user

Client can:

* Get a list of user
* Get a single user by id
* Add a new user
* Update
* Delete

HTTP method API

Process

1. Set up database dev Environment
2. Create Spring Boot
3. Get a list of User
4. Get single user by ID

Service method

1. Add a new ID
2. Update
3. Delete

4，5，RestController method

Employee

REST

Controller

Employee

DAO

（Hibernate）

Employee

Service

519节

SpringBoot will automatically configure data source

Based on entries from Maven pom file：

JDBC Driver:

Spring DATA (ORM)

DB connection info from *application.properties*

JPA

Java Persistence API: standard API for object-to-Relation-Mapping (ORM)

Only a specification

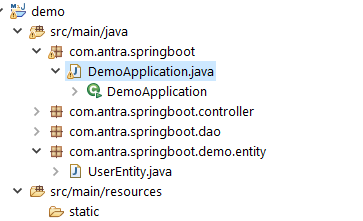
Define a set of interfaces

Requires an implementation to be usable

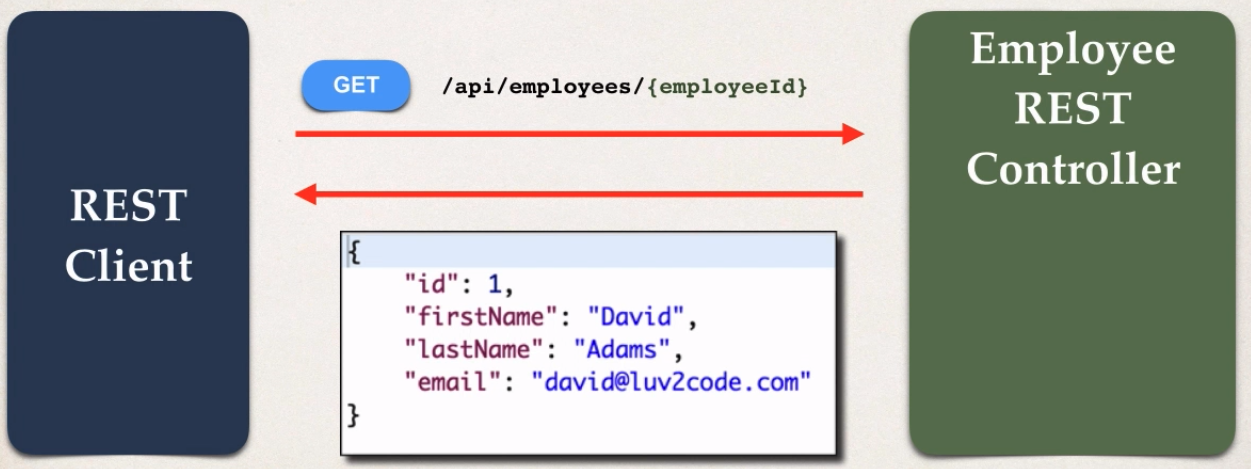
Development Process:

1. Update db Configs in application.properties
2. Create user Entity
3. Create DAO interface
4. Create DAO implementation
5. Create REST controller to use DAO

注意package的层级关系



Read a Single User



Create a new user

