Homework 2

Development instructions

I followed the steps below to complete the project.

Step 1: I went to start.spring.io and selected the required tools for my project, I chose Maven as the build tool and Java version 8. I also specified the group and artifact ID related to my project, which is "Restaurant.".

Step 2: I added the necessary dependencies, including Spring Web, Spring Dev Tools, H2, Lombok, Spring Data JPA, and Spring Data REST. Then, I clicked on "Generate" to create the project.

Step 3: I imported the project into Eclipse by selecting "Import Projects," then "Maven," and finally "Existing Maven Project." I provided the path to the project and selected the pom.xml file.

Step 4: I created packages for the controller, service, repository, and model.

Step 5: I created a POJO class named "Restaurant" to represent the data in the table. It includes fields such as "id," "name," "type," "size," and "price." I also defined two Enum classes, "Size" and "Type."

Step 6: Leveraging the Lombok dependency, I added annotations such as @Entity to indicate that this is a database class, @Table for specifying the table name, and @Getter and @Setter to generate getter and setter methods automatically. I used @NoArgsConstructor to generate a no-argument constructor and @AllArgsConstructor for an all-argument constructor.

Step 7: I created the RestaurantRepositoryI interface by extending the CrudRepository interface. This interface handles all database operations, including find, findAll, save, saveAll, delete, and count.

Step 8: I created the RestaurantServiceI interface, autowired the repository class, and declared all the CRUD method signatures.

Step 9: I implemented the RestaurantService class, which includes the methods defined in the interface, such as find, findAll, save, saveMany, delete, deleteAll, update, getByType, and count.

Step 10: I created the RestaurantController class and autowired the service class. I added method calls corresponding to those in the service class and included HTTP status codes.

Step 11: I used ResponseEntity to set the status codes for responses in Postman.

Step 12: I pushed the code to Bitbucket using Git commands, including "git add," "git commit -m," and "git push."

project purpose

I chose to work on Restaurant Data. I planned to store and analyze menu data. I created REST APIs to add data, retrieve data by ID, update data, delete data by ID, and obtain the count of records. I also analyzed recipes categorized as Appetizers, Entrees, Desserts, and Drinks. I developed an API that returns data by type. For instance, when specifying "Dessert" as the type, the API retrieves all dessert-related data.

Screenshots

I added the following screenshots to the project documentation:

* Spring application console
* Successful responses of CRUD APIs
* GetType API
* Examples of BadRequest and other error responses

Application Up and RunningA screen shot of a computer

Description automatically generated

Add Data API

A screenshot of a computer

Description automatically generated

Add Many Records

A screenshot of a computer

Description automatically generated

Get All Records

A screenshot of a computer

Description automatically generated

Get Record By Id

A screenshot of a computer

Description automatically generated

Update Record

A screenshot of a computer

Description automatically generated

Delete By Id

A screenshot of a computer

Description automatically generated

Delete All Records

A screenshot of a computer

Description automatically generated

Get the Count of all the records present in Database

A screenshot of a computer

Description automatically generated

Get records of specific Type

A screenshot of a computer

Description automatically generated

Some error Responses

Get Records (When there is not data in data base)

A screenshot of a computer

Description automatically generated