Pizza Service

I made this project as my main task at the Tel-ran course.

It contains:

**Entities:**

Pizza;

Café;

(enum) Size.

**Repositories:**

Café Repository;

Pizza Repository.

**Web Security Configuration** for Admin role.

**CaféDto** in order to show Café details without List of Pizza.

Necessary covered **Tests** for my Project.

The pizza project functionality includes the ability to create, read, update, and delete records for both pizzas and cafes. The functionality is secured by requiring a username of "admin" to access these features.

**Pizza Controller**: Users with the "admin" username is able to create new pizza records by providing information such as the pizza's name, size, key\_ingredients, cafe\_id. They are also able to view, update, and delete existing pizza records.

**Cafe Controller**: Users with the "admin" username are able to create new cafe records by providing information such as the cafe's name, location, and phone. They are also able to view, update, and delete existing cafe records.

**POST, DELETE and PUT requests**: Users with the "admin" username is able to create new resources by sending a POST request, delete existing resources by sending a DELETE request, and update existing resources by sending a PUT request.

GET is working for all.

**Security**: The application is secured by requiring a username of "admin" to access the pizza and cafe CRUD functionality, as well as the POST, DELETE and PUT requests. This can be implemented using Spring Security, which checks the user's credentials and restrict access to certain parts of the application based on their role.

Cafe to Pizza has bidirectional @OneToMany relationship. This means that one cafe can have multiple pizzas, but each pizza can only belong to one cafe.

**Rest API Description(Pizza Project):**

|  |  |  |
| --- | --- | --- |
| **Operation** | **Method** | **Resource Path** |
| List all cafes | GET | /cafes |
| Add a new cafe | POST | /cafe |
| Get cafe by id with all pizza details listed | GET | /cafe/full/{id} |
| Update cafe details (identified by id) | PUT | /cafe/{id} |
| Delete cafe by id | DELETE | /cafe/{id} |
| Basic search by cafe address (should return all cafes whose name contains search term) | GET | /cafes?address={cafe address} |
| List all pizzas of specific cafe | GET | /pizzas?cafe\_id={id} |
| Add new pizza to specific cafe | POST | /pizza |
| Get specific pizza details | GET | /pizza/{id} |
| Update pizza details (by pizza id) | PUT | /pizza/{id} |
| Delete specific pizza | DELETE | /pizza/{id} |
| List all pizzas from database | GET | /pizzas |
| Basic search by pizza name (should return all pizzas whose name contains search term) | GET | /pizzas?name={pizza name} |