Changes in Project

Introduction

Use the Hotel Application that you created in your last lecture. You need to use that project and make some changes as a task for this lecture. These changes are intended for the learners to better understand the topic through repetitive practices of the concepts.

Last Lecture Project link for reference: Hotel Application Project

1. Step 1:

A. Create an entity class named **Role** with the following attributes
   1. id (Long)
   2. roleName (String)

B. Add required annotations to the attributes (i.e. @Id, etc).

C. Add Lombok annotations to auto-generate the attribute's getter, setter, and constructors.

2. Step 2 (User Entity Class)

A. Create a new **User** entity class with the following attributes.
   1. id (Long)
   2. username (String)
   3. password (String)
   4. roles (Set<Role>)

B. Add required annotation for this entity class and Lombok annotations to auto-generate the constructors, getters, and setters for the attributes.

C. Implement a mapping of a many-to-many relationship between a **User** entity and a **Role** entity.

3. Step 3 (UserRequest class)

A. Create a DTO class named **UserRequest** with the following attributes.
   1. username (String)
   2. password (String)

B. Add Lombok annotations to auto-generate the getters and setters.

4. Step 4 (UserRepository Class)

A. Create an interface named **UserRepository** in the repository package**.**

B. Add proper annotations and extend the interface with *JpaRepository*<> with the required parameters.

5. Step 5

A. Create a **UserService** class and annotate with @*Service* to mark it as a service layer.

B. Declare a class variable of *UserRepository* and make it final.

C. Define a constructor that injects an instance of the *UserRepository*.

D. Create the following methods:

- *createUser()* - This method will take a **UserRequest** object as a parameter. Inside this method, you'll create a new User entity, set its properties (e.g., username and password), map it with **UserRequest,** and save it to the database using the save() method.

- *getAllUsers()* - This method will retrieve a list of all users from the database.

6. Step 6 (UserController)

A. Create a **UserController** class by adding the required @*RestController* and @*RequestMapping* annotation.

B. Declare a class final variable of type **UserService***.*

C. Define a constructor that injects an instance of the UserService.

D. Create a *getAllUsers()* method to handle a GET request for retrieving a list of all users. Annotate it with @*GetMapping* and specify the path as “/user”. Add security authorisation to check if the user has the 'ADMIN' role and only allow it if it does.

E. Create a *registerUser()* method to handle a POST request for registering a new user. Annotate it with @*PostMapping* and specify the mapping path(“/user/register”). This method will take a **UserRequest** object as the request body and call the *createUser* method of the **UserService** to create the user.

Please find the Final code of the **Hotel Application** provided below for reference: Link.