COMP 31, Assignment

Milestone 1

Goal

The goal of this milestone is to implement the static design created for your COMP 340 project. Each project group should identify a small set of high priority use cases (one use case per group member). The combination of use cases should amount to a working subset of the overall project application. Each group member should implement one use case. The implementation should be based on the System Class and Domain Class Diagrams created in COMP 340. I.e. the implementation should include a ThymeLeaf template page, a Controller, a Service, and one or more Repositories and Entity Classes. If time allows, you should create some initial data using a CommandLineRunner so you can view the data in the H2 Console and/or on the Thymeleaf web page.

The following steps summarize the development flow taken so far in this course:

Step 1 – Configuration

Create a new Spring Boot project using the following dependencies:

Spring Web, Thymeleaf, DevTools, Lombok, Spring Data JPA, H2

Add the following code to the **application.properties** file

spring.jpa.hibernate.ddl-auto=create
spring.h2.console.enabled=true
spring.datasource.url = jdbc:h2:mem:*testdb* (You may choose a different database name).

Step 2 – Add Controllers

1. Create a **controllers** folder.
2. Create a Controller, a static **index.html** page and a Thymeleaf template page. To simplify integration with other group members' projects name the controller based on the name of the use case you are implementing.
1. Add a link to the **index** page labelled with the name of the Use Case name you are implementing.
2. Add code to the Controller so that the Thymeleaf page is displayed when the link on the **index** page is clicked.

Step 3 – Add Services

1. Create a Service class. Add methods that related to the use case. Leave the methods empty for now.

Step 4 – Add Entities and Repositories

1. Define Entity classes related to your use case. This step may involve collaboration with other group members.

2. Define Repository classes for each of the Entity classes.

Step 4 – Initialize Data

Create a new class called **Initalize** that extends **CommandLineRunner**. Annotate the class with **@Component**. Implement a run method. (This can be done automatically by hovering over the class name and using Quick Fix to add unimplemented methods).

Inject the Repository classes via an **Initialize** constructor.

Add code to the run method to create and new entities and save them to the database. E.g.

> **yourEntityRepository.save (new YourEntity(*your,initial,parameter,values*));**
> …

Step 5 – Test Database

Run the application and open the H2 console (**http://localhost:8080/h2-console**) to verify that the database is being populated.

Step 6 – Display Entity in Template Page

Add code to the Controller, Service, and Repository classes to fetch data from the H2 database. Add code to the Template page to display the data in a table.

Hand In

At the end of the lab, submit a zipped copy of your project to Blackboard. To help with marking, please rename your zip file to **yourlastname.zip**. Hand in your project even if you do not complete all the above steps.