Homework 4, Web-Based Applications And Application Servers

Network Programming, ID1212

1 Goal

- You can develop a three-tier web-based application using frameworks for all layers.

2 Grading

The grading is as follows:

0 points Not passed

1 point Passed, but **no points** for higher grades

2 points Passed, and **one point** for higher grades because an accepted solution was submitted before deadline.

3 points Passed, and **two points** for higher grades because an accepted solution including the optional higher grade task was submitted before deadline.

3 Auto-Generated Code and Copying

You must be able to explain and motivate every single part of your code. You are *not* allowed to copy entire files or classes from the example programs on the course web, even if you understand it and/or change it. However, you are allowed to write code which is very similar to the example programs on the course web. You are also allowed to use GUI builders and other tools that generate code.

4 Task, A Currency Converter With a Web-Interface

Develop a three-tier web-based application for online currency conversion. The application shall convert a specified amount from one currency to another, e.g. from SEK to EUR. If you consider solving the optional task, you are advised to read and consider it before solving the mandatory task, since the solution of the optional task might affect how you solve the mandatory task.

Requirements on Your Program

All of the following requirements must be met in order for your solution to be accepted.

- Your solution must have an acceptable layered architecture and be well designed. This means it must follow the guidelines of the lecture on architecture, and of the programming examples on the course web. Your are, however, not required to use exactly the same layers as in those examples.
- The converter must be able to convert between at least 4 different currencies.
- The client must be a web browser.
- You must use frameworks for all layers in the server, for example the same as in the video on application servers, namely Thymeleaf for the view, Spring for controller and Spring/JPA for model and integration.
- The server must handle transactions, you can for example use the `@Transactional` annotation to let Spring handle container-managed transactions.
- The conversion rates must be stored in a database.
- The user interface must be informative. The current state of the program must be clear to the user, and the user must understand what to do next.

What is NOT Required of Your Program

Below is an explanation of things that do not affect your score.

- You are free to decide how conversion rates are inserted in the database. They might for example be inserted manually before the application is started, or be inserted by the application when it is started.
- Minor misunderstandings of the functionality are allowed, as long as your program does not become notably simpler than a program implementing the intended functionality.

5 Optional Higher Grade Task

The application shall have also an admin interface, where the administrator can enter conversion rates between the different currencies. The admin interface shall also show the total number of currency conversions made by all users together, since the application was started. Both conversion rates and the counter must be stored in the database. There are the same requirements on frameworks and transactions as in the mandatory task. You do not have to implement any authentication, the admin interface might be accessed by anyone who knows the URL.