This assignment was locked 22 Nov at 23:59.
If you need help with this exercise, you must show up at the lab sessions. There will be no private
help unless you also come to the lab.

Requirements for the assignment:

The following requirements must be fulfilled to get the assignment approved:

- Deadline is Friday, 18th of November at 16:00.
- You have to work and deliver in groups of maximum 5.
- The group must deliver together on Canvas.
- Submission is compulsory.
- The code must be easily readable. Use indentation to show the structure of the code.
- Submission is done by delivering an archive (zip or tar) with two maven project directories (with code) on Canvas.
- Only tar and zip archives will be accepted. A tar archive can be compressed.
- You must be able to solve all tasks of this exercises. Answers like "It did not work" will not
  be accepted. At the lab you will get guidance, and you should be able to solve all tasks.
- You must use Spring Boot with tomcat embedded servers on both projects to be delivered.
- If your applications fail to start and run successfully with "mvn clean package spring-boot:run', the delivery will not be
  approved.
- Name the archive file with WebFramework_group_<your_group_number>.<filetype>, and replace <your_group_number> with the number of your group on Canvas and <filetype> with either zip or tar (tar.gz if compressed).
- Java 1.8 (OpenJDK) or higher (should be documented in the pom.xml)

Example:

```
<properties>
  <maven.compiler.target>1.8</maven.compiler.target>
  <maven.compiler.source>1.8</maven.compiler.source>
</properties>
```

Part one: REST WebService

Here in part one you will create a REST Web Service application for the Item (ShoppingList) application we have worked with earlier.
The application should be based on Maven and Spring Boot with the following starters:

- Spring Web (MVC)
- (Optional) "H2 database" and "Spring Data JPA"

If not choosing to use the optional "H2 database" and "Spring Data JPA", you should use the "ItemDaoMemorySingelton" class from earlier exercises.

The REST application (and eventually the H2-console) should be running on port 8299.
All responses with Item data should be in JSON format.

The following services should be included in the REST API:

- GET /items: Return a list of all the items
- POST /items: Add a new Item to the list
- GET /items/{id}: Return the items given by "id" parameter
- DELETE /items/{id}: Delete an item from the list
- PUT /items/{id}: Update an item given by the "id" parameter and get updated information from the RequestBody.

The web service calls should give appropriate statuses in the HTTP Response (for both successful and unsuccessful requests/responses).

When the application is started there should be 3 items preloaded in the Item list.

Add the following two JUnit5 tests in a test class under the directory "src/test/java": (Note! All tests should pass when running the test.)

- Test1: Test that the GET /items request is returning 3 items.
- Test2: Test that GET /items/10100 is returning a HttpStatus.NOT_FOUND.

Part two: Web Application

Here in part twp you will rewrite the web application from the lab "**web-04-web-spring-mvc-exercise.pdf**" to use the REST Web Services from part one (instead of using the ItemDaoMemorySingelton class).
The application should be based on Maven and Spring Boot with the following starters:

- Spring Web (MVC)
- Thymeleaf

(Optional) The web application should include internationalization with English and a second language of choice.

The web application should be running on port 8080.

Archive to be delivered

The archive (tar or zip) to be delivered should include two directories:

- **item-rest-web-service**: This directory should include a pom.xml and a "src" directory with the project source files. The REST Service should be startable by running "mvn clean package spring-boot:run"
- **item-web-application**: This directory should include a pom.xml and a "src" directory with the project source files. The Web application should be startable by running "mvn clean package spring-boot:run"