Faculty of Computer Science and Mathematics
Prof. Dr. Felix Schwägerl
Summer semester 2025
28 March 2025

REGENSBURG

Web Technology Project (International Computer Science)

Exercise sheet 2 — Spring Boot Backend Development

Deadline: 11 April 2025 (2 weeks)

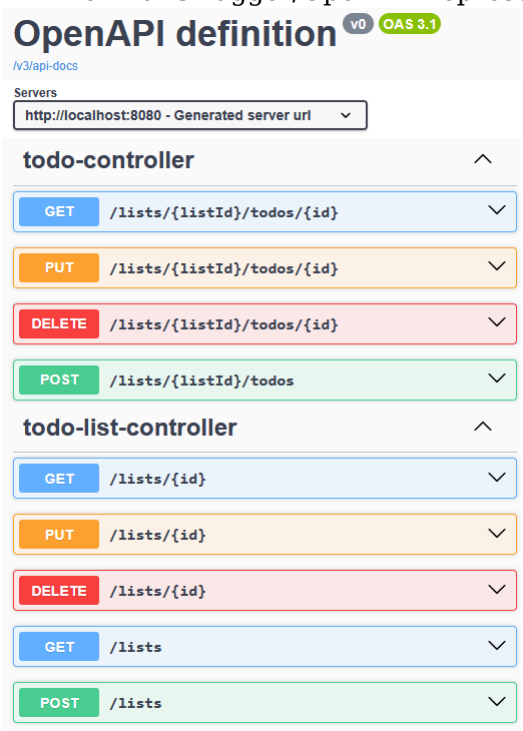
Exercise 2.1 (Specification)

You are supposed to design and implement a REST API for *to-do lists* using Spring Boot. The API should support the following operations, while not requiring any user management or authentication:

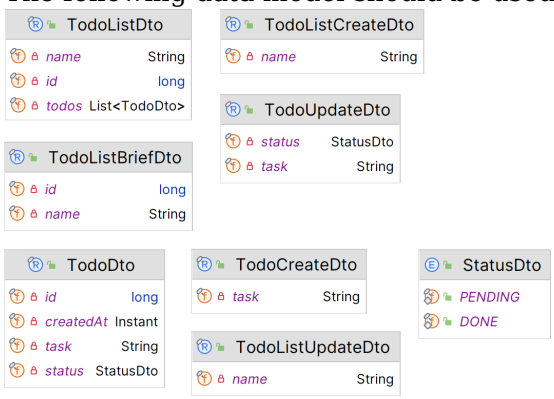
- Create a new to-do list.
- Retrieve all to-do lists in brief format (no to-dos included).
- Retrieve a specific to-do list (including its to-dos).
- Update the name of a specific to-do list.
- Delete a specific to-do list.
- Add a new to-do to a specific to-do list. A to-do has a name, a creation date, and a status (pending or done; initially pending).
- Retrieve a specific to-do.
- Update the task name and/or status of a specific to-do.
- Delete a specific to-do.

To-do lists have unique names, and the to-dos must have unique task descriptions within the same list.

The final Swagger/OpenAPI representation of the API should look as follows:

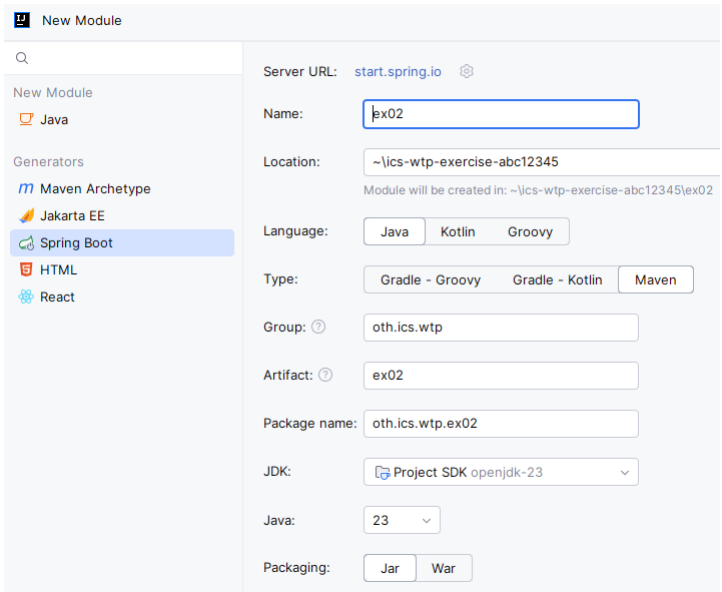


The following data model should be used for *data transfer objects* of the API:



Exercise 2.2 (Set-up)

Create a new IntelliJ *module* `ex02` as suggested by the following dialog:



Make sure you select your exercises repository as the parent directory in *Location*. Use Build tool *Maven* and the group, artifact and package names as provided in the screenshot. Java 23 is recommended. Select the following *starters* in the next dialog:

- Spring Web
- Spring Data JPA
- A suitable database driver (e.g., MariaDB or Postgres)
- H2 Database for unit tests.

Open the `pom.xml` file to add the `test` scope to the H2 dependency:

```
<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>test</scope>
</dependency>
```

Furthermore, to obtain Swagger/OpenAPI functionality, add the following dependency:

```
<dependency>
    <groupId>org.springdoc</groupId>
    <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
    <version>2.8.4</version>
</dependency>
```

To make sure the module gets correctly recognized as a Maven project, right-click on the `pom.xml` file and select *Add as Maven Project*.

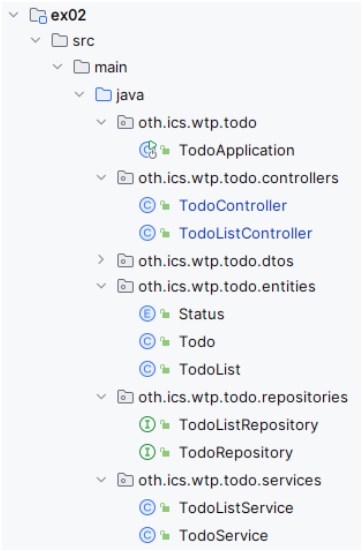
Next, to set up database connectivity, open `src/main/resources/application.properties` and add the following configuration (replacing the datasource URL with a suitable JDBC path if you use a different database deployment):

```
spring.application.name=To-Do List
spring.datasource.url=jdbc:mariadb://127.0.0.1:3306/todo?createDatabaseIfNotExist=true
spring.datasource.username=root
spring.datasource.password=asdf1234
```

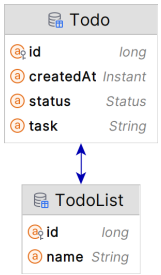
Finally, download the *JUnit test cases* provided for this exercise sheet from the E-Learning course page. Delete the existing folder `test` under `src`. Unzip the file `ex02-test.zip` and copy the root folder `test` with all contents into the existing `src` folder.

Exercise 2.3 (Implementation)

Implement the API according to its specification and the provided JUnit tests. Follow the *three-layered architecture* (controllers, services, repositories) discussed in the seminar. You should end up with a project structure resembling the following:



A data model similar to the following should be used for *entities* internally:



Exercise 2.4 (Tests)

Run the provided JUnit tests to verify the correctness of your implementation. Make sure that all tests pass successfully without requiring adaptations to test classes. If a test fails, fix the issue in your code and re-run the tests.

Once unit tests run successfully, you should end-to-end test your API using the generated Swagger UI or a REST client, e.g., Bruno.

Exercise 2.5 (Submission and next steps)

Commit and push to your exercises repository. In Exercise 5, you will build a frontend application that consumes the API you developed in this exercise sheet.