StackOverflow

1. Technologies

For the Back-End of this assignment I have chosen to use Spring-Boot 3.1.4. Along Spring-Boot I have used the following dependencies:

* spring-boot-starter-data-jpa
* spring-boot-starter-security
* spring-boot-starter-web
* mysql-connector-j
* spring-boot-starter-jdbc
* spring-boot-starter-test
* spring-security-test
* jjwt-api
* jjwt-impl
* jjwt-jackson

For login and security I chose to use JWT token because I have worked with it in the past and provide better security than simple authentication.

For the database I have used MySQL Server 8.0.28

2. OOP and Architecture

I have decided to use a layered architecture because, presentation, application processing and data management functions are physically separated. In my case I decided to use three-tier architecture:

* Controller
* Service
* Repository

The controller layer handles all the requests to the server. It is responsible to deny unauthenticated or unauthorized users access to resources and to ensure that the request format complies with some rules.

The service layer does all the preprocessing needed before querying the database. Mathematical operations for example can be done here.

The repository layer is where all the queries to the database are performed. Here we ask and receive answers. The answers are then sent back to the service layer and further to the controller which in the end sends them to the client.

For this project I needed the following entities:

* Answer
* AuthRequest
* Question
* QuestionAnswers
* User
* UserInfoDetails

Interesting OOP concepts that I have implemented would be the use of extended interfaces, constructor overloading and overriding functions.

3. Features and use cases

The user firstly can create an account (/user/register).

After the account is created, the user can log in (/user/login) using the credentials and he will receive a JWT Token available for 1 hour.

Only after the user is logged in, he can access the following endpoints by submitting the token preceded by “Bearer “ in the Authorization header of the HTTP Request. The endpoints accept either GET or POST HTTP Requests depending on the type of action.

The user is able to ask questions (/user/postQuestion). He can also list all of his questions (/user/getQuestionsOfUser) or search through all questions by more filters (/user/searchQuestions). The user can also edit (/user/updateQuestion) or delete (/user/deleteQuestion) his own questions. There can be no filter selected, in this way he will be prompted with all questions, or more filters and in this way he will receive questions which pass all the filters. All questions are received descending by their posted date, so newer questions will be first.





