StackOverflow

Student: Bogdan Tudor-Alexandru

Group: 30434

Teaching Assistant: Bindea Bogdan

Introduction

For this project, I want to implement a web application for helping all people around the world that are working in the IT industry, by asking questions (e.g. a problem they encountered when developing their application), or/and by helping other by answering their questions. Questions and Answers can be upvoted/downvoted as a form of feedback for their usefulness.

Technologies

The technologies used for this application will be:

- Java with Spring for the Backend of the application
- React and Angular for the Frontend
- MySql for the DataBase

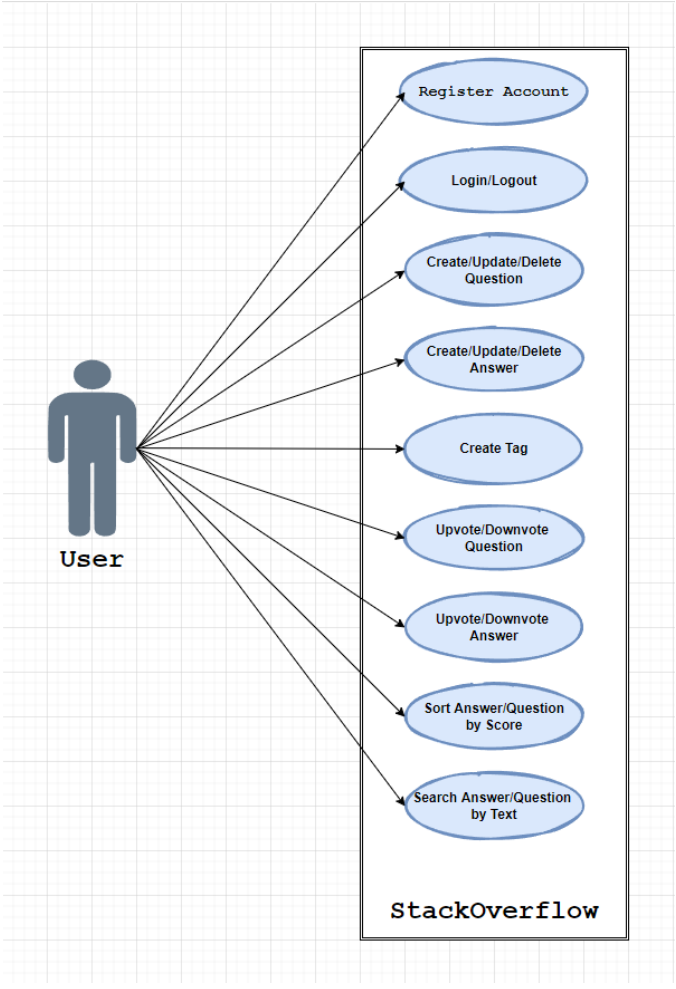
Use Case

Users will be able to:

- **Account**:
  - Register a personal account to use the platform
  - Delete the personal account they registered
  - Login/Logout into the account with the credentials they specified when registering the account
- **Question**
  - Create a question in the application

    - Edit/Delete a previously asked question
    - Search a question by text
    - Sort questions by scores (votes) or by date
- Answer
    - Create an answer in the application
    - Edit/Delete a previously typed answer
    - Search an answer by text
    - Sort answer by scores (votes) or by date

Use Case Diagram

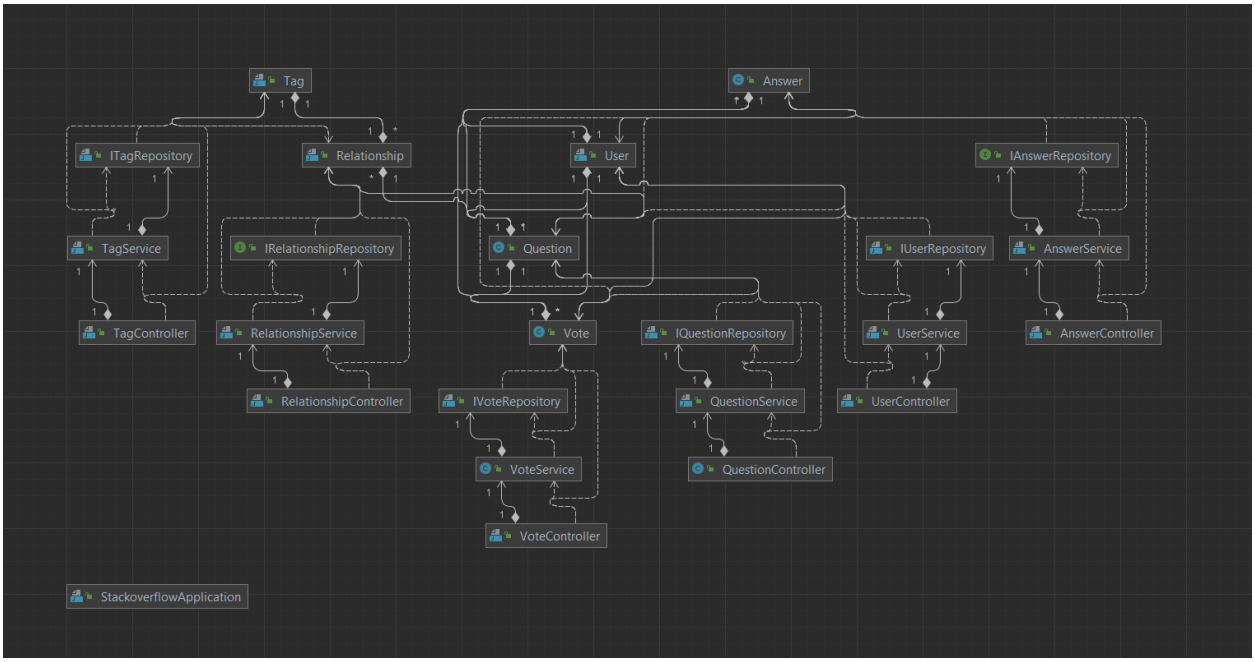


Layered Architecture: Backend

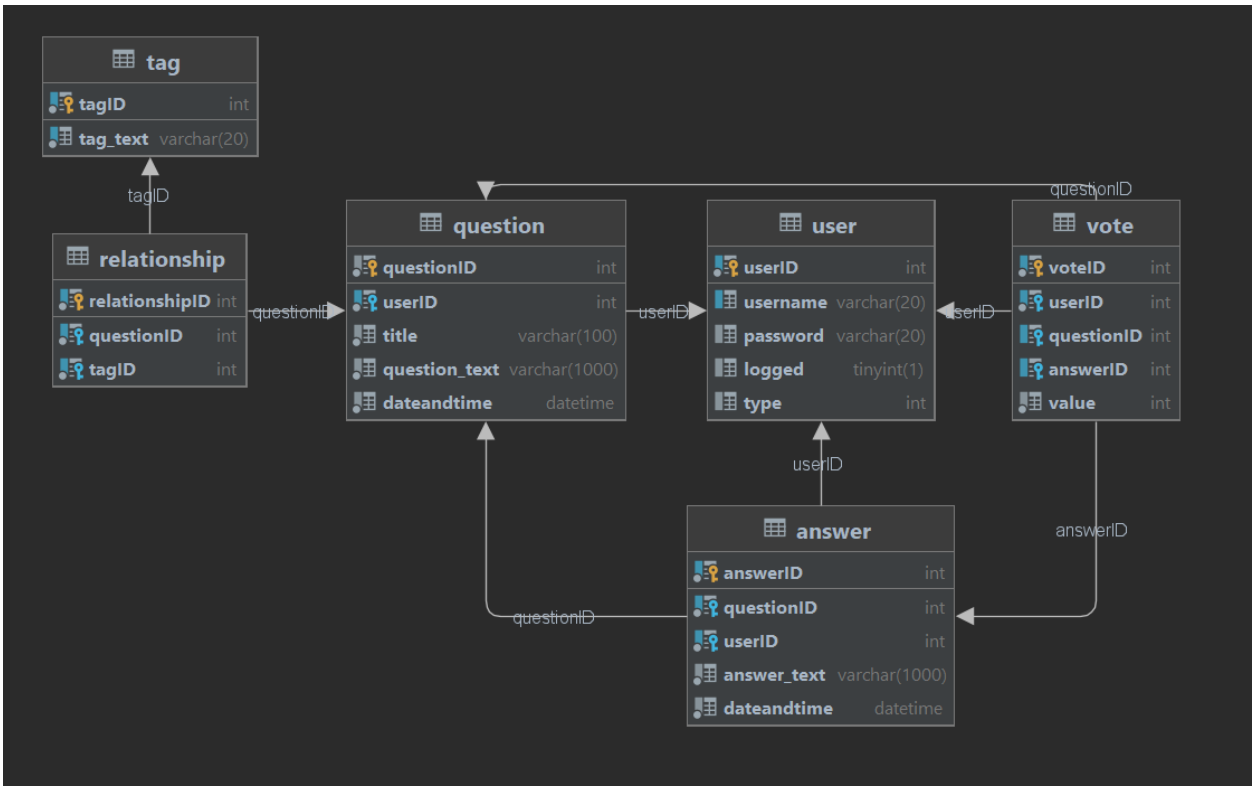
The program will use a layered architecture composed of:

- **Model Layer:** which will contain the entities from the databases and everything related to their representation
- **Repository Layer:** which will contain interfaces used for the implementation of the services
- **Service Layer:** which will contain all the CRUD (Create, Read, Update, Delete) operations for each of the models in the application
- **Controller Layer:** which will contain all the endpoints used in the application implemented by using the services
- **DataBase:** which will store all the Data necessary to the application.

Class Diagram



Database Diagram

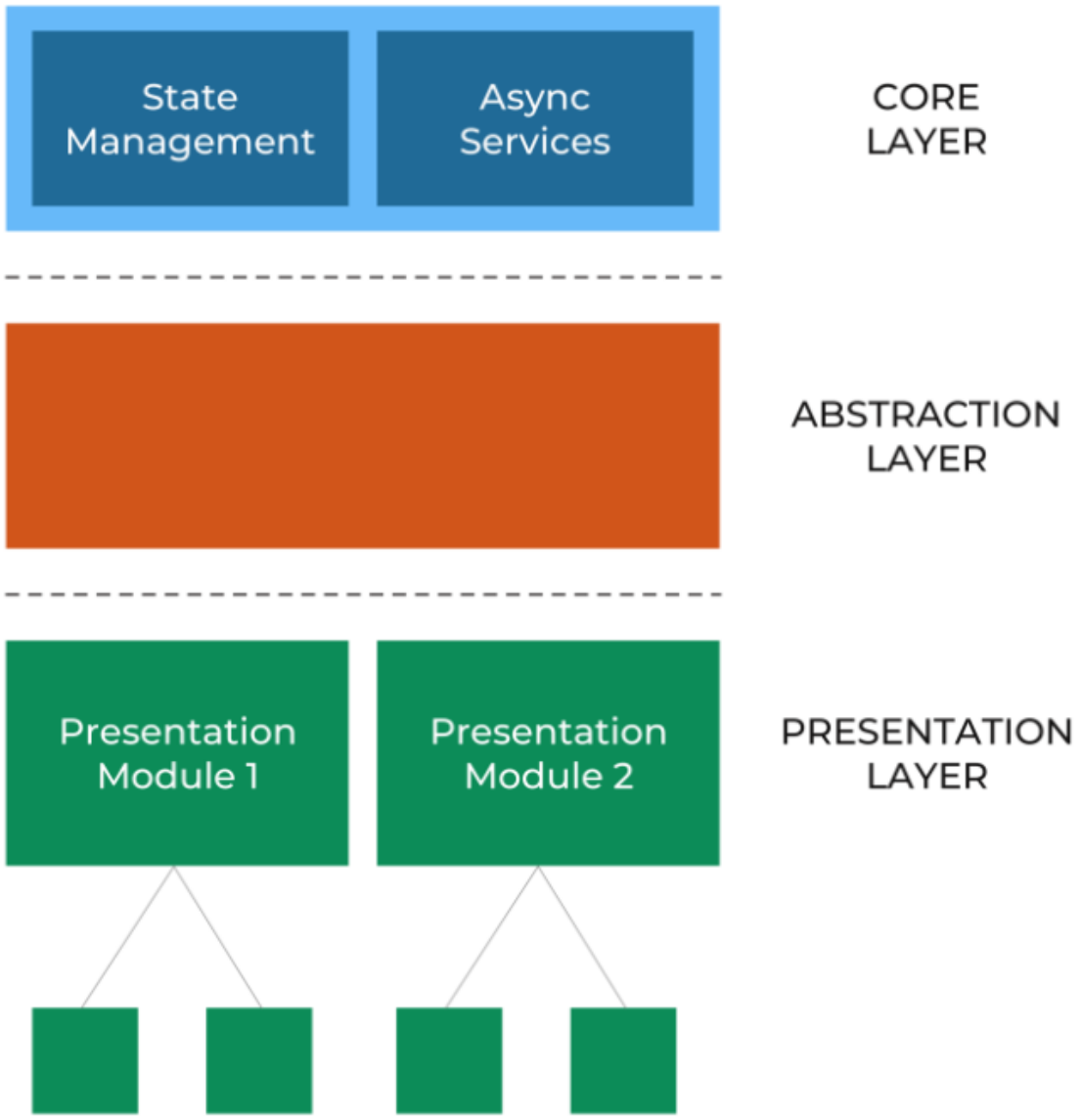


Layered Architecture: Frontend

The program will use a layered architecture composed of:

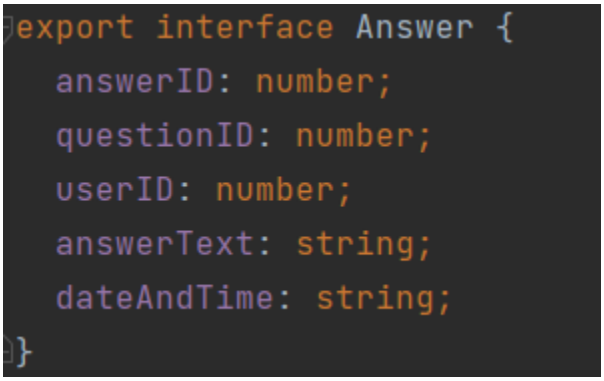
- **Presentation Layer (Components)—** This layer is mainly responsible for the design and user interface. Here we are defining all our angular components. It is displaying all templates and handling user events. Presentation layer also responsible for Unidirectional data flow.
- **Abstraction layer —** The abstraction layer decouples the presentation layer from the core layer. This layer will behave as a mediator and the facade for the presentation layer. It means that it will expose an API and coordinate the communication between presentation layer components and the core of the application. For example, here we call services to have data communication between template and core layer.
- **Core layer (Models and Services) —** This layer is the innermost layer of an application, though all outside communication happens here. In this layer, one can place all state management, core modules, and services. The core layer is responsible for encapsulating the state and behavior of an application. In angular, we have services to write all API calls which contain @Injectable to inject APIs to

communicate with outside. All services lie in the core layer. In this layer, we could also place any validators, mappers, or more advanced use-cases that require manipulating many slices of our UI state.



Entity Mapping

For an easier communication with the backend data (sending and receiving) I mapped the entities from the backend by creating some interfaces which represent the backend classes/ For example for the **Answer** class from backend, I implements the **Answer** Interface in the frontend:



Endpoints

The communication of the application with the DataBase is done through endpoints. To access a given table from the DataBase, the endpoints will be: **“/table/operation”**, and in case the operation requires some parameters they can be given in the endpoint. For example, to get the Answer with id =5 from the DataBase, the accessed endpoint will be: **“/answer/getAnswer?id=5”.** These endpoints can be found in the service corresponding to each model. For example, for the Answer model I have the Answer Service which implements the operations:

- `const baseUrl = 'http://localhost:8080/answer'` This will be the starting URL for the Answer Class, to which I will append the operation I want to be performed:
- `getAll()` Which will access the `` `${baseUrl}/getAllAnswers` `` endpoint.
- `get(id: any)` Which will access the `` `${baseUrl}/getAnswer?id=${id}` `` endpoint and where an additional parameter is given: the ID of the wanted Answer to be retrieved.
- `create(data: any):` Which will access the `` `${baseUrl}/createAnswer` `` endpoint and which will send along the Object to be created in the DataBase.
- `update(data: any):` Which will access the `` `${baseUrl}/updateAnswer` `` endpoint and which will send along the Object to be updated in the DataBase.

- `delete(data: any):` Which will access the `` `${baseUrl}/deleteAnswer?id=${id}` `` endpoint and which will send along the ID of the Object to be deleted in the DataBase.

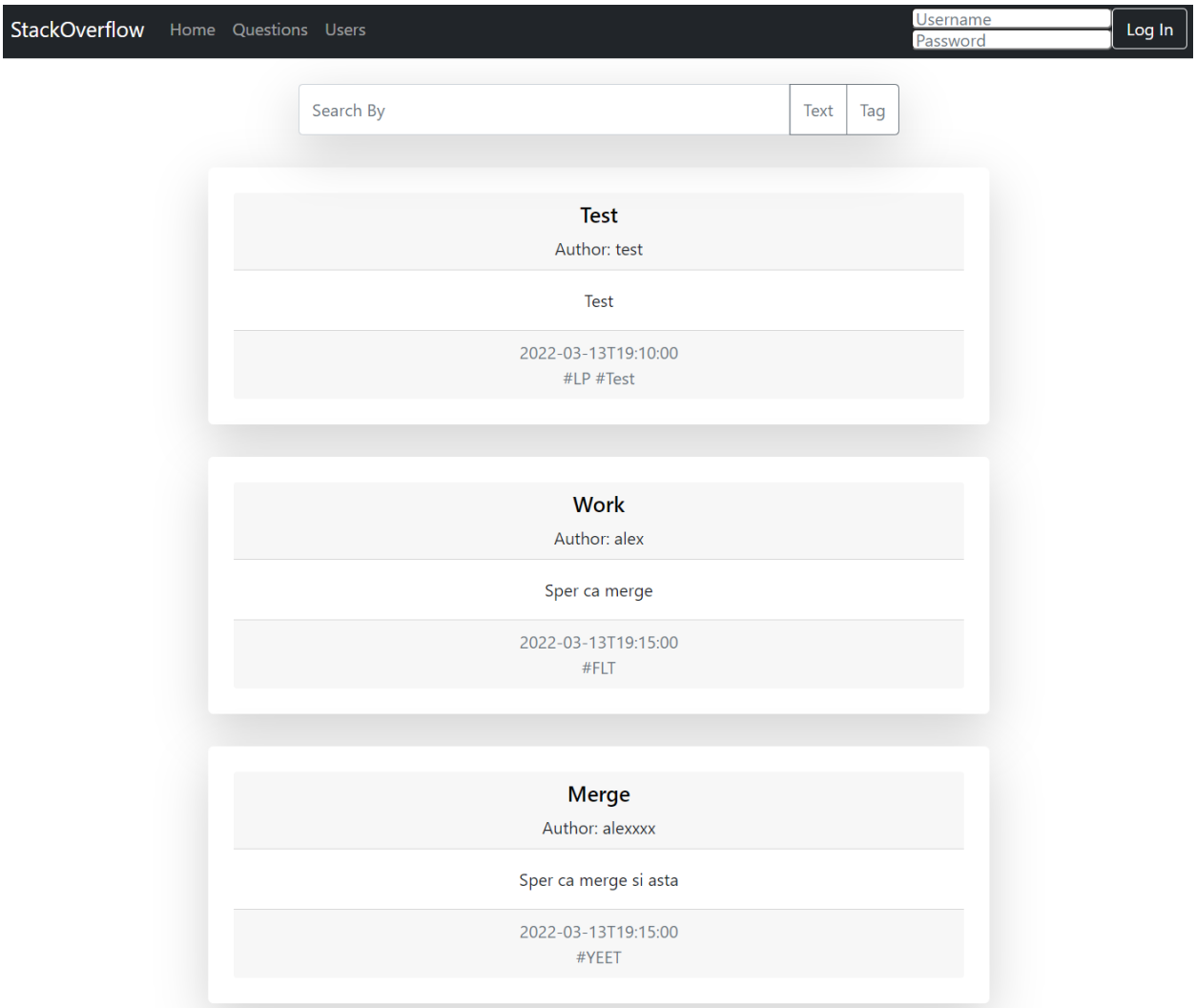
Communication between the Frontend and the Backend

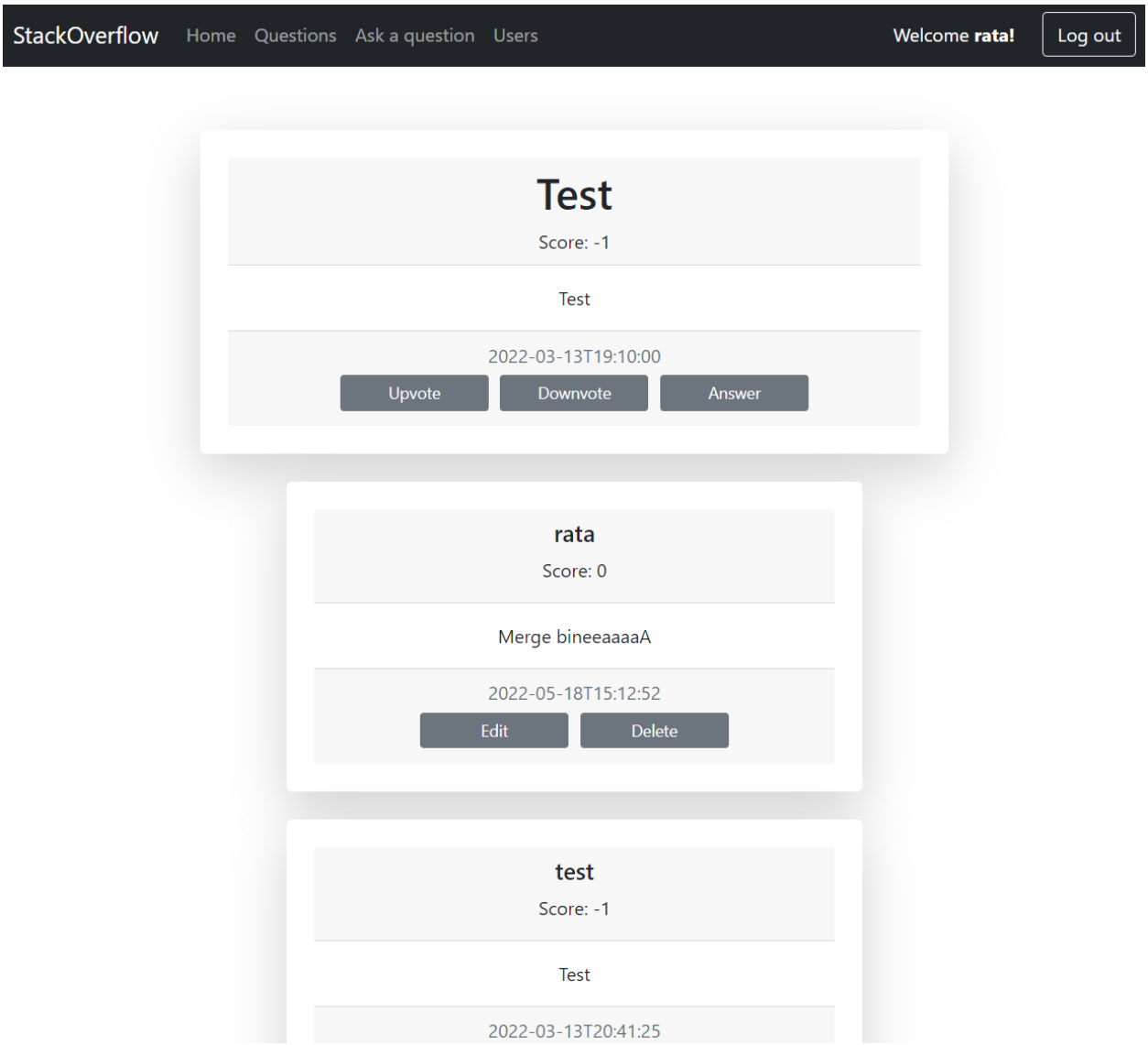
The communication between the Frontend and the Backend is done through the asynchronous functions implemented in the corresponding service using Observable.

User Interface:

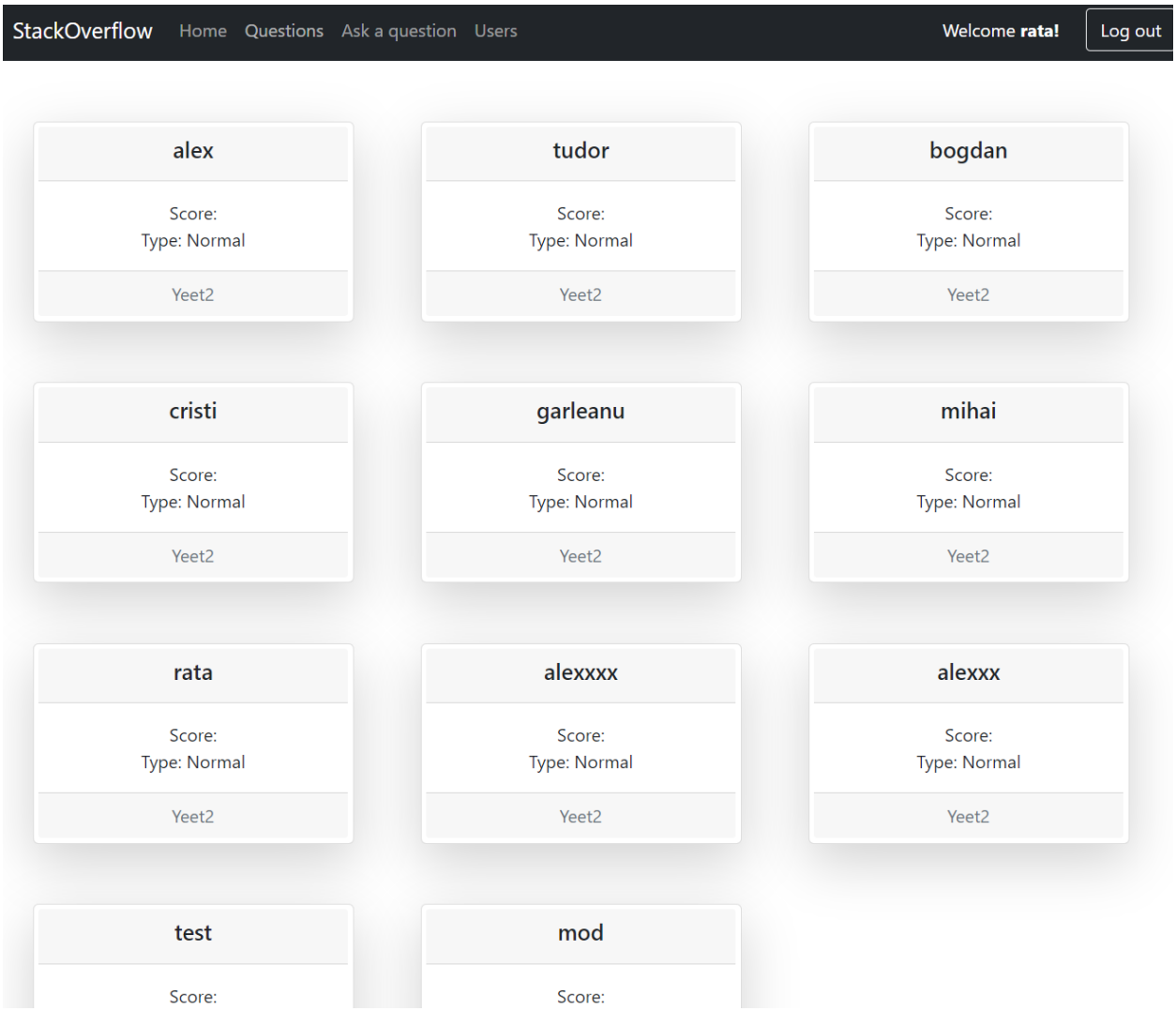
Home page:

Questions Page:

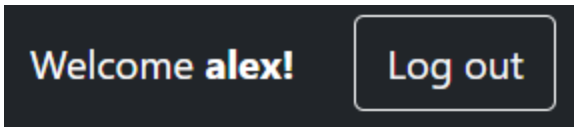

StackOverflow
Home
Questions
Users
Username
Password
Log In
Search By
Text
Tag
Test
Author: test
Test
2022-03-13T19:10:00
#LP #Test
Work
Author: alex
Sper ca merge
2022-03-13T19:15:00
#FLT
Merge
Author: alexxxx
Sper ca merge si asta
2022-03-13T19:15:00
#YEET


StackOverflow
Home
Questions
Ask a question
Users
Welcome rata!
Log out
Test
Score: -1
Test
2022-03-13T19:10:00
Upvote
Downvote
Answer
rata
Score: 0
Merge bineeaaaaA
2022-05-18T15:12:52
Edit
Delete
test
Score: -1
Test
2022-03-13T20:41:25

Users Page:



Login/Account



Bibliography

- Baeldung: Working with Relationships in Spring Data REST | Baeldung
- Lab Guidelines + Lectures
- Javapoint: Spring Boot CRUD Operations - javatpoint

- Pluralsight: Angular Fundamentals | Pluralsight
- Bootstrap: Bootstrap · The most popular HTML, CSS, and JS library in the world. (getbootstrap.com)
- Observable: Angular 13 Observables: How to use Observables in Angular (appdividend.com)