SD Laboratory Assignment –

Stack Overflow

Laboratory teacher: Eng. Bindea Bogdan Nicusor

Student: Petruta George-Calin

Group: 30434

Contents

[1. Introduction 3](#_Toc100146725)

[2. Technical Stack 4](#_Toc100146726)

[2.1 Programming language 4](#_Toc100146727)

[2.2 Frontend framework 4](#_Toc100146728)

[2.3 Data storage and querying 4](#_Toc100146729)

[3. Use cases 4](#_Toc100146730)

[3.1 Detailed use case: 4](#_Toc100146731)

[3.2 Log in: 5](#_Toc100146732)

[3.3 See question: 6](#_Toc100146733)

[3.4 Add question: 7](#_Toc100146734)

[3.5 Add answer: 7](#_Toc100146735)

[3.6 Like/dislike answer/question: 8](#_Toc100146736)

[4. Architecture 8](#_Toc100146737)

[5. Testing 9](#_Toc100146738)

[6. Bibliography 9](#_Toc100146739)

# Introduction

The application to be designed is a very simple copy of Stack Overflow. It will permit the user to enter questions with tags, like comment and question, see and search questions based on their title and/or tags and comment on questions.

The user is not able to do anything before logging in besides, of course, sign-up.

The users can like each other’s contributions, but not their own. They can also edit or delete their own material on the application when they want.

Beside the feature of adding new content on the web application, there will be a system which will rank the users depending on their contributions to the site.

Their score will be showed on every comment, and it will be updated when they receive reactions from others and depending on that, the score will increase or decrease.

Also, beside the normal user, there will be an/some admins which can ban/unban users if there is a need for that and delete/edit materials that are found to be against the rules of the application.

On short, the application will have 2 types of users: Admin and User and each of them will have the following available features:

* Admin:
  + Add users
  + Delete/Update comments/questions
  + Ban/Unban users
* User:
  + Add/edit/delete question
  + Add/edit/delete answer
  + Like/dislike content
  + Create account
  + Search questions

# Technical Stack

## Programming language

The project will be done using as a programming language **Javascript (React library).**

**Java version used: 17**

## Frontend framework

The UI will be done using the **React** library.

**Visual studio code** is used as the text editor for the development of the UI.

## Data storage and querying

The application used for storing the database and executing queries is the Mysql Workbench.

# Use cases

## Detailed use case:

**User logs in, comments on a question and like another answer:**

The user accesses the application and logs in.

After he successfully logs in, he can then browse through the questions and select one of them by clicking on the button to the right.

A new page will be displayed, showing the information about the question: title, author, vote count, tags and content.

Below this information added by the author, the user can find a comment section.

There, the user can like other’s user answer. Below the comment section he will find a button to add a new comment. After he enters the comment and press the button, the answer will be shown on the question page.

## Log in:

When starting the application, the user will see a login button which when pressed will redirect the user towards a page where he can log in or create a new account.

Graphical user interface, application

Description automatically generated

If the user logs in and the account if banned, he will receive a page saying that his account was banned.

Text

Description automatically generated

If the password is wrong or the account does not exist, another error page will be displayed.

Graphical user interface, text, application

Description automatically generated

Table

Description automatically generated with low confidenceIf the password and the account is correct, then the user is redirected to the main page where he will see a list of questions.

## See question:

Graphical user interface, text, application, email

Description automatically generatedWhen the user’s presses the button “See question” he will be redirected towards a new page containing information about that question such as: answer, likes, author.

## Add question:

Graphical user interface, text, application, email

Description automatically generatedWhen the user’s presses the button “Add question” he will be redirected towards a new page containing information for a new question: title, text and tags.

After adding the information for the question and pressing the “Save” button, a new question will be added and the user will be redirected towards the question list page.

## Add answer:

Graphical user interface, text, application

Description automatically generatedWhen the user’s is in a question page, he has the possibility of adding a new answer by pressing the button “Add answer”. This will redirect him towards a new page where he can add the answer and then he will be redirected back to the question page, where he will then see his newly added comment.

## Like/dislike answer/question:

When the user’s is in a question page, he has the possibility of liking or disliking every comment/question that is not his. After he likes/dislikes that answer/question the button will become unavailable. Also, the score for the users will be modified depending on the type of the content and the type of the like.

Graphical user interface, text, application

Description automatically generated

Fig 3.6.1 Image before adding a like to an already disliked question

Graphical user interface, text, application, chat or text message

Description automatically generated

Fig 3.6.2 Image after adding a like to an already disliked question

# Architecture

The architecture used for the front-end is called Component Architecture and Unidirectional Architecture. This architecture depends on components when rendering pages, breaking up the rendered page in more, smaller, specific components.

For example, when displaying a list of elements, first we create a component for that given element.

Diagram, timeline

Description automatically generatedThe Unidirectional Architecture is used for directing the data through the application using states.

Fig 4.1 Unidirectional Data Flow Architecture

In react, this architecture is implemented through the usage of components’ props and state. When an event takes place, an action is done and the state is modified, resulting in a new rendering of the page using the new state.

# Testing

It was done mostly manually, going through each feature and see the results and then modifying the code if needed.

The manual testing was done by verifying the data displayed to the user and then verifying if the data was correctly updated in the database.

There are no real tests written, just some trials for the main component.

# Bibliography

1. <https://suas.readme.io/docs/why-unidirectional-architectures>
2. <https://www.postman.com/>
3. <https://azeynalli1990.medium.com/software-architecture-patterns-for-front-end-development-9e43e43cdfb3>
4. <https://testing-library.com/docs/react-testing-library/intro/>