Joonatan Kallio 4.3.2023

CT30A3204 Advanced Web Applications – Final Project documentation

What was done:

I created front and backend to this project which is a code forum. Users can register and login to post problems and comment other people's posts to help. Users can also see other people's profile and bio. This was done using React, NodeJS with Express and MongoDB. More information about the technologies is listed down below.

Installation and user manual:

This project uses node version 16 and for some reason it did not start with node 12 so please use node version 16 for running this. To change the version type "nvm use 16" to the terminal/cmd.

First clone the project from the GitHub repository.

After that to run and install this project, you need to have NodeJS and MongoDB installed on your computer. Go into the root directory of the project and type "npm run install".

Then the first thing you should do is run the server with these commands:

in Windows:

"SET NODE_ENV=develpoment& npm run dev:server"

or with mac or Linux:

"NODE_ENV=development npm run dev:server"

After the server is up and running then open another terminal tab and type:

"npm run dev:client"

Now you should have this project up and running with the database connected. (Database in this address "mongodb://localhost:27017/projectdb")

To run the cypress tests, keep the servers open and type in to a new terminal in the root folder:

"npm run test"

Keep in mind that after running the tests once, you should remove the user, posts and comments made by the test to be able to run tests successfully again.

Every command is typed in the root folder of the project. The project README.md also includes these commands.

The technologies I used for this project:

- React-draft-wysiwyg and draftjs
    - This allowed me to use the good editor with different stylings which made the user experience better.
- Express
    - Server API was created with express.
- NodeJS
    - The whole server is made with NodeJS using the Express framework.
- React
    - Front end was created with React which allowed me to use Routers and MUI for example. React also allowed me to reuse some components in the project.
- React Router
    - React router was chosen because it helped me to show for example posts with their ID in the url "/post/postid".
- MUI
    - I chose MUI for its simplicity and its compatibility with react. MUI was used to style the elements and to get responsive website.
- Dompurify to sanitize the user's posts
    - Dompurify was used because it would be dangerous to otherwise put straight HTML code in to innerHTML but with dompurify it checks it for malicious code and purifies it.
- Luxon datetime
    - Luxon was used to get the datetime object from the database and format it to the way it is shown on the website.
- Passport and Passport-JWT packages and also bcrypt.js
    - These are used for the authentication process which was made with JWT and storing the token in the local storage.
- MongoDB and Mongoose
    - I used MongoDB Database for this project and accessed it in the server side with Mongoose. MongoDB was easy choice since it is easy to create good databases alongside with NodeJS.

Teachers points and features:

25 – minimum

- Application uses NodeJS and Express
- Application uses MongoDB with Mongoose
- Authentication is done with passport and JWT token. Users can login and register and not authenticated users cannot modify database in any way.
- Authenticated users can:
    - Post new problems, comment posts and edit their own comments and posts
- Non-authenticated users can:
    - View other people's posts and comments and click usernames to see the user profiles.
- There is the main page which lists posts, by clicking one post you get the post overview page where the post is shown and comments under it. By clicking usernames in the post overview page, the user can see others or their own profile page where username, register date posts and bio is listed.
- The page is responsive and works with mobile or desktop.

4 – Users can edit own comments and posts

5 – Using React framework

2 – Profile page can be shown by clicking the username in the post or comment in the post overview page and it shows bio, register date and username

2 – Posts and comments show timestamps when created or when edited

5 – Over 10 tests done with cypress

My own point proposals:

1 – Using WYSIWYG editor when posting a new post or editing old post, which allows the user to format their text better and change font size or bold it etc. There is also a code formatting tool with WYSIWYG which is highlighted in the post with gray background.

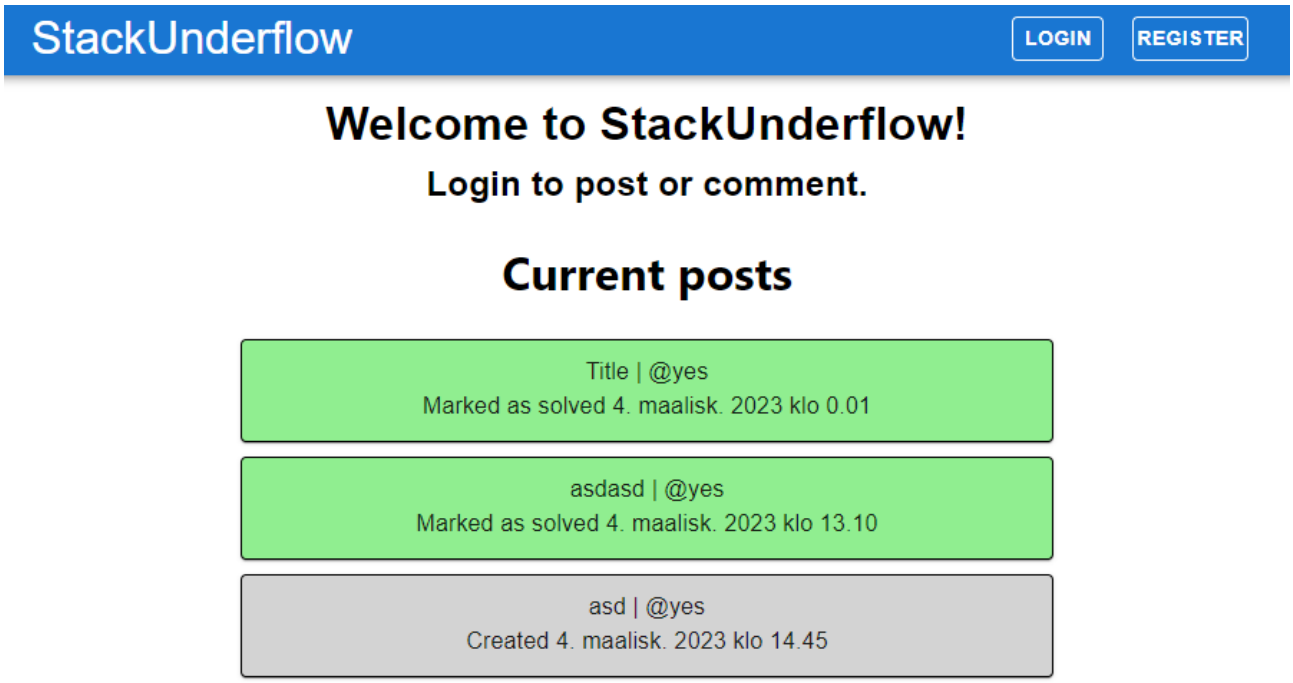
1 – User profile page lists also all the posts done by that user and by clicking the posts there it takes you to the correct post page (this extends the teacher's profile page feature).

2 – User can mark their own posts "solved" which is indicated with green color scheme so other users know they don't need help anymore. This also disables the commenting to that specific post so there can't be new comments posted, this way the solution can be found within the comments easily.

The overall point amount I am proposing from this project is 47 points.

Some pictures of the project

Post overview page in the mobile phone version.



Home page in the desktop version.