Advanced Web Applications

Project documentation:

Leevi Kämäräinen 0568396

Technologies used:

For the backend, I used Express as the framework, and utilized MongoDB as my database. MongoDB was used with mongoose. On MongoDB I had two schema, one for users and one for code posts.

For my frontend, I used React as frontend framework. In addition to React, I used MUI as my material UI which helped to make cleaner looking components and as well as helped with keeping the app to have responsive design.

For authentication I used JSON web token, and I stored the token inside browsers local storage so the sessions would be saved. This might not be such a smart thing to do for a real project, but I decided that it is suitable for this project.

Users’ passwords were crypted using Bcrypt.js, and the hashed passwords were saved to the database.

Installation process:

When the project is imported from GitHub, you need to do the following steps to install node modules. The project is able to run only on dev environment at the moment.

1. Change directory to server (cd server), and then run the command “npm install” to install packages. After you have installed the packages, you must create a new file in the server folder called .env. Then you need to write on that file the following “SECRET=<secret you want to use>”. The secrets value can be of your choosing. This is because .gitignore file ignores committing the .env file to GitHub.
2. Change directory to client folder (cd client), and then run the command “npm install” to install packages
3. After the packages has been installed, then change the folder back to main project folder. Now we need two instances of PowerShell open.
4. On the first PowerShell instance you need to first open the main project directory on it. Then you can start the backend server with the following command: “$env:NODE\_ENV="production"; npm run dev:server”

Please make sure that if you have MongoDB database named testdb that the document for users is empty before starting the backend server for first time. If the document is not empty, then the code can not initialize an admin account on the database.

1. On the second PowerShell instance you need to first open the main project directory on it. Then you can start the frontend React server with the following command: “npm run dev:client”
2. Now you can start using the application. The login details for admin account are

Username: “admin”, password: “admin1234567”.

USER MANUAL:

Register:

* You can register from the top right corner (button register). Username must be over 3 characters long, and password must have minimum length of 8, atleast 1 lowercase, uppercase, number and special symbol. Successful registering will redirect you to login page, unsuccessful register will not do anything.

Login:

* You can access login from top right corner (button login). Successful login will redirect you to main page, unsuccessful login will not do anything.

Adding a post:

* When you have logged in, you can add new post by writing on the pink box on bottom of the main page, and clicking submit. The new post should appear immediately on the page.

Adding a comment:

* On an existing post, you can add a comment by writing on the bottom of the box (below: “Leave a comment”) and pressing the arrow button.

Voting:

* You can leave a single vote for existing posts/comments. Changing the vote will change your previous vote to the new one. Logged in users will save the previous votes on the posts/comments, so you will always know what you have previously voted. Clicking a vote you have already given, will retract your vote.

Editing posts:

* When you are logged in, the text boxes of your comments/posts will change to editable ones. When you have edited your text, pressing enter will save the text and update the last edited timestamp. Refreshing the page in case of new comments/posts should help you being able to edit any new comments.

Admin account:

* Admin account has the following login details: username: “admin”, password:

“admin1234567”. When you are logged in as admin, you can see trashcan icons on posts/comments, and by clicking those you can delete those posts/comments. You can also edit any of the posts/comments there are on the page.

Logging out:

* Pressing the logout button will log out you from the current account, and redirect you to main page.

Pages:

* When there are more than 10 posts, you can utilize the pagination on bottom of the posts. There can only be 10 posts per page visible.

JWT-token expiring:

* When you try to do some action on expired JWT token, it will automatically log you out of your account and remove the token from local storage.

Features done:

|  |  |
| --- | --- |
| **FEATURE** | **POINTS** |
| Basic features | 25 |
| Utilizing React framework | 5 |
| Users can edit their own comments / posts | 4 |
| Last edited timestamp | 2 |
| Only 10 posts per page visible | 2 |
| Users being able to upvote/downvote posts and comments | 3 |
| Admin account (able to edit/delete all posts) | 3 |
| Utilizing MUI for all components (own) | 2 |
| Comments are dropdown on the main page (own) | 2 |
|  |  |

TOTAL POINTS : 48

Thoughts on the assignment, and what I would have liked to do better:

Overall I enjoyed doing the practical assignment very much. It was fun to getting to know working with React with JavaScript. However I noticed that it was hard to keep the code clean, and everything in the project in organized manner. I would have liked to keep the project more clean, using MUIs components managed to make the code very chaotic on some points, but I did not have much time on cleaning up the code so it might not now always be easy to follow.

I would’ve liked to add some features which would have enhanced user experience. For example adding messages to unsuccessful logins/registers, when the JWT token expires (it automatically logs you out now). Also adding a functionality that new comments are automatically being able to be edited (without refreshing) page is something that I would have liked to do.

It was hard sometimes to keep the whole project in order, since there seems to be so many moving parts. What I would have done differently now after finishing the project, is that I would have created a more detailed plan (schemas for databases especially) on what I will exactly do on the project. I had to make some changes on the MongoDB schemas, which essentially made me change a lot on the backend and the frontend code.