Web Applications Project, Summer 2022
Marianne Seppänen

Project “Fancy Forum” Documentation

Introduction

The project is a full-stack web application, that allows the user to register, login, and post and comment messages. Users that are not logged in can see the posts and their comments. A registered user can write their own posts and comments. This documentation presents the technology used in the application, its features, and a manual for the user.

Technology choices

Backend

The backend is built with Node.js and express. The backend handles different POST and GET calls from the frontend and saves and fetches queries from the database. The routes for the calls are in the server/routes/users.js file. Backend also handles password encryption with bcrypt and authentication with jwt-tokens. The tokens are only saved in the frontend as variables, which means that after a user updates the page, their session expires and they must log in again. The application uses MongoDB database with Mongoose. The database runs at *mongodb://localhost:27017/projectdb* and it has two collections, users and posts. The models for the collections are in the server/models folder.

Frontend

The frontend is built with React. There are seven different components, that together build the different parts of the frontend, including the register and login forms, comments, and posts. The components are in the client/src/components folder. To style the frontend, components from “MUI: The react component library” have been used. For responsive design, media queries are made for phones (max 480px) and tablets (max 700px). These are defined in the client/src/App.css file. As the application is quite simple, these media queries only change the width of the application to 100% to avoid the elements from being too wide for the screen.

Installation Guidelines

For the project to run properly, several libraries need to be installed with npm:

- express
- react
- mongoose
- bcryptjs
- dotenv
- express-validator
- jsonwebtoken
- multer

- @mui/material @emotion/react @emotion/styled

The project can be started with the command *npm run dev*
which starts both the client and server.

User Manual

The main page of the application shows a welcome-message, register and login forms and the posts. To register, the user must type in a username and a password that are both more than 5 characters. After that, the user can login with the same username and password. There is no way to fetch lost passwords, so the user must remember these. If the login is successful, the register/login forms should disappear and a personalized welcome message should appear, with a text-field for writing new posts. To write a new post, user must type in their post and press Post-button. The new post appears at the end of the post list. Under each post, there is a Comment-button that opens all the comments to a post, and a comment box if the user is logged in. To write a comment, the user can type in the textbox and press send, after which the comment appears on the screen under the other comments. The user will be logged out automatically when they update the page.

Requirements

Basic Requirements – 25 points

- Implementation of backend with Node.js
- Utilization of database
- Authentication
    - Users have to have an option to register and login
    - Only authenticated users can post or comment
- Features
    - Authenticated users can post new code snippets and comment on existing posts
    - Non-authenticated users can see posts and comments
    - There is some page listing all the post, after opening one post, comments are also listed
- Responsive design
    - The app needs to be usable with mobile devices and desktop browsers
- Documentation
    - There needs to be documentation describing the technology choices, installation guidelines and user manual

Possible Features – 5 points

- Utilization of a frontside framework, such as React (5p)

All project points: 30 points