Advanced Web Applications Course Project

By Jeremias Moberg

Description of the program

The program can be used to share messages including code snippets. If the user hasn't logged in, he or she can only view the message threads but not comment them or create new ones. When threads or new comments are created, they are saved on database. The front-end gets all the threads from the database and each thread shows the owner of the thread, title, and when was it created. The threads can be viewed by pressing the "Open" link. When the link is pressed, the thread will be opened below the list of threads stating the title and below that the thread's content. Each message in a thread shows who sent the message, what is the content of the message, and when was it sent.

If the user isn't logged in, on top of the page there is a login form and button for switching between registration form and login form. If the user is logged in, instead there is a welcoming message for the user and logout button. Now below the thread listing there is also a button for creating new threads which when pressed, opens a form which requires title and text. Creating a thread can be cancelled too. If the logged in user is viewing a thread, below all the messages are a button for adding new comments. When pressed, it simply opens a text field for sending a comment, which can also be cancelled.

Practical implementation

I started the project by making a full-stack application based on the lecture about the topic. Version control was achieved with Git and GitHub, while the project was developed in VS Code. I decided to use JavaScript Express as back-end language and MongoDB and Mongoose as the database for the project because they were used in the assignments on the course and I'm most familiar with them. As a front-end, I decided to use React over plain JavaScript since I wanted to learn more about React and I find it more interesting to use. As authentication I used JWT with local storage. The design is simple but usable on both desktop and mobile devices although I didn't use any libraries for it. The app looks pretty much the same on both platforms. On desktop, there is just empty space on the sides. The front-end works on http://localhost:3000, the back-end on http://localhost:5000, and the MongoDB on mongodb://localhost:27017/db.

I had some problems with the React part because I'm quite new to the framework but with the help of ChatGPT I was able to debug my code to make the app usable. The app isn't quite as I'd like it to be, for example, I wasn't able to show the thread the user is commenting about although I tried several ways to do it. Regardless, I'm satisfied of the project I was able to produce. There aren't many comments withing the code since I tried making the code easy to understand by making the variables self-explanatory.

Features and points

Feature	Points
Basic features with well written documentation	25
Utilization of React as frontside framework	5
Last edited (or rather the date of the post/comments since editing is not supported) timestamp is stored and shown with posts/comments	2
Total	32