Akseli Aula
0545267
12/2021

Web applications course – Project documentation

Technology choices:

This MERN project mainly uses the technologies used along the course. The files are stored in a MongoDb. Server side is built using Node.js and Express. Client side is built using React. I decided to use React as a client-side framework, even though I had previous experiences with angular.

In addition to those, the following node modules were used among others:

- Bcryptjs – Creating password hashes (salt+hash) that are stored in a database.
- Jsonwebtoken (JWT) – Creates tokens that are used for authentication
- Passport(-jwt) – Used to authenticate API requests using JWT-strategy. Where used, jwt is passed as bearer token in http request and only valid JWT tokens gain access to the API.
- Multer – Used for handling multipart/form-data (profile avatar upload).
- CORS – Allows http requests coming from client side to server side
- MUI – This was used as MUI provides broad variety of different components that can be used in JSX. Also offers great customization of components without using any CSS.
- React-router-dom – Used for routing of the app.
- Highlight.js – Library that is used for highlighting code snippets.

Installation guidelines:

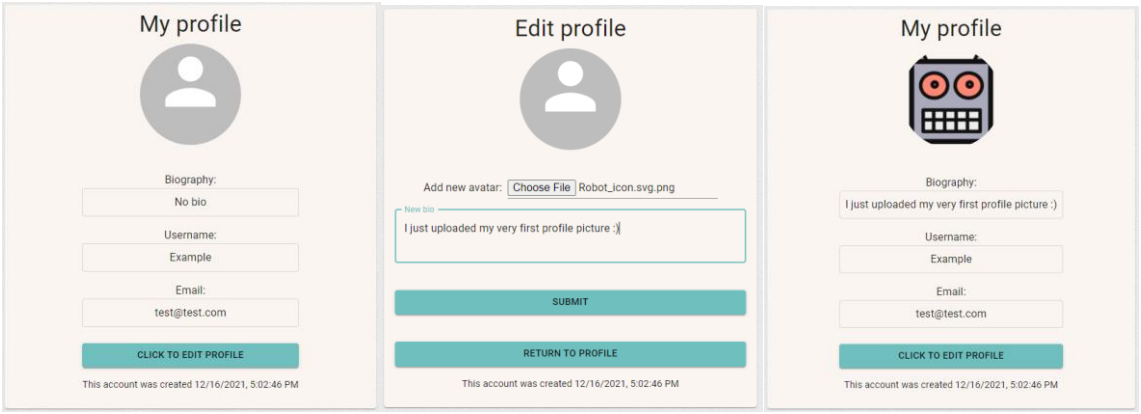
Installation guidelines are covered in README.md file, which can be found from the root of this repository.

User manual:

The usage of the site is rather simple; therefore, I will just explain some of the features.

When first starting the application, user will land on the homepage which is empty. From there the user can click on “Register” in top right corner. After successful registration user can log in.

After logging in, user can access their profile by clicking on the settings icon in top right corner. User can set a bio or profile picture by clicking “Click to edit profile”-button. To save changes press “Submit”-button.



(Example of editing a profile.)

Logged in users can also post posts. When submitting a new post, it is required to have a title and a content (description). Code snippets can be posted by clicking a Checkbox with a label “Add code?”.

After posting a post, the creator has options to remove or edit post by clicking the 3 dots in top right corner of a post. Beware: “delete” has no warnings and happens instantly when pressed. In addition to posting a post, logged in user can post comments to all posts, and again own comments have buttons to edit or remove them. When a post is removed, all its comments are removed too. When either post or comment is edited, a timestamp will appear in lower right corner.

When viewing a post in homepage or as a search result, a link “View post and comment” appears under. That link leads to a page in which a full post and its comments are displayed.

(Voted and edited comment, with the option to delete/edit comment in top right corner. Code highlighting and user icon also visible.)
By clicking the username next to a profile avatar user can see other profiles.

Voting system works as follows:

- User can have one vote per post/comment.
- Actions are reversible. By clicking the colored arrow again, a vote is removed. By clicking the uncolored arrow a vote is changed, for example (+1 -> -1).

Akseli Aula
0545267
12/2021

Finally, user can search for posts containing the search term. Search can be found by clicking the search icon in top right corner.

Summary of implemented features and total points proposal:

Implemented features	Points
Mandatory features	25
Additional features:	
Users can edit their own comments/posts	4
Utilization of a frontside framework (React)	5
Vote (up/down) posts and comments (one vote/user)	3
User profiles can have profile pics, which are shown next to posts/comments	3
User can click username and see user profile page where name, register date, picture and bio is listed	2
Last edited timestamp is stored and shown with posts/comments	2
Search that can filter out only those messages that have the searched keyword	2
Usage of highlighting library for code snippets (highlight.js)	2
Own features:	
Form validation when registering (strong password, email format) and properly stored passwords (salt+hash)	1
Mui snackbar alerts where necessary	2
Mui custom theme	1
Total points:	**52**

Why my own features should be accepted?

- When registering, form validation makes sure that the entered data is in correct form therefore it keeps the database consistent. Also, by requiring the use of strong passwords, user accounts are more secure. By storing the passwords as hashes, if the user's collection gets breached there are no plain text passwords.
- Mui snackbar alerts are used for informing users where necessary. For example, when input validation fails user is informed with an alert.
- Mui custom theme is used to make the user experience more pleasant with a nice color scheme.