Advanced Web Programming – Project Work

Veeti Rajaniemi
0599190

Tinder Clone

Technology Choices and Tools

I coded my project using React with Node.js backend and used Visual Studio Code. I used Materialize to get the basic functionalities work in both mobile and desktop environments. Users and chats are stored to MongoDB, and MongoDB Compass was used while doing the project.

Installation and Testing

First, install the application with the following steps:

- fork the repo using *git clone https://github.com/veetirajaniemi/Advanced-Web-Programming*
- go to the root folder (Advanced-Web-Applications) and install node modules with *npm install*
- run the app with *npm run dev* and it can be used in a development environment

The MongoDB connection used is *mongodb://127.0.0.1:27017/projectdb,* where users and chats are stored. The simplest way to test the application is to create a few new users (register and login). With the first user, it is good to check different pages, and after that a good way is to create matches between the users so the chat can be tried as well.

User Manual

The application has a navbar which includes buttons for different pages when user is logged in. For mobile devices, the buttons can be seen in a side bar after clicking on hamburger icon.

User can register and after that log in from the main page. Without logging in, user can't do anything. After logging in, user is redirected into profile page, where it is possible to write and save a profile text. Under the user's profile, in addition to the update profile button, there is a button which takes the user to the browse page to search for new users, and a button which takes user to a chat page.

On browse page, user can browse through other users. It is possible to like or dislike other users by clicking the corresponding buttons. After doing so, user sees an info text about it (or a possible match) and then the page reloads and a new user is shown. When all users are browsed through, user is redirected to the chat page.

If user has matches, a list of them is shown on left on chat page. User can click on a match and send a chat by writing a message on a field and clicking send button. New messages can then be seen by clicking a refresh button (or by choosing another person from the chat list and then again the old user). When a chat is long enough, it is possible to scroll through it. If user doesn't have matches, info text about it is shown on chat page.

User can logout from the logout button, which redirects to logout page. From the link on the page it is possible to get to the main page.

If there is an authentication error, user is redirected to the error page, which has a link to the main page.

Features and Points

Feature	**Points**
Basic features + documentation	25
Frontend with React	5
App has a navbar which updates based on the page – sidebar for mobile users with a hamburger icon	2
Chat is scrollable with enough messages and includes a time stamp for messages	2
Chat includes a refresh button to update it without page load	1
Authentication combines jwt with HTTP-only cookie, also possible to logout → redirection to logout page which includes a link to main page.	3
Simple front-end design on the whole app – colours, icons, usage of grids etc.	2
Info messages to user about different situations: when registration doesn't work, when user likes/dislikes other users, when user gets a match, when user can start a chat, when authentication does not work, text input placeholders…	2
Redirection to the chat page when no more users to browse	1

Error page shown when any authentication error occurs, includes a link to main page	1
Sum	44