Project Documentation

Lappeenrannan teknillinen yliopisto

Advanced Web Programming

Juho Rekonen, 441410

Description

The project was about creating a Full Stack Kanban Board Application by using Node.JS and Express.JS to handle the backend together with MongoDB, which would handle the database. On top of that, I decided to use React as the frontside framework, since I liked the way that websites are built with it, combining single components into bigger structures. To make the application more responsive, I decided to use Material UI the same way that was done in the lecture video on week 12. As usual, VSCode was chosen as the code editor.

The application lets authenticated users create columns, cards, and comments on their own Kanban Board (main page). Authentication is done by using JSONWebToken to authenticate registered users during the login process. Users must either register a new account or login to an already existing account to view and edit their board. This is where the home page comes in. It contains a welcoming message and asks the users to login or register to continue. Once the user has logged in, he/she will be redirected to the Kanban Board.

Creating columns works by giving them a title. After creation, the user can either delete the column or add cards to it. Creating cards works by giving them title, some content, a color (that must be chosen from four options), and an estimated time to finish the tasks in hours. These cards can then be moved up or down within a column, except when the user tries to move the very first card upwards or vice versa the very bottom card downwards. To move cards from one column to another, user must first create a new column (so that a column on the right or on the left exists). *Since the backend uses an async function during switching cards, the front end must be updated by reloading the page*. This is shown in the tutorial video below. So even if the front end doesn't update the cards after switching, the switch has happened.

In addition, the user has the option to add comments to his/her cards. Each component (column, card, comment) can be edited in some way, whether it's a title or content. The addition of components has been implemented to use different websites (e.g., /addColumn) in order to give the front-end better structure. Deletion of components is done by first deleting all child objects (if exists), and then the parent object. Finally, the user has the option to log out.

Overall, this project was very challenging but also very interesting to conquer. Since many companies nowadays use Node.JS and React to build websites and applications, I believe that this course has prepared me very well for the future.

Here's a YouTube video showcasing the project :

https://youtu.be/ZDxcWiTm0Mo

Installation Guide (How to run the project)

- Clone the GitHub repository (https://github.com/juhorekonen/Advanced_web_Project.git) to an empty folder using VSCode or other tools
- Make sure that you are in the "Advanced_Web_Project" folder. If not, type "cd Advanced_Web_Project" in the terminal
- Run "npm install" in the terminal to install all packages to both backend and frontend
- Run "npm run dev" in the terminal to start the application
- Open "http://localhost:3000/"

User Manual

- To begin using the application, use the buttons on the header or the links on the home page to either register a new user or log in to an already existing one
- During registration, do note that usernames are unique *(means you cannot choose a username that already exists)*, and passwords must contain at least one number and one special character
- After registration, the user will be redirected to the login page. Another way to get to this page is by first going to the home page, and then choosing to log in *(from link or header button)*
- After entering the user's credentials correctly to the login page, the user will be redirected to the Kanban Board. At this moment, an option for logging out appears to the header. In order to log out of the application, simply click Logout.
- To create new columns, click the "**Add Column**" -button found in the Kanban Board. The user will be redirected to another page where a title for the new column must be given. After filling in the fields, click "**Add Column**".
- To delete a column, click the "**Delete Column**" -button found inside the column. This will delete all possible child objects (*cards, comments*) before deleting the column.
- To update a column, click the title of the column. A text field will appear, and the user can rename the title. Click anywhere else on the UI to update the title.
- To create new cards, click the "**Add Card**" -button found inside a column. The user will be redirected to another page where title, content, color, and estimated time of completion must be given. After filling in the fields, click "**Add Card**".
- To delete a card, click the "**Delete**" -button found inside the card. This will delete all possible child objects (*comments*) before deleting the card.
- To update a card, click the title or the content of the column. In the case of the title, a text field will appear, and the user can rename the title. Click anywhere else on the UI to update the title. In the case of the content, a text area will appear, and the user can rewrite the content. Click "**Save**" to save your content. The time of creation (*createdAt*) will be updated upon updating content.
- To move a card within a column, click the "**Up**" -button to move the card upwards, and "**Down**" -button to move the card downwards. Do note that in columns that only have one card, clicking these buttons will not move the card, since it is already at the top and at the bottom simultaneously. Also, if a card is already at the top, the user cannot move it upwards. Vice versa, if a card is already at the bottom, the user cannot move it downwards.
- To move cards to another column, the user needs to make sure that more than one column exists. Click the "**Left**" -button to move the card to a column on the left side, and "**Right**" to move a card to a column on right side. Do note that if the card is on a column that is already on the very left side, the user cannot move it to the left. Similarly, if a card is on a column that is already on the very right side, the user cannot move it to the right. After successfully moving a card, the user must refresh the page to load the front-end.
- To create a comment, click the "**Add Comment**" -button found inside a card. The user will be redirected to another page where content must be given. After filling in the fields, click "**Add Comment**".
- To update a comment, click the content of the column. A text area will appear, and the user can rewrite the content. Click "**Save**" to save your content. The time of creation (*createdAt*) will be updated upon updating content.

- At any point in time, the user can switch languages from English to Finnish or vice versa by clicking the “**FI**”- or “**EN**” -buttons.
-

Installed Packages

The Application uses the following packages

- *concurrently* (allowing user to run both ends concurrently)
- *react, react-dom, and react-router-dom* (allowing the usage of routes in front-end)
- *i18next, 18next-http-backend, 18next-browser-languagedetector* (allowing UI translations)
- *@mui/material, @mui/icons-material, @materializecss/materialize* (allowing UI styling)
- *express* (allowing routing in the back-end)
- *express-validator* (allowing to validate user input during registration and login)
- *bcrypt, jsonwebtoken* (allowing password hashing and user authentication)
- *tsc-watch, nodemon* (allowing the server-side to restart running upon updating files)
- *dotenv* (allowing access to .env files)
- *morgan* (allowing user to see response statuses of his/her requests in the terminal)
- *mongoose* (allowing the back-end to connect to the MongoDB database)

Declaration of AI Usage

ChatGPT-4o was used quite a lot to help with finding errors (like small typos that took forever for me to find) and problem-solving, when for instance I tried to get the back-end routes for moving cards within a column and between columns to work, and I just couldn't figure them out and kept getting errors when handling card Ids and column Ids. ChatGPT was also used when I encountered error messages in both back-end and front-end that I had never seen before.

In addition to that, GitHub Copilot (which also uses GPT-4) was used for styling the front-end components, like containers, buttons, Material UI cards etc. For instance, I kept having errors with the column containers, where deleting one column would reduce the size of the others, and then the cards weren’t fully visible anymore. Copilot helped me set the widths correctly.

Grading

Feature	Points
Basic Features with well written documentation	25
Utilization of a frontside framework React	3
User can set the color of a card	1
Test Software for Accessibility	3
User has the option to double click any edible content and edit it	4
Cards can have comments on them, one or many	3
Cards and comments have visible timestamps when they've been created and updated	4
Cards have an estimated time when the work is done	1
Translation of the whole UI in two or more languages (EN, FI)	2
Total Points	**46**