Amal Khan

2021 July 11

Project Overview - FreshWorks Studio Full Stack Developer Selection Test

Problem Approach

The planning and implementation of this project took approximately 9 hours. I began by isolating key requirements and determining the best way to store data. Then I looked at the general flow of data access between the front and back ends and drew out concepts for the UI. Finally, I wrote an outline of the steps and the timeline of the build.

I started with the backend using Node.js and Express. I used mongoose to connect to a MongoDB database and set up a document scheme. I added endpoints for database data insertion and access and added joi validation as middleware. As I built out the endpoints I continuously tested them using Insomnia.

I then moved on to building the frontend using React. I connected it to the backend using axios, and styled it using Material-UI. Later in my frontend build, I decided to use React Hook Forms to minimize the form component re-rendering.

My final steps were to finish the page content and add the project README file.

Technology Choices

React

React allowed me to organize my code into modularized components, and the one-way data flow made it easy to modify child components without affecting their parents. The React virtual DOM also prevented constant costly HTML DOM modification.

Node.js/Express

I opted to use Node.js and Express because they allowed me to develop the backend using JavaScript. The asynchronous nature of Node.js also gave the advantage of better performance and speed. Additionally, with Node.js I was able to access tools like Express, joi validation, and mongoose.

MongoDB/mongoose

I used MongoDB for the database because I could store data in a similar way to how I would use it. If I had used an SQL database I would have had to transform the data each time I accessed or saved it. The use of mongoose gave me valuable features like scheme declarations and validation minimizing the possibility of bugs.