PPIT Project 2021

Kealan O’Callaghan – Dean McGowan

**Introduction**

Our project is a buy and sell web application specifically for automobiles. We aim to provide a clean and user-friendly front end that anyone can use. The application will allow users to register an account and then use that account each time they log in. Users can also post their own advertisements to the store. The user will also be able to add automobiles to a cart.

**Technology Used**

We used a wide array of technologies for this application. We used ‘Node Js’ for the server, a ‘Mongo DB’ database to store both user and product information, ‘Nodemon’ for the server updates, ‘JWT’ for the token authentication and finally ‘React’ for the application build.

Node JS will be used as the main back-end API service as the application will have a very push-based architecture. Node JS will be the most effective API for this because of its non-blocking, event driven servers.

MongoDB will be used as the database technology as the document data model is well suited to this style of application, as well as being coupled well with Node JS.

React will be used as the main building block of the application as it will allow the user to update and receive information without reloading pages. It is fast, scalable and simple.

Json web token will be used as the token authenticator as its self-contained and contains all the information it needs for authentication.

Redux was used in the login/register for global state management.

**Features of The Project**

A few of the key features for the project include the following: a multi-user system, a register/login that uses token authentication, talking to a Mongo database to store user login credentials and motor information.

**Implementation of Features**

We started with the log in/register page. It is a minimal full-stack login using the MERN stack. It also uses Mongo DB for the database, Express and Node for the backend, and React for the frontend. We also integrated Redux for state management for our React components.

Firstly, we created the backend. We initialised our backend using npm and installed necessary packages. We then set up a MongoDB database. We used Node JS and Express to set up the server. We created a database schema to define a user for registration and login purposes. We then set two API routes (register and login), using passport and jsonwebtokens for authentication and validator for input validation.

We then began the frontend using create-react-app. We made static components for our Navbar, Landing, Login, and Register pages. We also set up redux for global state management.

Finally, we connected the backend with the frontend. We linked Redux to our components. We also added code that would let the user stay logged in whenever the page was refreshed or if they leave the page altogether.

For the Buy and Sell, we used React to scaffold out the interface of a basic shopping cart, products page, and add product page. We used context to move data and methods between multiple components. We then got the application reading the data from our MongoDB database so that the products remained on the app after you left.

**Complications and learning outcomes**

* 1st March: There was an issue with the jwt as it couldn’t get a handle on the token and was returning ‘undefined’ meaning the user couldn’t login to the landing page. This was later solved by making sure all the classes could access each other with the exports to give the classes a handle on their variables and tokens.
* 7th April: We added a delete button to each automobile product. However, upon further discussion we removed it as although it worked, it didn’t fit with the products page as any user could delete another user’s product.

**Future Development**

If we were to take this application further, I think we could add some exciting new features. The checkout functionality is at the top of that list. At the moment we have a button to check out, but we have not implemented any features for it. I think a checkout feature would certainly add a lot to this project.