Dorset College Dublin

Module Title: Back-end Web Development

Assessment Type: MERN Application

Project Title: Attraktiva

Luciana Ramos Alves

Student Number: 18350

Report submitted in partial fulfilment of the requirements for the BSc in Computing at Dorset College.

GitHub repository link: https://github.com/ramosluciana/CA4-BEWD-Attraktiva.git

*The last update is on the gitpod branch

1. User Stories

Navigation/Search

- As a user I want to have product categories, so that I can browse through products easily;
- As user I wan to go to shopping list easily, so that I can see what I have added till now;
- As a user I want to search for item easily, so that I can buy exactly what I want;
- As a user I want to see my shopping list with current added items, so that I can save time and avoid mistakes

Product categories/filter

- As a user I want to browse by products, so that I can find options within the type of product;
- As a user I want to be able to filter by category, so that I can save time in searching;
- As a user I want to be able to filter by price and availability, so that I can select between my prince range and avoid frustration by not selecting unavailable items;

Product Details

- As a user I want to see the image of the product, so that I can have a better view of how the product looks like;
- As a user I want to be able to see the description of the product (price, colour, dimension, so that I can choose wisely;

Cart

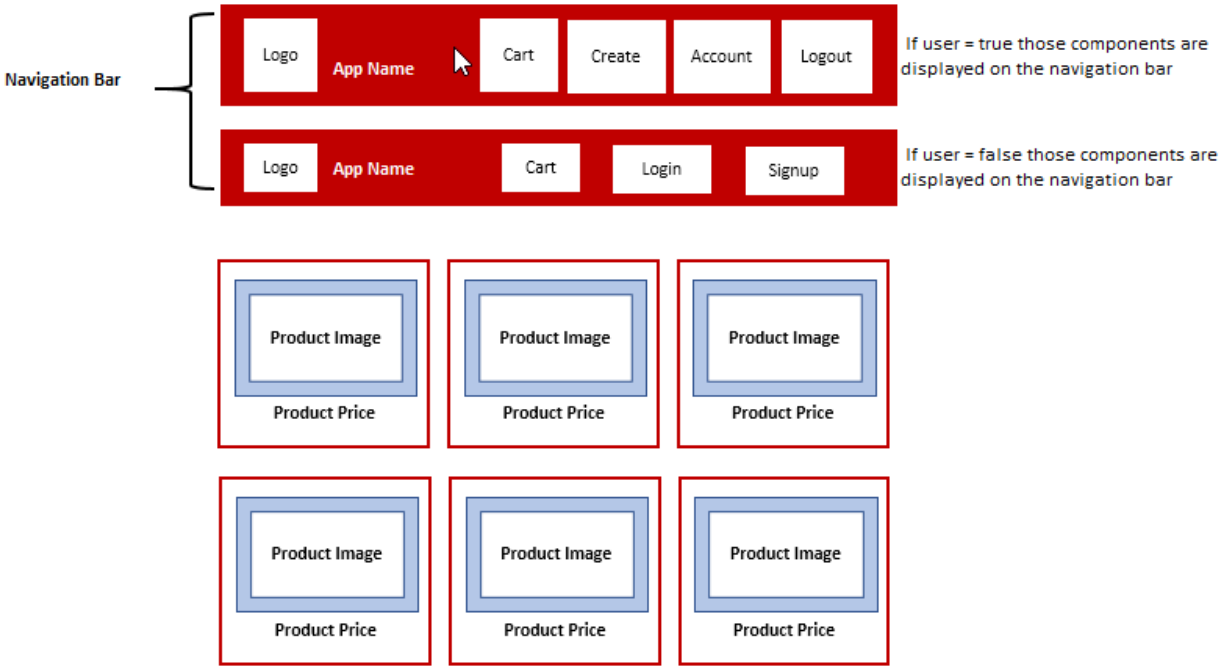
- As a user I want to be able to check the quantity easily, so that I can see if I need or not more items;
- As a user I want to able to remove an item which I don't want anymore, so that I can avoid mistakes;

Login

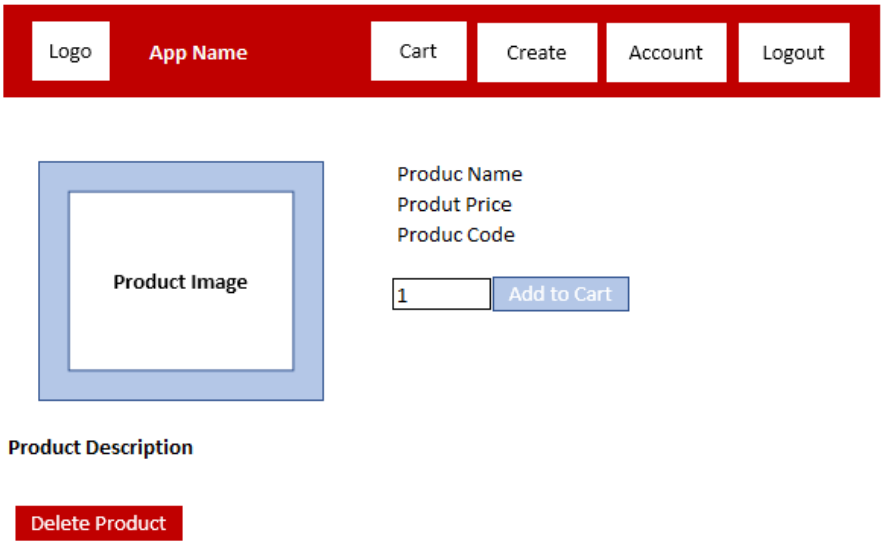
- As a user I want to create a profile, so that I can save my details;
- As a user I want to login without having to enter details every time, so that I can save time;
- As a developer manager I want to create login permissions, so that I can control who is allowed to add or remove items;

2. Web App Sketch design

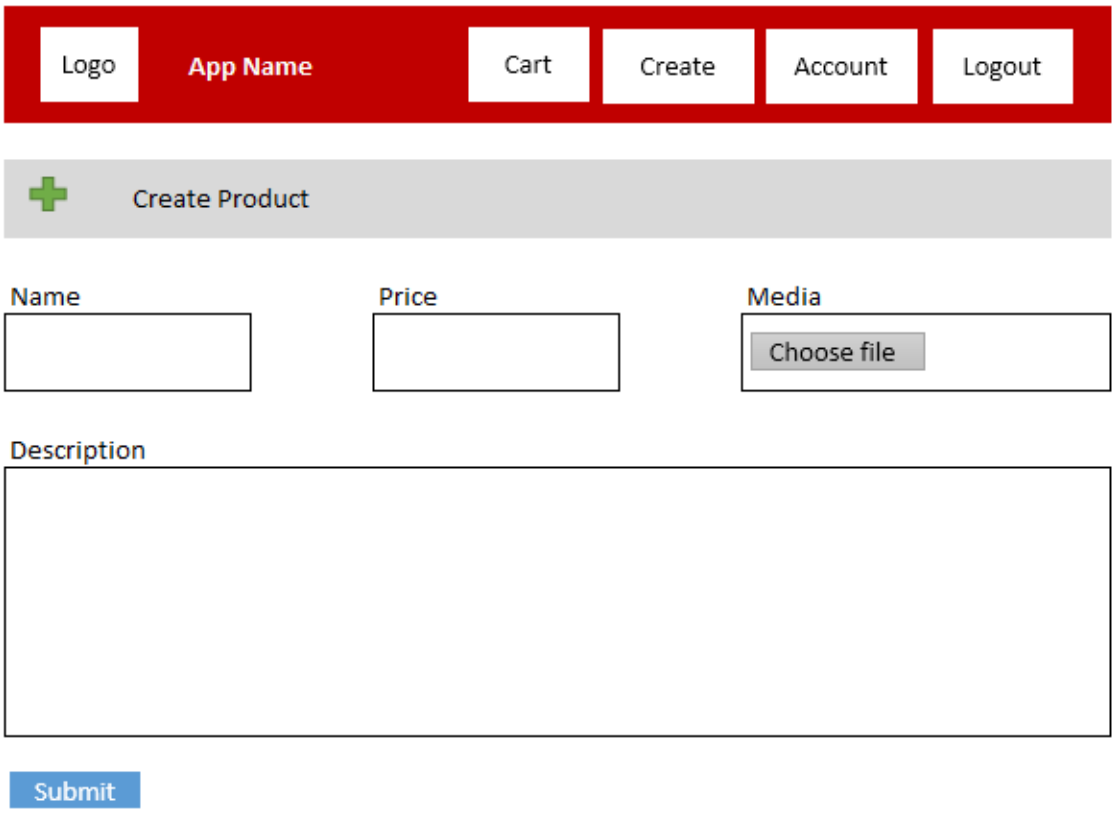
Home Page

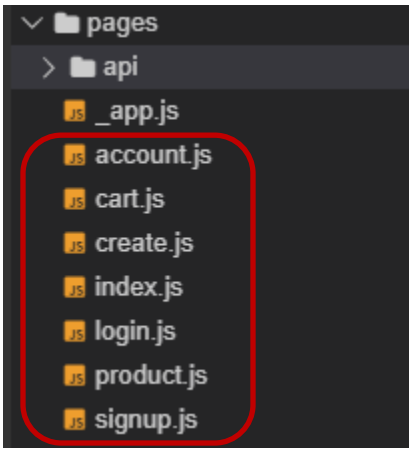


Product Page



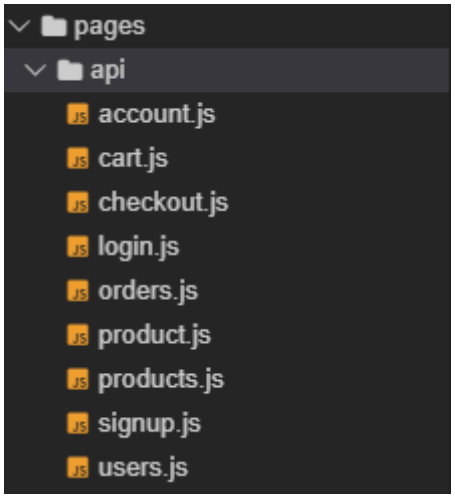
Create Product Page





Those are the components that the app contains, however not all of them had been fully developed.

3. REST API endpoints



All the API endpoint are inside the folder page > api

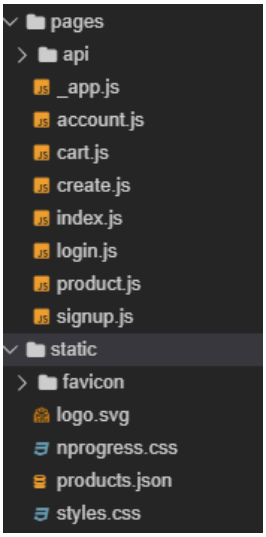
The js file was create considering the structure I had in mind the project, however due the lack of time to only files which actually contain some code are product.js which is handle the get request, post request and delete request and the products.js which connect to DB and create a route.

4. App Description

So, the idea of the app was to create an online shop to sell furniture, kind similar to Ikea.

On the development of this app I used Next.js framework, just I thought would be easy to work with once it doesn't require any configuration of webpack, it come with its configuration and also provide some good features and add tools.

However, for it to work I had to create a pages and static folder, next.js won't work without it, because it works with a file system-based routing, that is the reason why on the project structure I have those 2 folders.

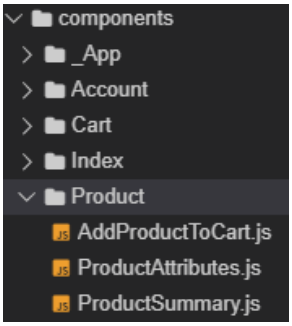


The static directory contains the page logo, the ccs file for the styling the app, and a products.json file containing a list of products which was basically used to populate the database for the first time.

The page directory would take care of all routes in the application, all the pages inside it contain their own react components.

Each page is consisting of multiple components, those components can be found on components folder;

E.g. Product page contains the components Add Product to cart, Product Attributes and Product Summary.



- **Add Product to cart** – As the name says basically would allow the user to add a product cart;
- **Product Attributes** - Basically make a request to the API/product file, and create the structure to allow the user to see product description, buttons, etc
- **Product Summary –** Just allow to see a summary of the product photo, name, price, etc.

To modelling the structure, the way I want the elements/data in the pages to be displayed I create the folder **models** which contain the modelling file for it component of the app.

The component library semantic UI react was used to help with the structure and appearance of the app because it provides several elements (buttons, Icons, Menu, etc) which facilitate on styling the app.

The folder **utils** basically contains some js file to create the database connection, the baseURL code which will be use control the requests, and some other files which I didn't have time to work with.

Cloudinary was used to take image file uploaded on the create new product page and return it as a media url to be store in the database, the connections for the database and cloudinary is on the file next.config.js

```
// must restart server whenever you make changes in next.config
module.exports = {
  env: {
    MONGO_SRV: "mongodb+srv://ca4user1:1Re$u4ac@ca4-lyweg.mongodb.net/static?retryWrites=true&w=majority",
    JWT_SECRET: "<insert-jwt-secret>",
    CLOUDINARY_URL: "https://api.cloudinary.com/v1_1/dwv21pssh/image/upload",
    STRIPE_SECRET_KEY: "<insert-stripe-secret-key>"
  }
};
```

5. Reflection

Understand how all the connection works was one of the big challenges, there are so much details and so many things going, that made very complicated to put in practice all functions I want in the app.

I had trouble to understand how mongo DB works and then to import the data from the products.json to it, the import command would just not work and I couldn't figure out what was wrong so I used the studio 3T platform to import it.

The connection with the server was another big problem couldn't get it right I guess it can be because I was working with next.js and some communication between client and server wasn't

properly settled, however, once it was my first time working with all those elements/frameworks I still not sure how to look and understand error properly.

Anyway, the server connection led to other problems as I couldn't test the application 100%. Tested it using Visual Studio Code and was able to create and delete product but couldn't do the same using Gitpod, also couldn't get to deploy it to Heroku every time I tried to log in from the terminal I would get a message saying it was forbidden.

Not really sure about all the things I could have done different, maybe would have been easier and I could have got more things done if I had just followed the structure given by the lecturer and worked on it instead of trying to learn different a framework and created a big project structure especially considering the deadline.

However, although there is a bit of frustrated building this app once I couldn't get it to work as I wanted, I learnt quite a lot during the process, if I had just followed the structure given probable wouldn't have understand properly some functions and components, props, and so on.

6. Reference

gitconnected. 2020. Tutorial Review - MERN Stack - The Complete Guide. [ONLINE] Available at: https://gitconnected.com/learn/node-js/mern-stack-the-complete-guide-1d0e31. [Accessed 30 April 2020].

API Routes: Dynamic API Routes | Next.js. 2020. API Routes: Dynamic API Routes | Next.js. [ONLINE] Available at: https://nextjs.org/docs/api-routes/dynamic-api-routes. [Accessed 03 May 2020].

Layouts | Semantic UI. 2020. Layouts | Semantic UI. [ONLINE] Available at: https://semantic-ui.com/usage/layout.html. [Accessed 03 May 2020].

Studio 3T. 2020. How to Connect to Compose | Studio 3T. [ONLINE] Available at: https://studio3t.com/knowledge-base/articles/connect-to-compose/. [Accessed 02 May 2020].

MDN Web Docs. 2020. Express Tutorial Part 3: Using a Database (with Mongoose) - Learn web development | MDN. [ONLINE] Available at: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/mongoose. [Accessed 03 May 2020].

Create a Web App and RESTful API Server Using the MEAN Stack | Heroku Dev Center. 2020. Create a Web App and RESTful API Server Using the MEAN Stack | Heroku Dev Center. [ONLINE] Available at: https://devcenter.heroku.com/articles/mean-apps-restful-api. [Accessed 04 May 2020].

Stack Overflow. 2020. javascript - Getting "npm ERR! code ELIFECYCLE npm ERR! errno 126" while running npm install - Stack Overflow. [ONLINE] Available at:

https://stackoverflow.com/questions/55022219/getting-npm-err-code-elifecycle-npm-err-errno-126-while-running-npm-install. [Accessed 06 May 2020].

Nic Raboy. 2020. Building A REST API With MongoDB, Mongoose, And Node.js. [ONLINE] Available at: https://www.thepolyglotdeveloper.com/2019/02/building-rest-api-mongodb-mongoose-nodejs/. [Accessed 08 May 2020].

The Code Barbarian. 2020. Unhandled Promise Rejections in Node.js | www.thecodebarbarian.com. [ONLINE] Available at: https://thecodebarbarian.com/unhandled-promise-rejections-in-node.js.html. [Accessed 08 May 2020].

GitHub. 2020. Problems when Next is installed globally and you want to use react alpha version with hooks. · Issue #6184 · zeit/next.js · GitHub. [ONLINE] Available at: https://github.com/zeit/next.js/issues/6184. [Accessed 19 May 2020].