Overview

As NRail we would like to display our items in a detailed tiled view. The view should show a list of items to be sold.

Page description

- The page should have a title and navigation with the home page and another page with some text on it. The content of the second page is not important. **DONE**
- Each item on the page has an image, name, description, price, the amount available as a select list and a button, “add to cart”. The layout of the information is to you. **DONE**
- On clicking the add to cart button:
  - The selected amount is added to the total number of items in the basket and shown in the toolbar at the top of the page. **DONE**
  - After the total number of items has been updated the select list is to be reset to 1. **DONE**
- A checkout button should be placed at the top of the page that will output all the items in the basket. This output can be just a simple console log or how you see if. `I created a separate page for cart. It contains the listing of added products with quantity. additionally, I also calculated total prices of added product items.` **DONE**

Example shop item:

```
{
 name: 'nrl-engine',
 description: 'faster than anything you have ever seen before',
 price: '400',
 image: 'any-image-you-like.jpg',
 amountAvailable: 40
}
```

Requirements completed

1. Create a react project from scratch or using create-react-script. **DONE**
2. Simulate a fetch response to get the data required to build your application. (The shop items) `I created a products data json file "products.json" as a fetch response. It can be get from remote api and it can be implemented in improvements if you give me more time to do it.` **DONE**
3. Your shop should have a minimum of 10 item. **DONE**
4. Reuse code where possible **DONE** You are free to use any libraries as you see fit. I love `loadash` wanted to use it 😃. but I will use it in improvements. NB: The aim of the exercise is not to have a perfect looking page but to assess your ability of breaking down the requirements into a functional(working) code. code is refactored and broked down into components like `MainHeader.tsx, MainMenu.tsx,`

`ProductCard.tsx`. It can be further refactored after discussing with your team. I am flexible to learn these things.

Tools integration

- Cypress and Jest is integrated for unit testing integration testing and end to end testing.
  - **prequisites:** install all packages using `npm install` or `yarn` and install cypress globally using `yarn cypress -g`.
  - to get cypress report run `yarn cypress-test` or `npm run cypress-test`
  - to see Cypress GUI, run `yarn cypress` or `npm run cypress`
- Docker and docker compose is integrated.
  - **Prequisites:**: install docker-destkop, vs code, vs code exentions for docker and docker compose,
  - run `docker-compose up -d` to run your app container.
- Azure-pipeline also integrated
  - CI/CD pipeline file is created. you make it part of Azure CI/CD pipeline if interested in to do so.

Let me know your availability to have a review and discussion meeting.

Looking forward to hearing from you soon

BR
Waseem
waseembt10029@hotmail.com