Shopping Page

Summary

Develop a small shopping page with NodeJS, MongoDB, Apollo GraphQL and ReactJS. The frontend should show three different categories of items in the sidebar. Clicking on the category should fetch and render the corresponding items. There should also be a shopping cart, to which the user can add the desired items. Authentication, user accounts and user roles are not required.

Frontend (ReactJS)

You are free to design the frontend as you like. If you want, you can use UI frameworks to save time. CSS is fine, SASS is preferred.

Header:

- Name of the currently logged in user (hard coded user, no login required)
- Avatar of the user (picture can be anything)
- Logo of page (you can take the jacando logo, or something else)
- Cart, which on click, opens a modal (see Modal description below for description)

Sidebar:

Available categories:

- Vegetables (initial view)
- Fruits
- Cheese

When the category is clicked, it should display a spinner until the data is fetched from the backend via GraphQL.

Once received, it should render each item with following information:

- Price
- Short Description
- Amount in stock available

Shopping Cart Modal:

- Needs to display all the selected items
- Needs to display the total cost
- There should be a possibility to add more of the item or remove it.
- There should be a Button for “Buy”, which once clicked, should send a request to the backend, acknowledging it of the bought items.

Main View:

Implement client-side routing with “react-router-dom” with the following routes:

/vegetables, /fruits, /cheese:
should display a spinner while fetching the paginated (max 5 at once) data via graphQL. Once fetched, it should render the received items.

With a navigation (arrow left - right) it should be possible to fetch the next or previous 5 items of the current category.

Backend (nodeJS, express, Apollo GraphQL, MongoDB, Mongoose)

Implement a query with Apollo GraphQL that fetches all items paginated for a given category name and page number. Per page, it should return a maximum number of 5 items.

In addition, implement a mutation that receives an order and stores it in the database.