Grocery Shopping WebPage

Think about following :-

1. Shopping Experience
2. Including stock visibility
3. Shopper centric-offers : customized user dashboar d

BACKEND API : products classified in three categories : fruit, drinks and bakery

And all products are shown by queryParam category=all

API : <https://uxdlyqjm9i.execute-api.eu-west-1.amazonaws.com/s?category=all>

COMPONENTS :-

1. Card for products display :-

* Add button to add to cart
* If product inventory is 10 or more : “available” OR display exact number of items

1. Product Details Page : depending upon API response

FUNCTIONAL REQUIREMENTS :-

1. Search
2. Offers : See items with offers [show free item being added when the offer applies]

* 6 cans of coke 1 free
* 3 croissants, 1 coffee free

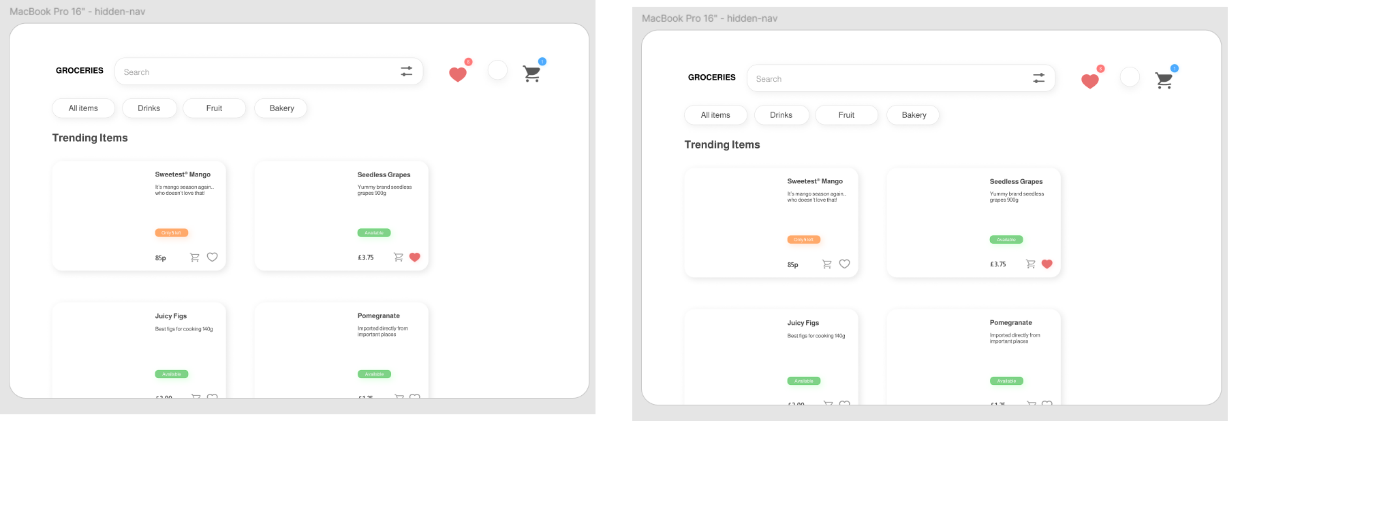
1. Go back and forth the 2 pages retaining cart info
2. Add or remove items smoothly, offer being applicable or not should be taken into consideration
3. Subtotal, discount and totals must be calculated and displayed during checkout
4. Responsive Design

SUBMISSION :-

1. Host on platform like Vercel
2. Public Git Link. Repo must include Readme.md file [setup instructions and how to run the code]
3. Detailed implementation flow, your approach, business logic and scenerios : DOCUMENT YOUR WORK

SOFTWARE ENGINEERING PRACTICES :-

1. Class Diagram
2. Object Diagram
3. Activity Diagram
4. Sequence Diagram
5. State Transition Diagram
6. Package Diagram
7. Component Diagram
8. Communication Diagram
9. Deployment Diagram



So, in order to keep the track of which items user has kept in liked and which in cart and what are offers : for now I will create three JSON objects to be stored in localStorage :

1. Offers object : contains whatever offer shop is providing
2. Liked object : stores info of what items customer has added to wishlist
3. Cart Object : stores info of what items customer has added to cart

CRITERIA THAT I SET FOR TRENDY ITEMS : rating grater than or equal to 4.5

* Liking and DisLiking is handled [here I took track of items with the hekp of item.name which ideally should have done on the basis of id of the item, because items with same name can co-exist but IDs will always be different]
* Cart icon button :

1. Ideal Situation : a dialog box should open up, giving a little bit detailed info of the product and letting user select how much quantity of this product user wants to buy
2. Current Solution : user clicks on cart icon and item is added to cart and the Main Cart Icon on the app, its count increases.

* I realized that we need to keep track of inventory with respect to cart items, so I need one more localStorage object : Inventory : which gets populated with item-name corresponding to quantity available
* D

UI COMPONENTS :-

* I was supposed to create generic button component : which could be used all over the platform if platform scales, but for now I am okay with no button component