

React Test

Test Name: Ecommerce React app

Deadline: 27th Nov midnight

Problem statement

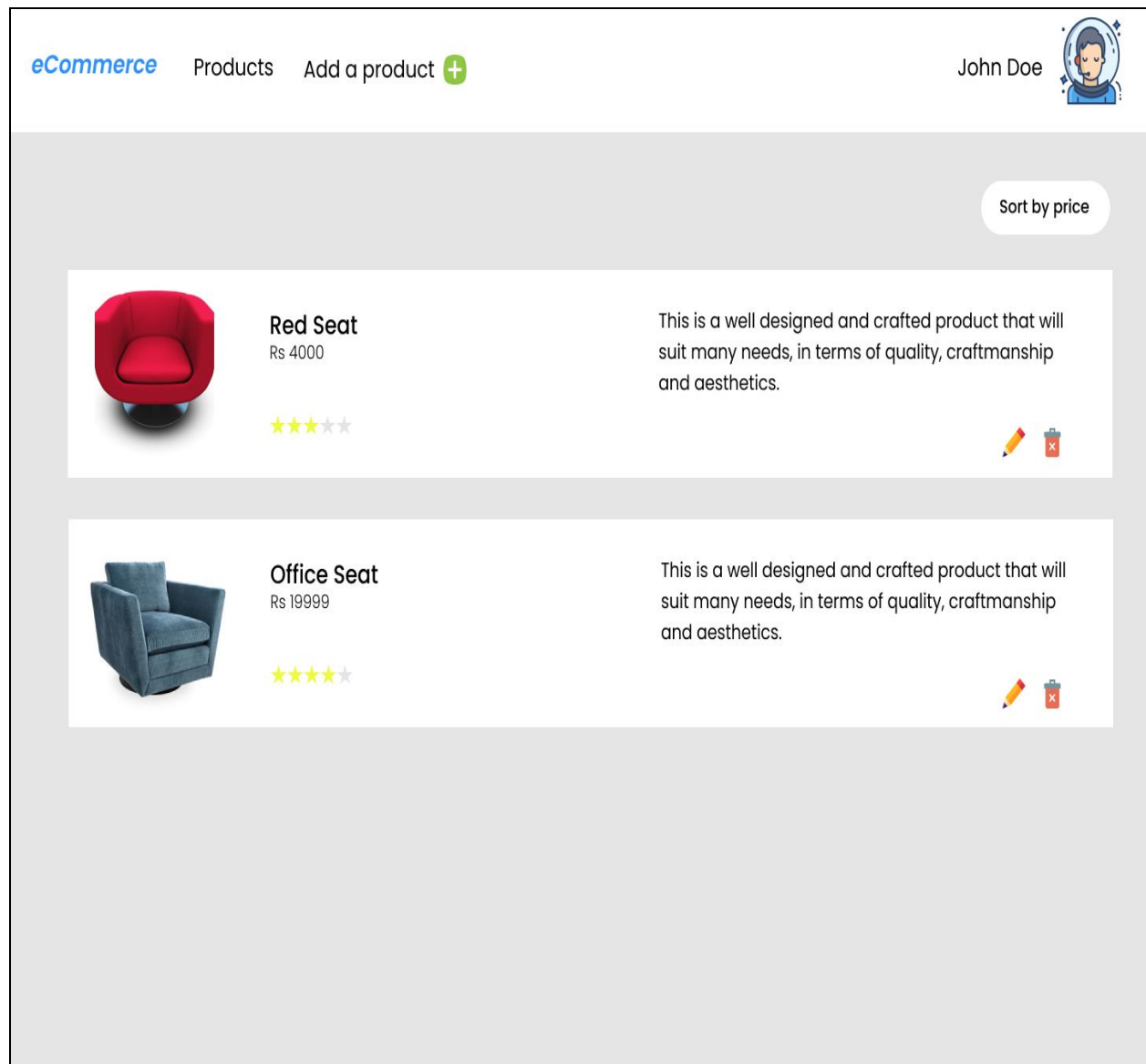
Create the frontend for an ecommerce website. You can check out the mockups down below.

Instructions

- Use the following service to create a dummy ecommerce api service (create good amount of data): <https://my-json-server.typicode.com/>
- Functionality
 - Navbar
 - Show cart items count
 - Show relevant navigation links
 - All products page
 - Show list of products from the API (using the above server)
 - Each product is editable by clicking on the **“pencil”** button. And we can edit that product inline. On finish editing the product, show some sort of Alert/Notification
 - Each product is deletable, on clicking of the delete button you should delete the product and show some sort of Alert/Notification
 - Implement a sort button. On clicking it should sort by **“price”** and show a cross button just beside it (see the sort view). On clicking the cross button remove the sort.
 - Give button to add a product to cart
 - Create page
 - On clicking of the Add button add the product in the DB, and show some sort of Alert/Notification
 - Product detail page
 - Show all the details of a product
 - Give button to add a product to cart
 - Cart page
 - Show all the items in the cart
 - Handle errors and success alerts etc.
 - Handle errors as well from the API and show appropriate Alert/Notification
 - You can use react-router if you want. It's not mandatory.
 - **You have to use redux. It's mandatory.**
 - Bonus feature: Make the redux data persistent such that after refresh, the cart items etc remains same

- You are **ALLOWED** to style the app the way you like. Do whatever you want with the styling. **ONLY** the functionality should work.
- You are **ALLOWED** to google and read how to approach the problem
- **DO NOT** copy and paste code from the internet
- **DO NOT** cheat with other students, I am trusting you all for this one!
- Once you have finished with the test, make a video recording your computer screen (via phone or a software like OBS/screencastify etc) and explaining how you have approached the problem **in code and showing the final product**. The video can be **UPTO 3 mins** long. ***Please don't record videos longer than that.***
- Upload the code to github.
- Host your code on github (using gh-pages)/netlify/firebase etc.. (This is mandatory)
- Once you have recorded the video ,upload it on youtube (unlisted or whatever), drive or wherever you want to upload it and then fill out this form - <https://forms.gle/bsLGfrn78vTvnS3h8> with the video link and the github link.


The App *(Zoom in to view the images properly)*





Sort price view


eCommerce

Products

Add a product 






John Doe 

Sort by price 






Red Seat

Rs 4000








This is a well designed and crafted product that will suit many needs, in terms of quality, craftsmanship and aesthetics.







Office Seat

Rs 19999




This is a well designed and crafted product that will suit many needs, in terms of quality, craftsmanship and aesthetics.




Edit product view (can edit description, name, price, rating)


eCommerce

Products

Add a product 

John Doe 

Sort by price




Red Seat

Rs 4000

Rating
3

This is a well designed and crafted product that will suit many needs, in terms of quality, craftsmanship and aesthetics.

CancelSave





Office Seat

Rs 19999

★★★★☆


This is a well designed and crafted product that will suit many needs, in terms of quality, craftsmanship and aesthetics.




Add product view

eCommerce

Products

Add a product 

John Doe 

Add a Product

Name

Description

Price

Rating

Add