Case Study

You are tasked to create a simple eCommerce web app using React. The specifications are provided below. Create a new React project to do it, do not use the sample app as your starter project kit.

General Requirements

1. You are only required to develop the frontend portion.
2. Implement routing using react-router-dom.
3. Use localStorage simulate the backend (Save and persist data).
4. You do not need to develop an authentication system to store the data by account.
5. Styling is optional (no restriction on CSS libraries), but the UIUX should still be usable at the minimum.
6. Store the list of products available to purchase in a JavaScript object or JSON file.
    a. You are required to at least have 3 products.
    b. Your product data should contain information
        i. ID
        ii. Name
        iii. Description
        iv. Price
        v. Path to image (optional)

Below is a sample of the product data object

```
[
  { "id": 1, "name": "Product 1", "description": "Product 1 description",
"price": 100, "path": "/images/product1.jpg" },
  { "id": 2, "name": "Product 2", "description": "Product 2 description",
"price": 150, "path": "/images/product2.jpg" }
]
```

Pages

Below is the list of pages that you are required to create for the web application.

Page	Path	Description
Product List	/	Display a list of products
Specific Product	/product/:id	Display details for a specific product
Cart	/cart	Display items in the cart added by the user
Checkout	/checkout	Display checkout form and cart summary
Order Confirmation	/orderconfirmed	Display order summary after checkout completion

Detailed Requirements

Navbar

Include a navigation bar with the items below:

- Product – "/"
- Cart – "/cart"
    - Only accessible when logged in (go to Cart section for more information)
- Login/Logout button

The navbar should be present on EVERY page.

You are only required to store the logged in or logged out state globally.

Product List Page

Route: /

- Display ALL the products in the page. Each product should display:
    - Product image (Only if you are adding images in)
    - Name
    - Price

- Clicking on the product container should navigate to its specific product page (/product/:id).

Specific Product Page

Route: /product/:id

- If the product ID does not exist/is invalid, redirect to homepage (/)

- Display detailed product information:
    - Name
    - Price
    - Description

- Implement the functionality to:
    - Adjust the quantity of product
        - Create the buttons +/- to set the quantity.
        - Minimum quantity is set to 1
        - Max quantity is set to 5.
    - Add to Cart button:
        - When clicked, the product and quantity are saved to the cart (saved in localStorage).

        - If the product already exists in the cart, its quantity is added to it instead of being overridden.
        - This button is disabled if the user is not logged in.
            - Display “You need to be logged in to add items to cart.” message

Cart Page

Route: /cart

- If the user is NOT logged in, it should display a message “You need to be logged in to access the cart” message.

- If there are NO items in the cart, it should display a “No items in cart” message. This error message is LESS priority over the above message. (E.g., if the user is not logged in AND the cart is empty, it should show the not logged in message)

- Display the list of products in the cart localStorage. Each product should display:
    - Name
    - Price
    - Quantity
    - Total (Price x Quantity)

- Implement functionality to:
    - Remove product from the cart with a Delete button
    - Remove ALL products from the cart with a Clear Cart button
    - Update the quantity of the products directly in the cart
        - If the quantity is 0, delete the item from the cart
        - Max quantity per product is 5.
        - Create the buttons +/- to set the quantity.

- Display the subtotal price of ALL items in the cart

- Display the shipping cost (any number is fine)

- Display the total price of ALL items (adding in the shipping cost)

- Checkout button
    - Clicking the button should navigate to the checkout page (/checkout)

Checkout Page

Route: /checkout

- If the cart is empty (no products in localStorage) OR the user is NOT logged in, automatically redirect to the homepage (/)

- Display the list of products in the cart:
    - Name
    - Quantity
    - Total (Price x Quantity)

- Display the subtotal, shipping and total price of the cart

- Provide inputs for: (Inputs only need to be filled, doesn't need to be valid)
    - Name
    - Email
    - Address (for this assignment, just use 1 textarea for simplicity)
    - Card Information

- Purchase button, when clicked it should:
    - If ANY of the inputs are empty, prompt the user.
    - If ALL inputs are filled
        - Clear the cart in localStorage
        - Redirect to order confirmed page(/orderconfirmed)

Order Confirmed Page

Route: /orderconfirmed

- If no passed state exists, redirect to homepage (/)
    - If you forgot what this is, look into the route redirection section

- Display the order summary:
    - Name
    - Email
    - Address
    - List of products in the order
        - Name
        - Quantity
        - Total (Price x Quantity)
    - Subtotal of ALL items and Shipping
    - Total price (Subtotal + Shipping)

Component Structure

This is only a general guideline; you can choose how you want to structure your project and how many components you need.

- **App**: Handles routing and rendering the navbar and main routes
- **Navbar**: Contains links to Product List and Cart page and button to login/logout
- **ProductList**: Displays the list of products
- **ProductPage**: Displays details of specific product and allows adding to cart
- **Cart**: Displays items in the cart, allows deleting and updating quantity
- **Checkout**: Displays the checkout form and cart summary
- **OrderConfirmed**: Displays order summary

This is only for the pages; you might need more files than this such as context providers or custom hooks.

A Helping Hand

- This case study was made in consideration with all the tutorials you have went through. You do not need additional React features not taught in here to develop this.

- Implementations are mostly the same as the tutorials with additional JavaScript coding or more complex objects.

- You are not forced to do this alone! Remember you can always look for answers on the internet (or ChatGPT).

Bonus

If you are up for it, you can implement some of the extra features:

- Better checkout form
    - Multiple address fields
    - Form validation, ensure address and card information is valid.
- Styling! Make your web app look nice.
- Responsive design. Make your UI work well with smartphone screens.
- Product list can be filtered to show specific category of item (E.g., Windows OS or Mac OS)

Evaluation Criteria

Functionality: Does the app meet all the requirements, including adding/removing items from the cart and handling form submissions?

State Management: Is state managed properly using useState and persisted with localStorage?

Routing: Are the routes correctly implemented, and can the user navigate between different pages?

User Experience: Is the app user-friendly, and does it handle errors or edge cases gracefully?

Code Quality: Is the code well-structured, clean, and easy to maintain?