ENSF-381: Assignment 4

Winter 2024

Department of Electrical and Software Engineering

Schulich School of Engineering

Due: March 25, 2024, at 11:59 PM

Welcome to the next phase of our web development journey! In this assignment, we are diving into the world of React to elevate our E-commerce website. Building on the foundational knowledge acquired in previous assignments covering HTML, CSS, and JavaScript, we are now ready to harness the power of React to create a more dynamic and interactive website. We will be focusing on creating reusable components, managing state, and incorporating interactivity to make our online store more engaging and efficient.

Administrative Details

- The submission should be as a single compressed file (.zip) which includes a complete source code and resources (e.g., images) ready for execution without modifications. The name of the submitted file would have the following format: "Assignment#_ ENSF381_Section#_Group#". For example: "Assignment4_ENSF381_L01_Group01".
- When working in a group, submit ONLY ONE copy; i.e. do not submit 2 separate copies.
- Please mention the group member's name and UCID at the top of your code as source code comments in the index.js.
- Do not add any additional styling, functionality, or components if you haven't asked in the assignment. You must name the component as specified in the assignment.

Project Setup: 20 marks

1. Create React App:
   - Run **npx create-react-app my-ecommerce-app** in your terminal to create a new React project named my-ecommerce-app.
2. Navigate to Project Directory:
   - Move to the project directory: **cd my-ecommerce-app**.
   - Create a text file named "**readme.txt**" in the project directory. List all the libraries you used/imported in your React project. Once listed, provide the necessary commands to install these libraries using npm (e.g., npm install styled-

components). This will ensure that all required dependencies are installed, and your project can run smoothly in the marker's environment.

3. File Structure:
   - Within the **src** folder, you will find the following files:
     - App.js: The main component that renders other components.
     - index.js: Entry point of the React application.
     - index.css: Default CSS file for the project.
4. Create a directory named **component** inside the **src** directory. We will place all the components built in this assignment in this directory.
5. Create a directory named **data** inside the **src** directory.
6. Create two files named **reviews.js** and **products.js** into the **src/data** directory of your React project.
7. Copy the following contents into the **reviews.js** file:

```
const reviews = [
          {
           productName: "Product A",
           customerName: "John Doe",
           reviewContent: "Great product, highly recommended!",
           stars: 5
          },
          {
           productName: "Product B",
           customerName: "Jane Smith",
           reviewContent: "Good quality, fast shipping.",
           stars: 4
          },
          {
           productName: "Product C",
           customerName: "Alice Johnson",
           reviewContent: "Satisfied with the purchase.",
           stars: 4
          },
          {
           productName: "Product A",
           customerName: "Bob Brown",
           reviewContent: "Could be better.",
           stars: 3
          }
         ];
         export default reviews;
```

8. Copy the following contents into the **products.js** file:

```
const product = [
	{
		id: 1,
		name: "Product 1",
		description: "Description for Product 1",
		price: 10.99,
		image: 'images/product1.png'
	},
	{
		id: 2,
		name: "Product 2",
		description: "Description for Product 2",
		price: 20.99,
		image: 'images/product2.jpg'
	},
	{
		id: 3,
		name: "Product 3",
		description: "Description for Product 3",
		price: 10.99,
		image: 'images/product3.jpg'
	},
	{
		id: 4,
		name: "Product 4",
		description: "Description for Product 4",
		price: 10.99,
		image: 'images/product4.jpg'
	},
	{
		id: 5,
		name: "Product 5",
		description: "Description for Product 5",
		price: 10.99,
		image: 'images/product5.jpg'
	},
	{
		id: 6,
		name: "Product 6",
		description: "Description for Product 6",
		price: 10.99,
		image: 'images/product6.jpg'
	},
	{
		id: 7,
            name: "Product 7",
            description: "Description for Product 7",
            price: 10.99,
            image: 'images/product7.jpg'
      },
      {
            id: 8,
            name: "Product 8",
            description: "Description for Product 8",
            price: 10.99,
            image: 'images/product8.jpg'
      },
      {
            id: 9,
            name: "Product 9",
            description: "Description for Product 9",
            price: 10.99,
            image: 'images/product9.jpg'
      },
      {
            id: 10,
            name: "Product 10",
            description: "Description for Product 10",
            price: 10.99,
            image: 'images/product10.jpg'
      }
];
export default product;
```

9. Create a directory named **images** inside the **public** directory. Copy all the provided images with the assignment into the **public/images** directory.

Homepage Requirements: 30 marks

- Create the homepage layout with React components.
- The homepage will include navigation links to other pages of the website, a section displaying the company's vision and mission, and showcasing feedback from previous customers.
- Display at least two customer feedback entries fetched from a data file (**reviews.js**) stored within the project.
- Each customer feedback entry should include the customer's name, review content, and a rating.

Components and Structure:

- **Header:**
    - Display the website's logo at the top left and the company name at the top right.
    - Include navigation links to other pages (home, products, login). Utilize **React Router DOM** to set up routing and navigate between pages. Style the navigation links with equal space between them for smooth navigation.
- **HomeMainSection:**
    - This component includes section for the company's vision and mission, and customer reviews. The section will describe the company's mission and vision under the heading "**About Us**", a "**Shop Now**" button that directs users to the **Productpage** when clicked, and customer reviews under the heading **"Customer Reviews".**
    - Display customer reviews fetched from **reviews.js** in the "**Customer Reviews**" section. You must display 2 (two) reviews in this section.
    - Each review should include the customer's name (**customerName**), review content (**reviewContent)**, and a rating (**stars**). For example, if the value of **stars** is 3, present 3 stars (⭐⭐⭐) in the rating.
    - Every time the **HomeMainSection** component renders or the page is refreshed, the "Customer Reviews" section will be updated with 2 (two) random reviews fetched from **reviews.js**. You can use **useEffect** hook to select random reviews from the **reviews.js**.
- **Footer:**
    - Add a footer with a copyright notice at the bottom of the page.

Now, implement **Header, HomeMainSection,** and **Footer** components.

Assembling Homepage Component:

- Inside the **src/components** directory of your React project, create a new file named **Homepage.js**.
- In the **Homepage.js** file, import the **Header**, **HomeMainSection**, and **Footer** components to compose the structure of the homepage.

The structure of the Homepage will have the following:

```
<div>
      <Header />
      <HomeMainSection />
      <Footer />
</div>
```

Integrating Homepage into the Project:

- In the **App.js** file, import the **Homepage** component and render the **Homepage** component within the **App** component.
- See the "**Homepage.gif**" for a visual representation of the Homepage.

Productpage Requirements: 50 marks

- Create the Productpage layout with React components.
- Add the **Header** component **(Header.js)**, that we have already created earlier, at the top of the **Productpage** layout to maintain consistency with other pages.
- Display **<u>all the products</u>** fetched from the **products.js** file.
- For each product, display the product image, name, product price, and "Add to Cart" button.
- Implement a toggle mechanism to show/hide product details on hover over the product name.
- When a user clicks on the "Add to Cart" button for a product, the product details including their image, name, price, quantity, and total price of the product should be added to the cart.
- Implement functionality to calculate and display the total cost of all items in the cart.

Components and Structure:

- **Header:**
  - Reuse the **Header** component created earlier in this assignment and place it at the top of the **Productpage**.
- **ProductItem:**
  - **ProductItem** component displays individual product details including the product image, name, price, and "Add to Cart" Button.
  - Implement a hoovering effect on the product name, product description will be displayed onMouseEnter and disappear onMouseLeave. You can use the **useState** hook to implement this toggling functionality.
  - Once the "Add to Cart" button is clicked, the product with its associated data will be displayed as a **CartItem.** You will receive the details of **CartItem** component in the later part of this assignment.
- **ProductList:**
  - **ProductList** component displays a list of products.
  - Import the **ProductItem** component to render individual product items.
  - Import the **productsData** array from the **products.js** file, which contains information about all the products.

    - Pass each product in the **productsData** array to the **ProductItem** component to create an instance of the **ProductItem** component.
- **CartItem:**
    - **CartItem** component represents an individual item in the cart. The **CartItem** component will render this product with its associated details when “**Add to Cart**” is clicked, as specified earlier in the assignment. This component will display the product image, name, price, quantity, and total price of a product. For example, if Product 1 has a quantity of 2 and unit price of $10, the total price would display a value of $20. Also, if you click the “Add to Cart” of a product that is already in the cart, the quantity of the product will be increased by 1 instead of creating a new **CartItem** component.
    - Include and display a button named **“Remove”** with this component. Clicking the button will decrease the quantity of the product by 1. When the quantity becomes 0, the item will be removed from the cart.
- **Cart:**
    - Pass each item in the cart to **CartItem** component to create an instance of the **CartItem** component.
    - Calculates and displays the total price of all items in the cart.
- **Footer:**
    - Reuse the **Footer** created earlier this assignment and place it at the bottom of the **productpage**.

Now, implement the **ProductItem, ProductList, CartItem,** and **Cart** components.

Assembling Productpage Component:

- Inside the **src/components** directory of your React project, create a new file named **Productpage.js**.
- In the **Productpage.js** file, import the **Header, ProductList, Cart, and Footer** components to compose the structure of the productpage.
- You need to implement a functionality to save the cart items in the local storage.
- Uses the **useEffect** hook to load the cart items from local storage when the component mounts, ensuring that it's initially populated with any previously saved items. Also, whenever cart changes, make sure to save the changes in the local storage.
- Define functions to handle adding and removing items from the cart.

The structure of the Productpage will have the following structure:

```
<div className="product-page">
   <Header />
   <table>
     <tr>
       <td><ProductList /></td>
       <td style={{verticalAlign:'top'}}><Cart /></td>
     </tr>
```

```
      </table>
      <Footer />
    </div>
```

Integrating Productpage into the Project:

- In the **App.js** file, import the **Productpage** component and render the **Productpage** component within the **App** component.
- See the "**Productpage.gif**" for a visual representation of the Product Page and the functionality of hovering over the product.