Day 4 Assignment: Building Dynamic Frontend Components

Day 4 Assignment: Building Dynamic Frontend Components for My Furniture Marketplace

Objective:

The focus of this assignment is to design and develop dynamic frontend components for My Furniture Marketplace. These components dynamically display data fetched from Sanity CMS or APIs, emphasizing modularity, scalability, and responsiveness.

Key Components Developed:

1. Product Listing Component:

The product listing component displays product data dynamically in a grid layout. Each product card shows essential details like the name, price, image, and stock status. It ensures a visually appealing and user-friendly presentation.

Features:

- Dynamic rendering of product data from the API.
- Clean and responsive card layout.
- Modular design for reuse across pages.

Code:

```
import React from 'react';
```

```
function ProductCard({ name, price, image, stock }) {
    return (
        <div className="product-card">
            <img src={image} alt={name} />
            <h3>{name}</h3>
            <p>${price}</p>
            <p>{stock ? 'In Stock' : 'Out of Stock'}</p>
        </div>
    );
}

export default ProductCard;
```

---

2. Product Detail Component:

This component provides a comprehensive view of a single product. It uses dynamic routing in Next.js to render individual product pages and displays detailed information like description, price, available sizes, and colors.

Features:

- Dynamic routing using Next.js for individual product pages.
- Displays detailed product information.
- User-friendly layout with sections for sizes and colors.

Code:

```
import React from 'react';


function ProductDetail({ name, description, price, sizes, colors }) {
    return (
        <div className="product-detail">
            <h1>{name}</h1>
            <p>{description}</p>
            <p>Price: ${price}</p>
            <div>
                <h4>Available Sizes:</h4>
                <ul>
                    {sizes.map((size) => (
                        <li key={size}>{size}</li>
                    ))}
                </ul>
                <h4>Colors:</h4>
                <ul>
                    {colors.map((color) => (
                        <li key={color}>{color}</li>
                    ))}
                </ul>
            </div>
        </div>
    );
}
```

```
export default ProductDetail;
```

---

3. Category Component:

The category filter component allows users to refine their product search by selecting specific categories. It fetches category data dynamically from the backend and updates the product listing accordingly.

Features:

- Interactive category filtering.
- Dynamic fetching and rendering of category data.
- Enhances user experience by narrowing down options.

Code:

```
import React from 'react';

function CategoryFilter({ categories, onSelectCategory }) {
    return (
        <div className="category-filter">
            <h4>Categories</h4>
            <ul>
                {categories.map((category) => (
                    <li key={category} onClick={() => onSelectCategory(category)}>
                        {category}
                    </li>
```

```
                ))}
            </ul>
        </div>
    );
}

export default CategoryFilter;
```

---

4. Search Bar Component:

The search bar enables users to find products by name or tags. It uses a debounced input field to reduce unnecessary API calls and enhance performance.

Features:

- Real-time product search with debounce functionality.
- Intuitive and responsive design.
- Easily integrates with the product listing page.

Code:

```
import React, { useState } from 'react';

function SearchBar({ onSearch }) {
    const [query, setQuery] = useState('');

    const handleSearch = (e) => {
```

```
        const value = e.target.value;
        setQuery(value);
        onSearch(value);
    };

    return (
        <input
            type="text"
            value={query}
            onChange={handleSearch}
            placeholder="Search products..."
        />
    );
}

export default SearchBar;
```

---

5. Cart Component:

The cart component provides a summary of the items users have added to their cart. It dynamically updates quantities, calculates the total price, and uses state management for seamless interactivity.

Features:

- Dynamic cart item management.
- Displays quantities and total price.
- Responsive design for multiple devices.

Code:

```
import React, { useContext } from 'react';
import { CartContext } from '../contexts/CartContext';

function Cart() {
    const { cartItems, totalPrice } = useContext(CartContext);

    return (
        <div className="cart">
            <h2>Your Cart</h2>
            <ul>
                {cartItems.map((item) => (
                    <li key={item.id}>
                        {item.name} - ${item.price} x {item.quantity}
                    </li>
                ))}
            </ul>
            <p>Total: ${totalPrice}</p>
        </div>
    );
}

export default Cart;
```

---

6. Pagination Component:

This component divides large datasets into smaller pages. Users can navigate between pages using previous, next, or numbered buttons.

Features:

- Dynamic page navigation.
- Accessible and responsive design.
- Handles large datasets efficiently.

Code:

```
import React from 'react';

function Pagination({ currentPage, totalPages, onPageChange }) {
    return (
        <div className="pagination">
            <button
                onClick={() => onPageChange(currentPage - 1)}
                disabled={currentPage === 1}
            >
                Previous
            </button>
            {[...Array(totalPages)].map((_, index) => (
                <button
                    key={index}
                    className={index + 1 === currentPage ? 'active' : ''}
```

```
                    onClick={() => onPageChange(index + 1)}
                >
                    {index + 1}
                </button>
            ))}
            <button
                onClick={() => onPageChange(currentPage + 1)}
                disabled={currentPage === totalPages}
            >
                Next
            </button>
        </div>
    );
}

export default Pagination;
```