

# Day 4 - Dynamic Frontend Components for [Bandage-App E-Commerce Marketplace]

## Introduction

This report documents the entire process I followed to develop the dynamic frontend components for my e-commerce platform. The purpose of this platform is to create a seamless shopping experience for users, featuring dynamic product listings, category filtering, search functionality, and other essential e-commerce features. Below, I describe each step I took in detail, highlighting the challenges I faced and the solutions I implemented.

## Process Overview

### 1. Tags and Categories

- **What I Did:**
  - Implemented dynamic filtering based on categories and tags.
  - Created a reusable `FilterComponent` for consistent filtering functionality.
  - Used dropdowns and checkboxes for user input.
- **Challenges I Faced:**
  - Ensuring efficient API calls.
- **Solution:** Applied debouncing and caching mechanisms to optimize performance.



## Rustic Vase Set

Bring the charm of nature into your home with the Rustic Vase Set. Perfect for those who appreciate ...

**\$210**

rustic

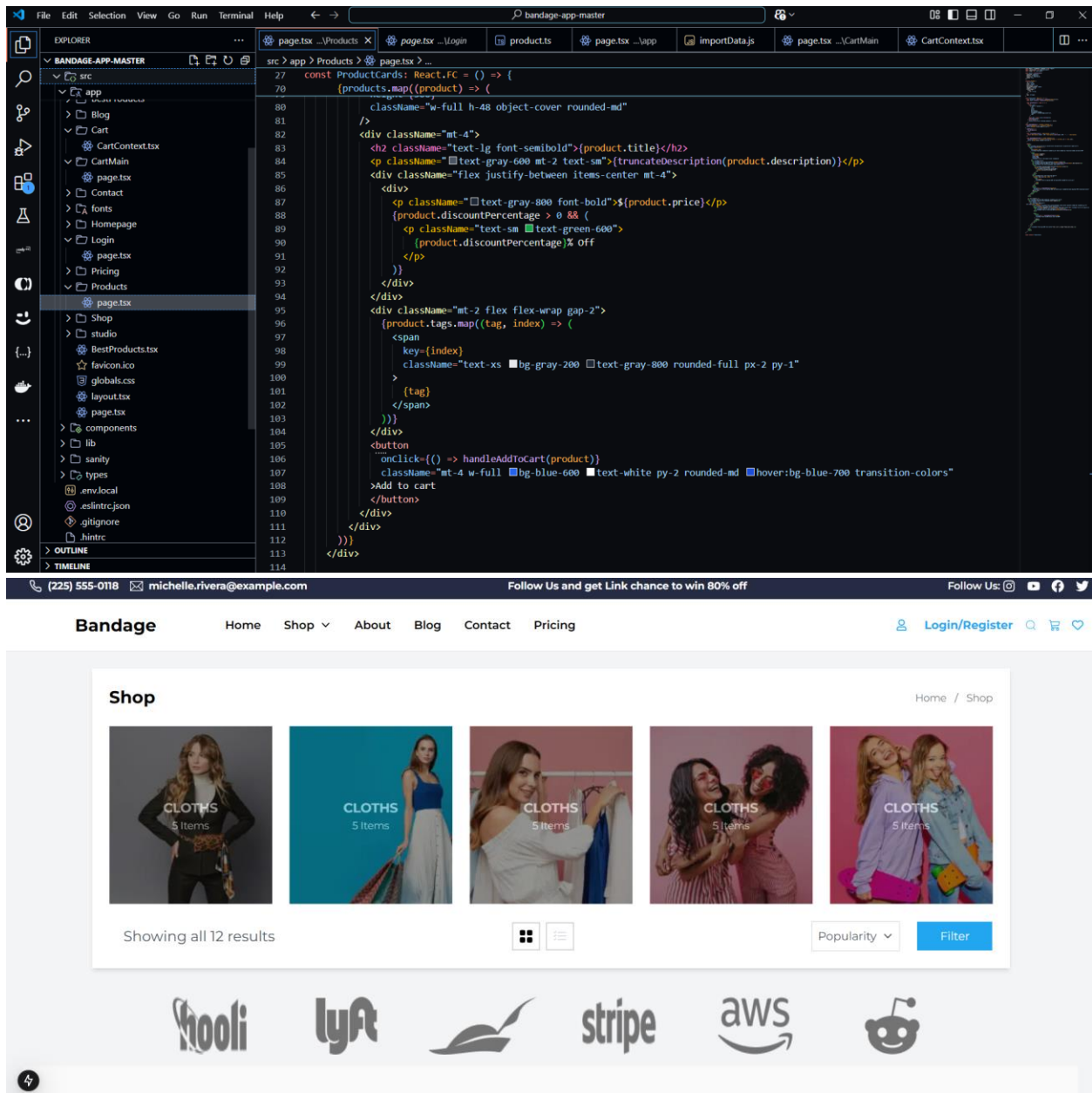
vase

home decor

vintage

interior design

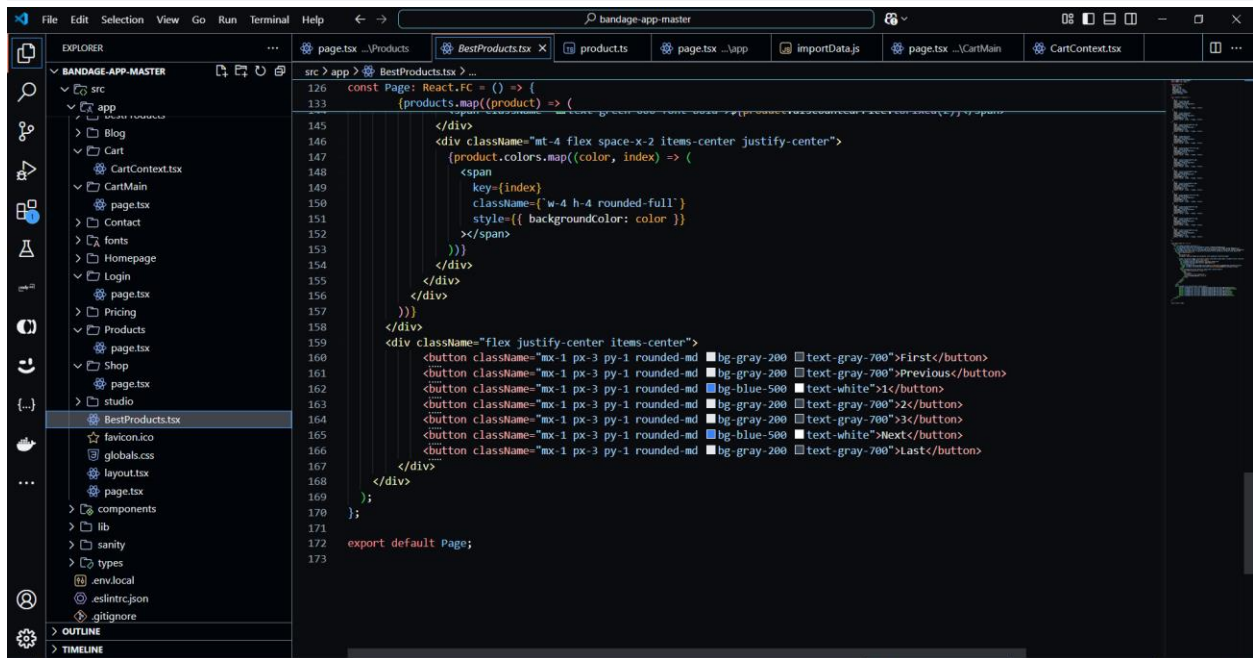
Add to cart



## 2. Pagination

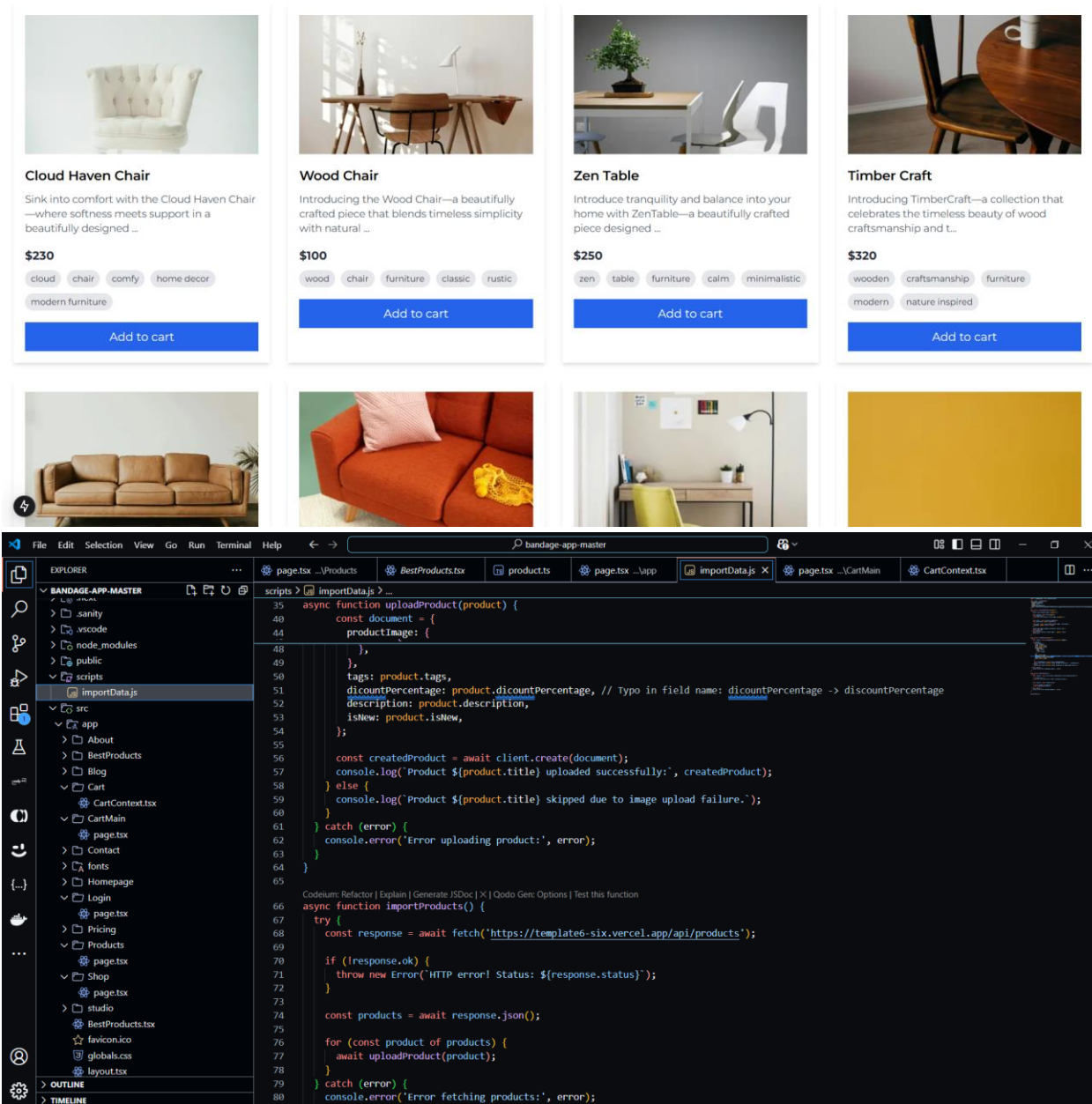
- **What I Did:**
  - Developed a `PaginationComponent` to divide products into smaller, navigable pages.
  - Used Next.js's `getStaticProps` and `getServerSideProps` for efficient data fetching.
  - Styled navigation buttons for easy use.
- **Challenges I Faced:**

- Smooth page transitions.
- **Solution:** Added loading indicators to enhance user feedback.



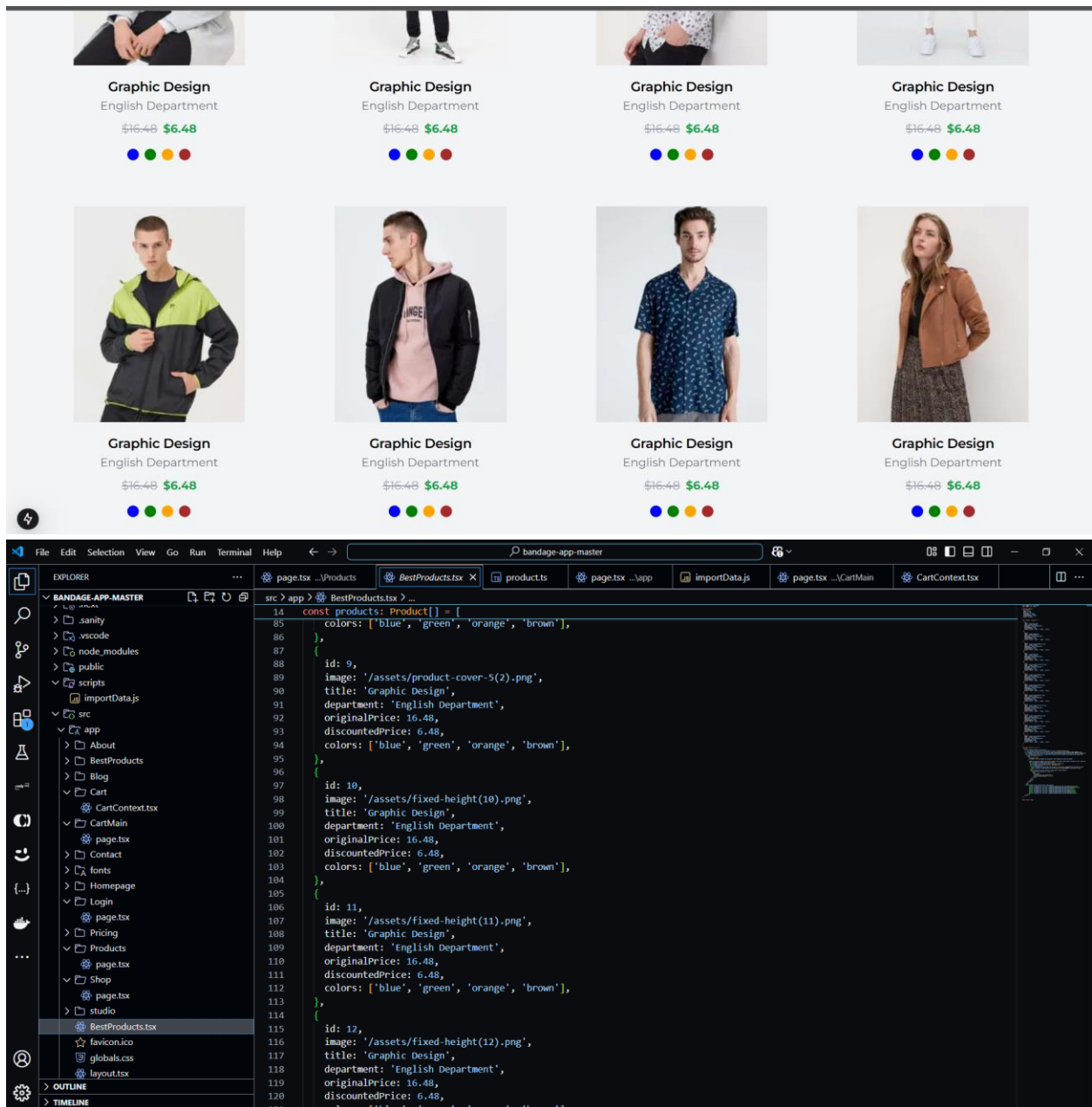
### 3. Product Card

- **What I Did:**
  - Designed a ProductCard component to display product details like name, price, stock status, and image.
  - Styled the cards using Tailwind CSS for responsiveness.
  - Added interactive buttons for “Add to Cart” and “Wishlist.”
- **Challenges I Faced:**
  - Handling inconsistent product image sizes.
- **Solution:** Used CSS object-fit properties to maintain uniformity.



## 4. Product Listing

- **What I Did:**
  - Created a ProductList component to display a grid of products fetched dynamically from the API.
  - Implemented lazy loading for improved performance.
- **Challenges I Faced:**
  - Managing large datasets.
- **Solution:** Implemented infinite scrolling for smoother user experience.



## 5. Add to Cart

- **What I Did:**
  - Developed the cart functionality using React Context API to manage the global state.
  - Enabled dynamic quantity updates for cart items.
- **Challenges I Faced:**
  - Syncing cart state across multiple components.
- **Solution:** Used local storage to persist cart data.





A photograph of a green velvet sofa. A red telephone receiver is lying on the sofa, with its cord extending towards the left. A small red object, possibly a pen or a small toy, is also on the sofa. A white spiral notebook is partially visible on the right side of the sofa. The background is dark and out of focus.

Introducing RetroVibe—a perfect blend of vintage charm and modern style, designed for those who appr...

retro vintage furniture modern decor

Add to cart

Introducing The Lucky Lamp—a unique blend of charm, elegance, and positive energy designed to illumi...

lamp lucky decor lighting vintage

Add to cart

Elevate your space with Nordic Elegance—a collection that brings the minimalist beauty and understat...

nordic elegance furniture minimalistic

Add to cart


The screenshot shows a VS Code editor with a React application. The Explorer sidebar on the left displays the file structure, with 'page.jsx' selected under the 'Products' directory. The main editor window shows the code for 'page.jsx', which is a React functional component. The code includes a 'const ProductCards: React.FC = () => {' block, followed by a conditional rendering of a cart summary if the cart is not empty. The JSX uses Tailwind CSS for styling, including a 'bg-white shadow-md rounded-lg p-4' class for the cart container. A 'handleRemoveFromCart' function is defined to remove items from the cart. The code ends with 'export default ProductCards;'.

```

src > app > Products > page.jsx > ...
27 const ProductCards: React.FC = () => {
118   <div className="text-gray font-semibold">Cart Summary</div>
119   {cart.length > 0 ? (
120     <ul className="space-y-4">
121       {cart.map((item, index) => (
122         <li key={index} className="flex justify-between items-center bg-white shadow-md rounded-lg p-4">
123           <div className="flex items-center space-x-2">
124             <img src={item.imageUrl} alt={item.title} width={50} height={50} className="w-14 h-14 object-cover rounded-full" />
125             <h3 className="text-sm font-semibold">{item.title}</h3>
126             <p className="text-xs text-gray-500">${item.price.toFixed(2)}</p>
127           </div>
128           <div>
129             <button
130               onClick={() => handleRemoveFromCart(item)}
131               className="text-red-600 hover:text-red-800"
132             >
133               Remove
134             </button>
135           </div>
136         </li>
137       )})
138     </ul>
139   ) : (
140     <p className="text-gray-600 text-center">Your cart is empty Please add items.</p>
141   )}
142 </div>
143 </main>
144 );
145 };
146
147 export default ProductCards;
148

```


### Cart Summary



Serene Seat

\$350.00


Remove



Rustic Vase Set

\$210.00


Remove



The Lucky Lamp

\$200.00

Remove



Retro Vibe

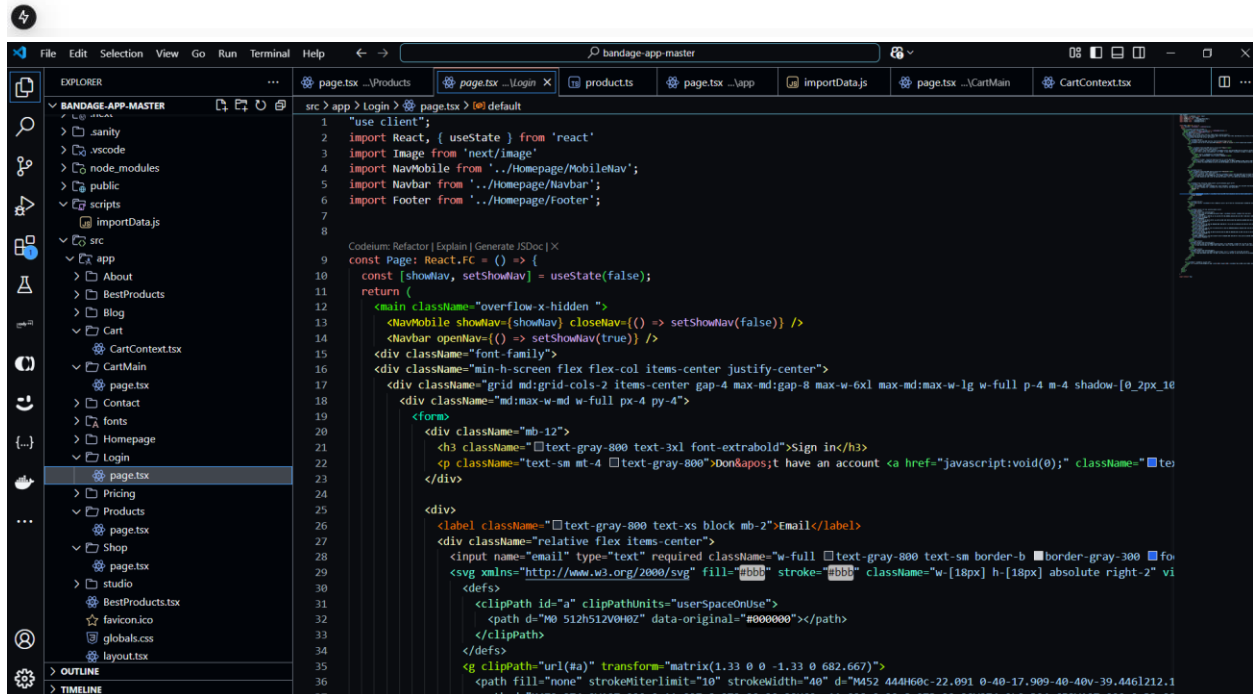
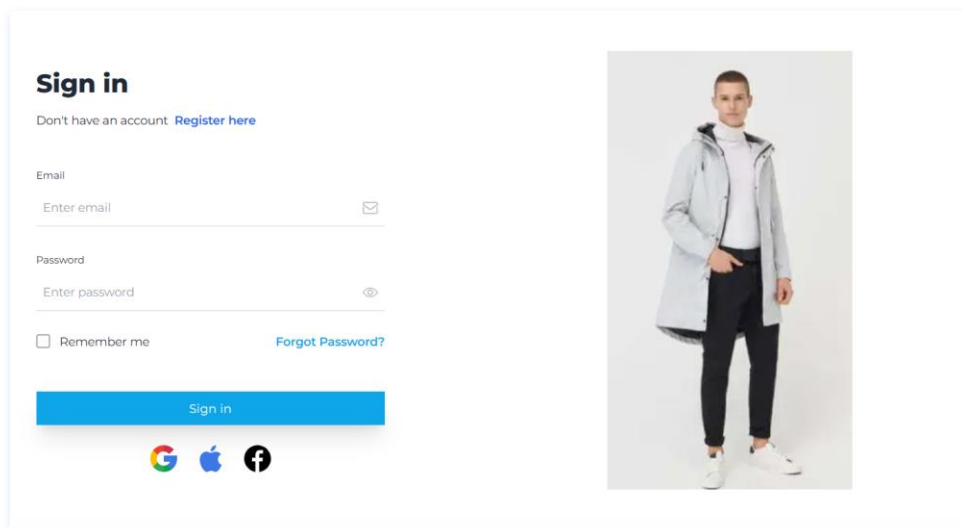
\$340.00

Remove



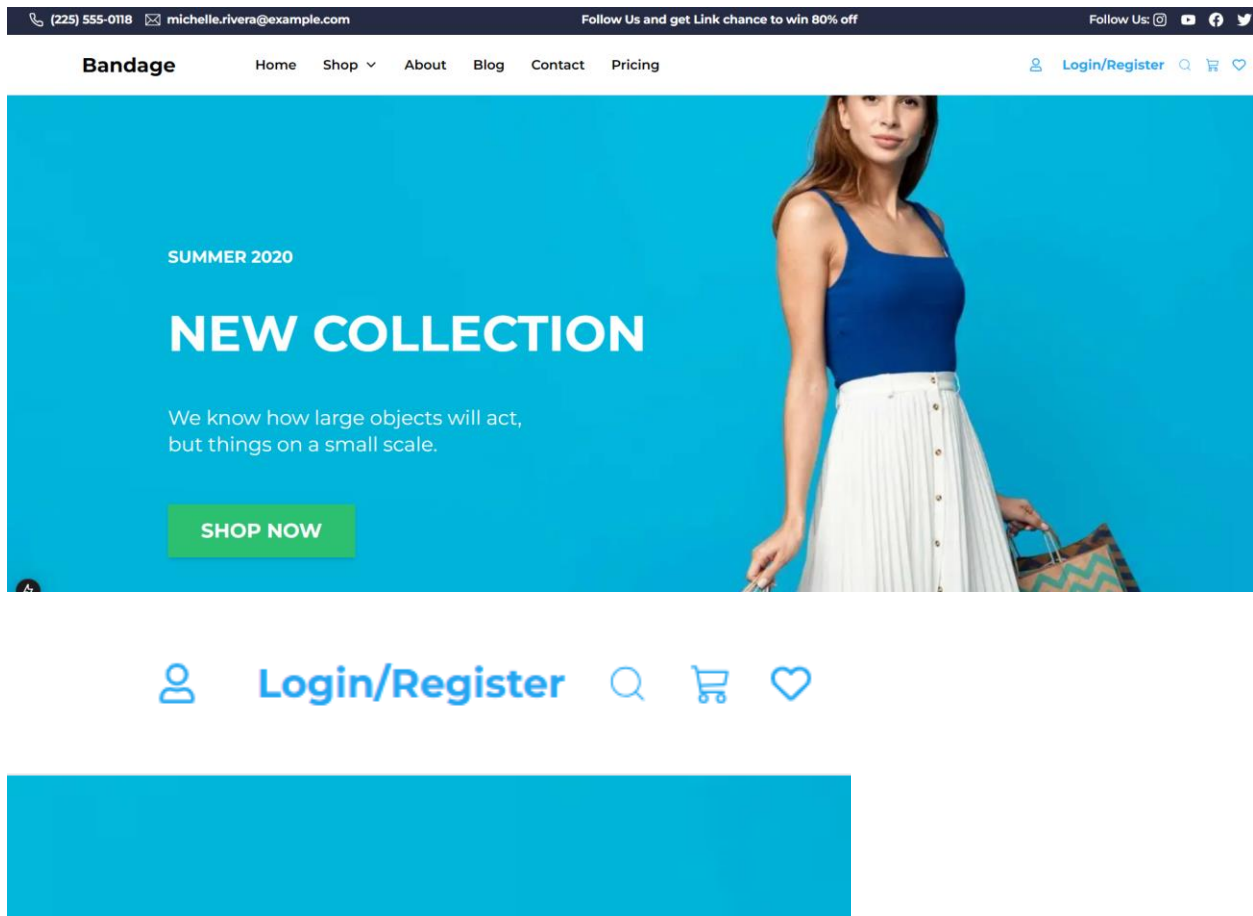
## 6. Login and Sign-Up Page

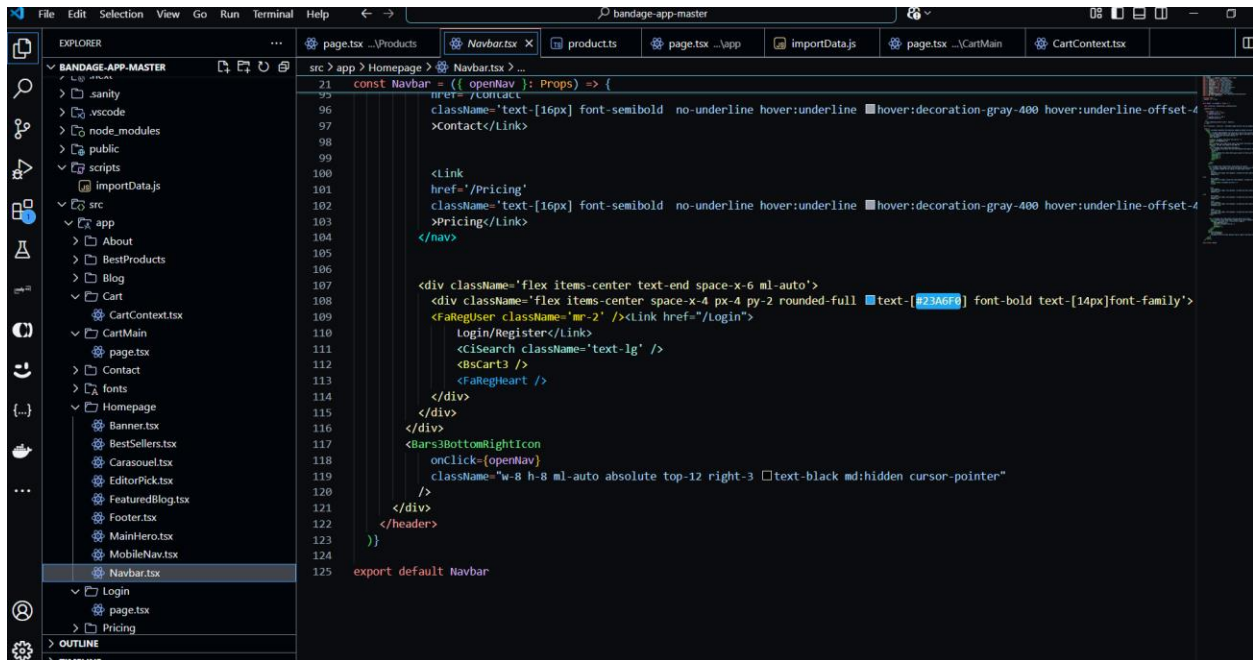
- **What I Did:**
  - Built user authentication pages for login and sign-up with form validation.
  - Integrated Firebase for secure authentication and user management.
- **Challenges I Faced:**
  - Handling errors during login and sign-up.
- **Solution:** Provided user-friendly error messages for better feedback.



## 7. Search Bar Component

- **What I Did:**
  - Developed a SearchBar component for searching products by name or tags.
  - Integrated API calls to fetch search results in real time.
- **Challenges I Faced:**
  - Ensuring case-insensitive searches.
- **Solution:** Normalized search input and data using JavaScript string methods.





## Achievements

By the end of the development process, I successfully completed:

1. A fully functional product listing page displaying dynamic data from the API.
2. Individual product detail pages implemented with dynamic routing.
3. Advanced filters for categories and tags.
4. A search bar for efficient product discovery.
5. Pagination for better user experience with large datasets.
6. Responsive and professional styling for all components.
7. Modular and reusable components for future scalability.