Report on Dynamic Frontend Components for Your Marketplace

Introduction

The website Hacthon2 is an online marketplace featuring products categorized for user convenience. The platform provides functionalities such as product listings, product details, cart management, and notifications, ensuring a smooth shopping experience.

This report outlines the key frontend components implemented in the project and specifies where each provided snippet should be integrated within the codebase.

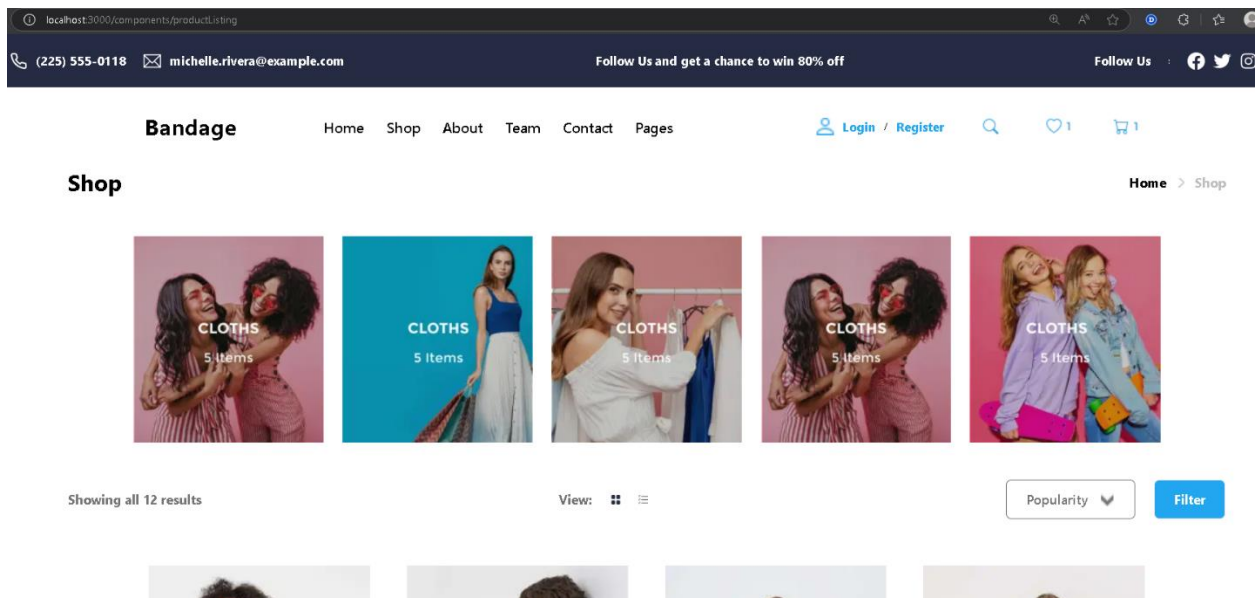# Frontend Components & Snippet Integration

1. Navbar
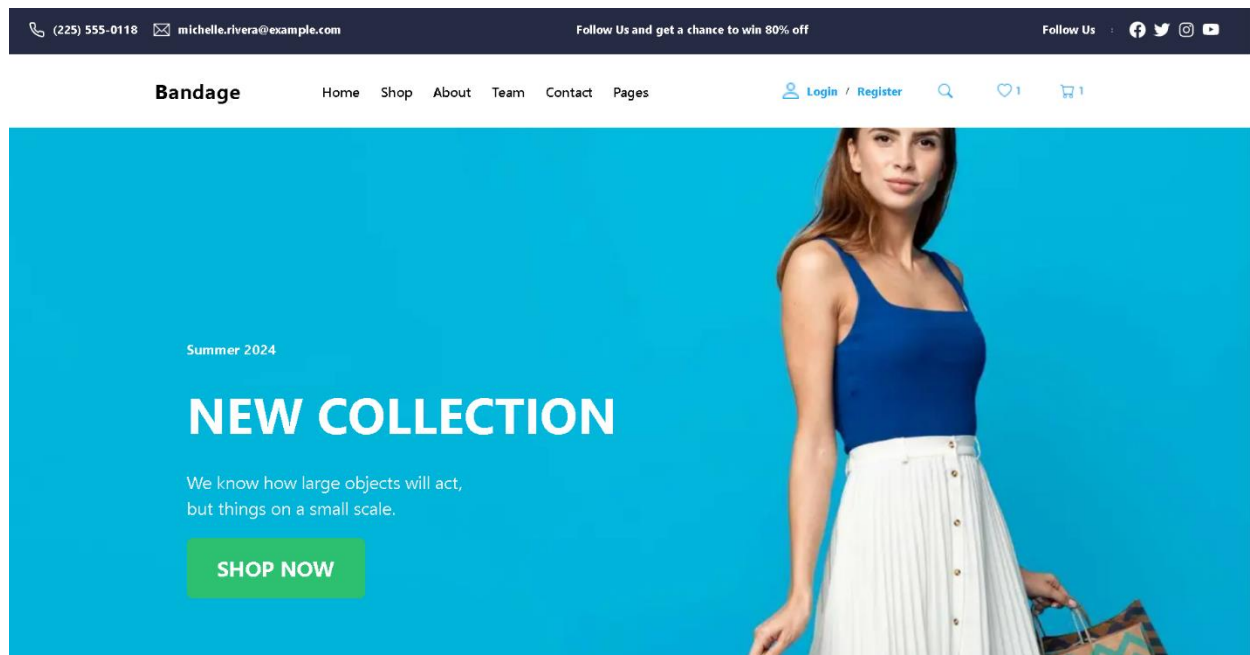
The Navbar component provides site-wide navigation.

Snippet to attach: Navbar

Location: Should be placed inside App.js or the Layout component to ensure it appears on all pages.

2. Categories

This component displays the product categories.

Snippet to attach: Categories

Location: Used in the homepage and product listing page for filtering products.

3. Categories Code

Contains logic to dynamically fetch and display categories.

Snippet to attach: Categories code

Location: Inside the Categories component.

4. React Hooks (useEffect, useState, etc.)

```
my-app > src > app > products > page.tsx > ProductCards
1   "use client"
2   import React, { useEffect, useState } from "react"
3   import sanityClient, { createClient } from "@sanity/client"
4   import Image from "next/image"
5   import { apiVersion, dataset, projectId } from "@/sanity/env"
6
7   const sanity = createClient({
8       projectId: "zgf109um",
9       dataset: "production",
10      apiVersion: "2023-01-01",
11      useCdn: true,
```

This snippet includes hooks used for state management and API requests.

Snippet to attach: ReactHooks

Location: Across multiple components such as Products listing, Cart, and Notifications.

5. Products Listing
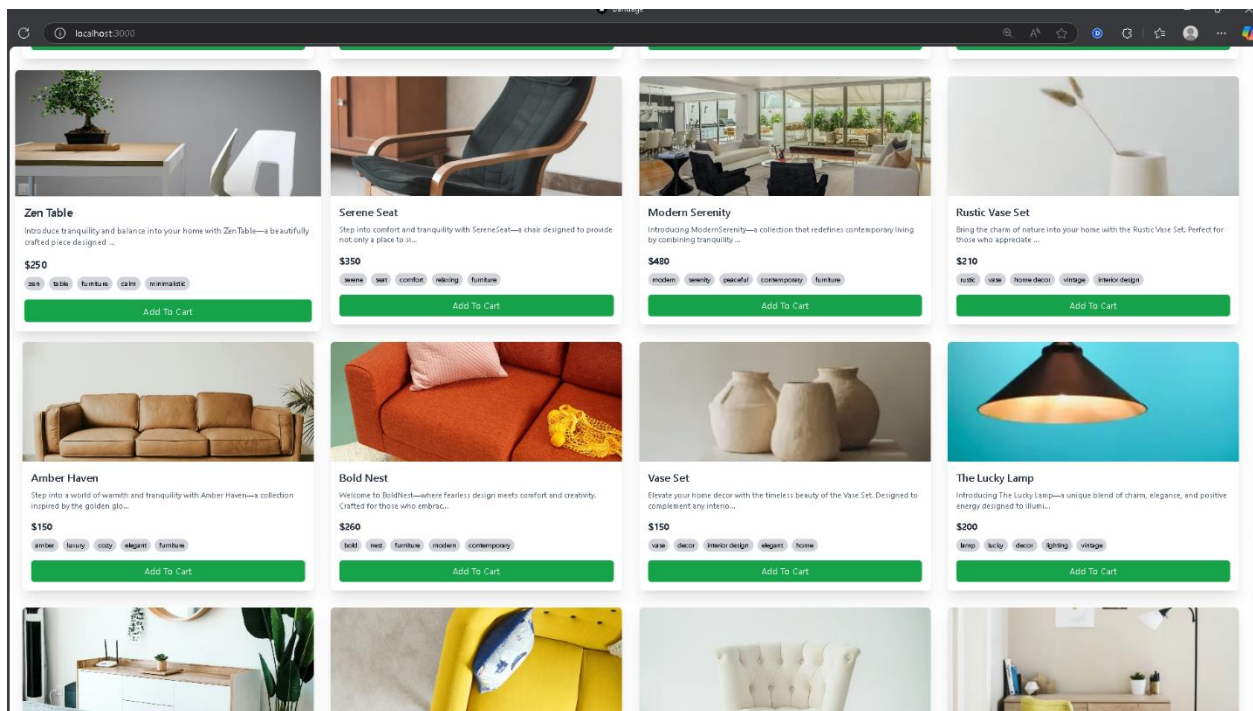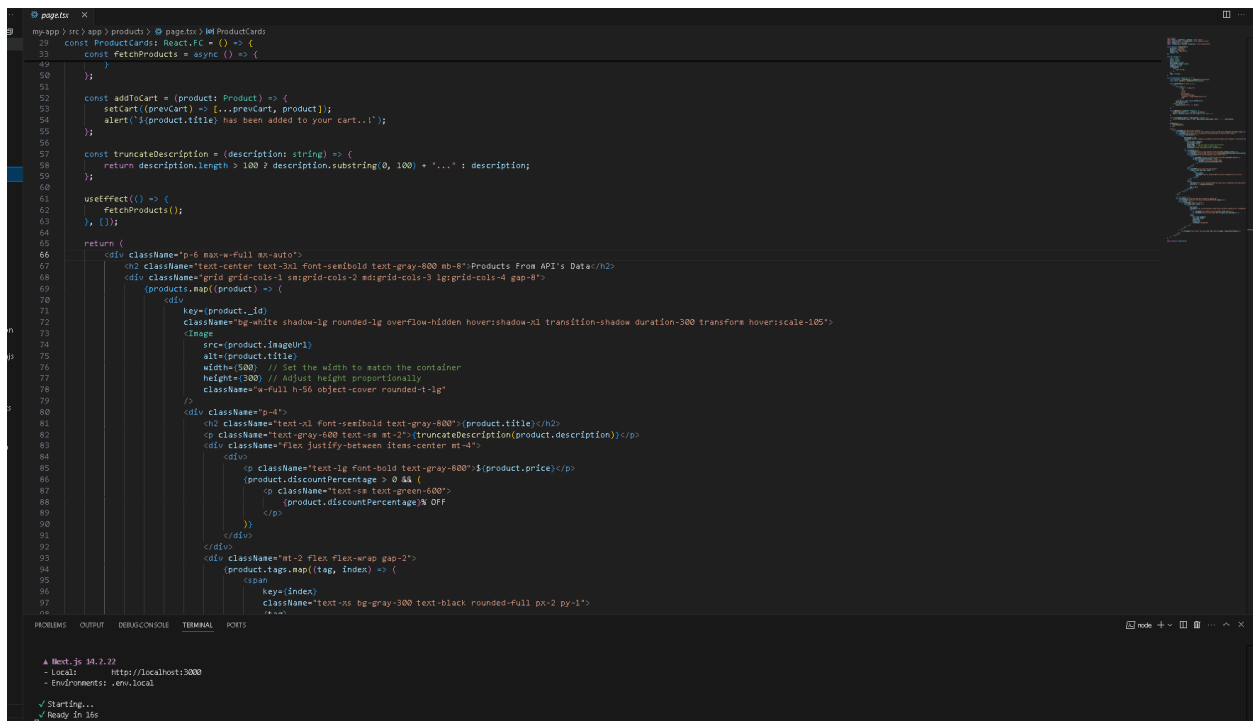
Displays all available products on the homepage and category pages.

Snippet to attach: Products listing

Location: Used in Home.js or a dedicated Products.js page.

6. Product Listing Code

Fetches and structures product data dynamically.

Snippet to attach: Product listing code

Location: Inside the Products listing component.

7. Product
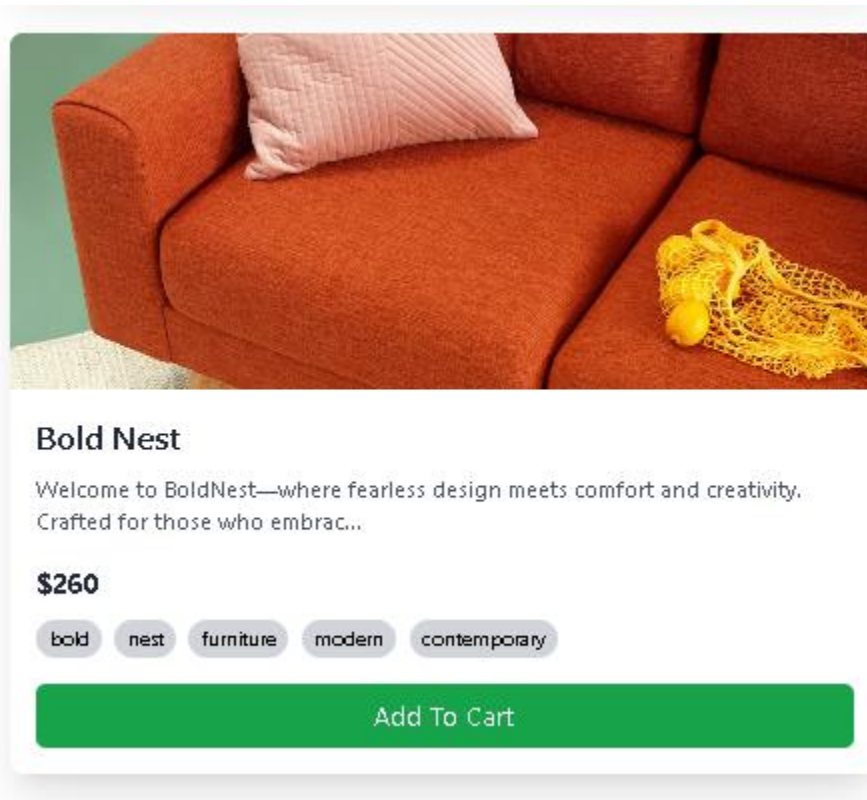
Represents an individual product displayed in the listing.

Snippet to attach: Product

Location: Used inside Products listing as a reusable component.

8. Product Code

```
                </div>
                <div className="mt-2 flex flex-wrap gap-2">
                    {product.tags.map((tag, index) => (
                        <span
                            key={index}
                            className="text-xs bg-gray-300 text-black rounded-full px-2 py--1">
                            {tag}
                        </span>
                    ))}
                </div>
                <button
                    className="mt-4 w-full bg-green-600 text-white py-2 rounded-md hover:bg-green-700"
                    onClick={() => addToCart(product)}
                >
                    Add To Cart
                </button>
            </div>
        </div>
    ))}
</div>

{/* Cart Summary */}
<div className="mt-8 bg-gray-100 p-6 rounded-lg shadow-lg">
    <h2 className="text-2xl font-bold text-gray-800">Cart Summary</h2>
    {cart.length > 0 ? (
        <ul className="space-y-4 mt-4">
            {cart.map((item, index) => (
                <li
                    key={index}
                    className="flex justify-between items-center bg-white shadow-sm p-4 rounded-md">
                    <div>
```

Contains logic for displaying individual product details.

Snippet to attach: Product code

Location: Inside the Product component.

9. Cart
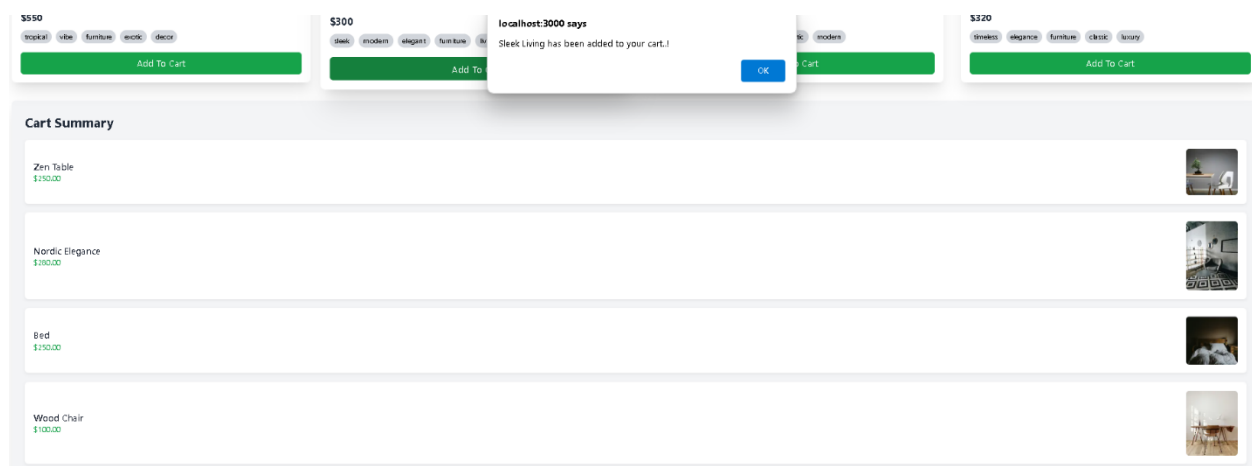
Manages and displays items added by users.

Snippet to attach: Cart

Location: Implemented in a Cart.js page or a sidebar component.

10. Notification

localhost:3000 says
Marble Ease has been added to your cart..!
OK

Displays messages such as "Item added to cart."

Snippet to attach: Notification

Location: Used globally in App.js or as a Toast component.

11. Notification Code

```tsx
    page.tsx     ✕

my-app > src > app > products > 🌐 page.tsx > [∅] ProductCards
  29   const ProductCards: React.FC = () => {
  33       const fetchProducts = async () => {
  35             const query =
  44                 }`;
  45             const data = await sanity.fetch(query);
  46             setProducts(data);
  47         } catch (error) {
  48             console.error("Error..!", error);
  49         }
  50       };
  51
  52       const addToCart = (product: Product) => {
  53         setCart((prevCart) => [...prevCart, product]);
  54         alert(`${product.title} has been added to your cart..!`);
  55       };
  56
  57       const truncateDescription = (description: string) => {
  58         return description.length > 100 ? description.substring(0, 100) + "..." : description;
  59       };
  60
  61       useEffect(() => {
  62         fetchProducts();
```

Contains logic to handle different types of notifications.

Snippet to attach: Notification code

Location: Inside the Notification component.

Pages Overview

1. Homepage

Displays the product listing with categories and a search bar.

Uses Navbar, Categories, Products listing, and Notification.

2. Product Detail Page

Displays a single product's information.

Uses Product and Product code.

3. Cart Page

Shows items added by users.

Uses Cart and Notification.

Conclusion

This report provides the necessary integration points for each snippet in the project. Following these placements will ensure that the website functions smoothly with a modular and maintainable codebase.