

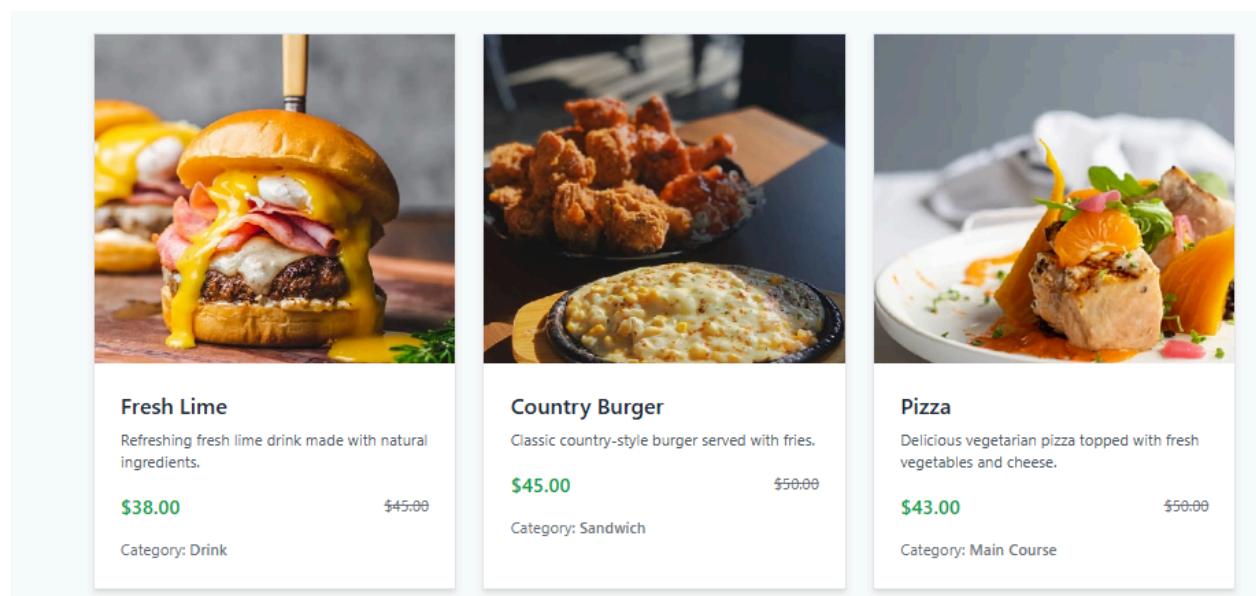
Hackathon 3

DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE

1. Product Listing Component: Render product data dynamically in a grid layout.

PREPARED BY AMEEN ALAM 4 DAY 4
BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE

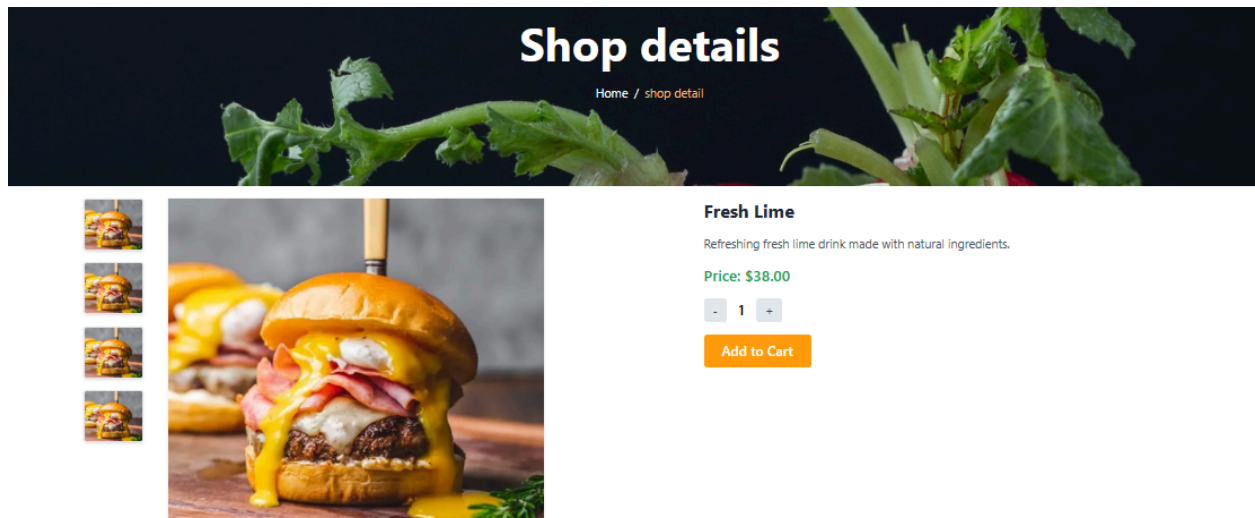
- o Product Name
- o Price
- o Image
- o Stock Status



2. Product Detail Component:

Create individual product detail pages using dynamic routing in Next.js. Include detailed fields such as:


- o Product Description
- o Price




3. Search Bar:

Implement search functionality to filter products by name or tags


Food List



Fresh Lime
Refreshing fresh lime drink made with natural ingredients.
\$38.00 ~~\$45.00~~
Category: Drink




Country Burger
Classic country-style burger served with fries.
\$45.00 ~~\$50.00~~
Category: Sandwich



Pizza
Delicious vegetarian pizza topped with fresh vegetables and cheese.
\$43.00 ~~\$50.00~~
Category: Main Course

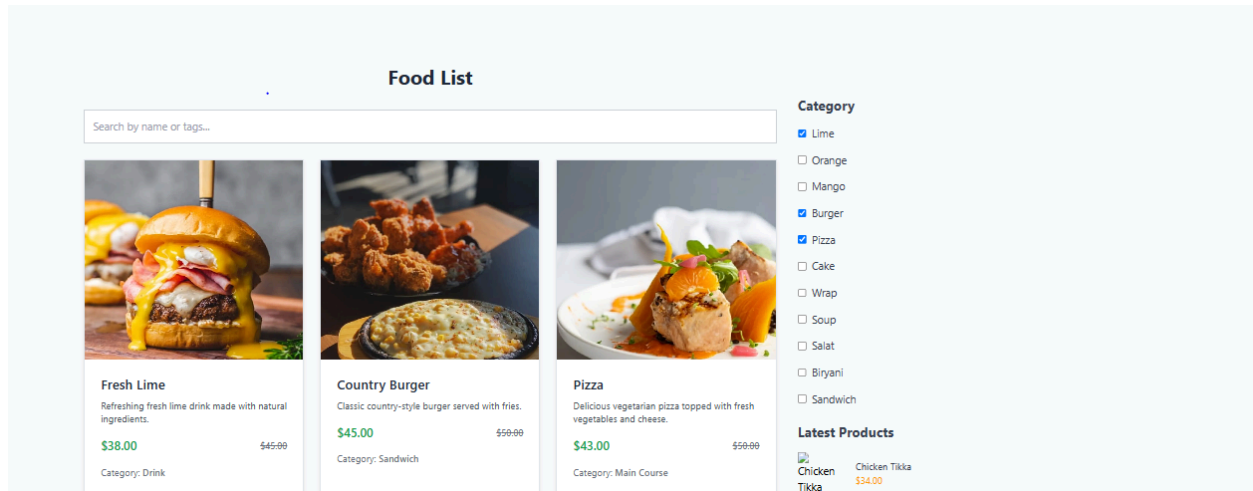
Food List



Chicken Chup
Crispy fried chicken bites served with dipping sauce.
\$12.00 ~~\$45.00~~

4: Categories:

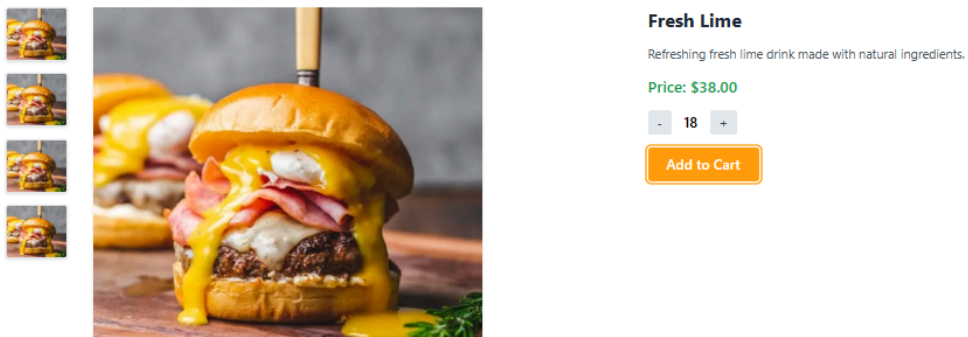
Enable filtering of products by selected categories



5. Cart Component:


Display added items, quantity, and total price.

Use state management for tracking cart items



```
src/components/ShopPages/ShopDetail/ShopDetail.tsx
4
5 Codeium: Refactor | Explain
6 type CartItem = {
7   id: string; // Unique identifier for the item
8   name: string; // Name of the product
9   price: number; // Price as number (no need for string format)
10  quantity: number; // Quantity of the item
11  rating: number; // Product rating (e.g., 1-5 stars)
12  image: string; // Thumbnail image URL
13  largeImage: string; // Main large image URL for detailed view
14  status: string; // Availability status ("In Stock" or "Out of Stock")
15 };
16
17 Codeium: Refactor | Explain
18 type CartContextType = {
19   cart: CartItem[]; // Array of cart items
20   addToCart: (item: CartItem) => void; // Add item to the cart
21   removeItem: (id: string) => void; // Remove item from the cart
22   updateItemQuantity: (id: string, quantity: number) => void; // Update the quantity of an item
23   clearCart: () => void; // Clear all items from the cart
24   calculateTotalPrice: () => number; // Function to calculate the total price of all items in the cart
25 };
26
27 const CartContext = createContext<CartContextType | undefined>(undefined);
28
29 Codeium: Refactor | Explain | Generate JSDoc | X
30 export const CartProvider = ({ children }: { children: ReactNode }) => {
31   const [cart, setCart] = useState<CartItem[]>([]);
32
33   // Add item to the cart
34   Codeium: Refactor | Explain | X
35   const addToCart = (item: CartItem) => {
36     setCart((prevCart) => {
37       const existingItem = prevCart.find((cartItem) => cartItem.id === item.id);
38       if (existingItem) {
39         return prevCart.map((cartItem) =>
```

Checkout Flow Component:



Chocolate Muffin

Price: \$28

-

22

+

Total: \$616.00

Remove

Grand Total: \$616.00

Checkout

Coupon Code

Use coupon code **DISCOUNT10** for a \$10 discount on your order.

Enter Coupon Code

Apply

Total Bill

Cart Subtotal:	\$616.00
Shipping Charge:	\$10.00
Total Amount:	\$626.00

Proceed to Checkout

CheckSquareOffset
Icon

Final Thoughts: Laying the Foundation for a Scalable E-Commerce Platform

By the end of Day 4, we have successfully laid the groundwork for a future-ready, feature-rich Q-commerce platform. Our focus has been on creating a seamless, real-time shopping experience by integrating essential marketplace components. Here's what we have accomplished so far:

1. **Dynamic Product Listing**

We have developed a fully functional product listing page that dynamically fetches data from Sanity CMS or APIs. This ensures that customers always have access to the latest offerings with real-time updates.

2. **Dedicated Product Detail Pages**

Each product now has its own dedicated page, dynamically generated using Next.js routing. These pages are optimized for SEO and provide in-depth details, including product descriptions, pricing, and customizable options like sizes and colors—all managed via Sanity CMS.

3. **Advanced Category Filtering**

We have implemented an intuitive category filtering system, allowing users to refine their searches effortlessly. The categories are dynamically managed from Sanity, making it easy to scale and adapt the platform.

4. **Powerful Search Functionality**

Our newly integrated search bar enables users to find products quickly using names or tags. With real-time search suggestions, the feature enhances discoverability and ensures smooth navigation, even with a vast product catalog.

5. **Responsive & Aesthetic Design**

Every component has been designed with responsiveness in mind, ensuring a visually appealing and fully functional interface across various devices. The platform maintains a professional look and seamless accessibility, regardless of how users interact with it.

The Road Ahead

This foundational work sets the stage for a dynamic and scalable e-commerce experience. With Next.js and Sanity CMS at the core, we are well-positioned to refine the user experience, optimize performance, and introduce more advanced features in the coming days. The journey towards building a robust, user-centric Q-commerce platform has just begun!
