Market Blace Builder Hackathon 2025

Day 4 Report

Building Dynamic Frontend Components

Project Name: “Foodtuck: A Q-Commerce Marketplace”

Date: 19/01/2025

1. Objective

The goal of Day 4 was to create and integrate dynamic frontend components for the website, ensuring features are reusable, responsive, and user-friendly. The primary focus was on the **Food** and **Chef** pages.

2. Components Built

a) Navbar

- **Description**: A responsive navigation bar for easy access to the Home, Food, and Chefs pages.
- **Features**:
  - Links to Home (/), Food (/food), and Chefs (/chefs).
  - Highlighted styling for hover effects.

b) Food Page

- **Description**: A dynamic listing of food items fetched from the Sanity CMS.
- **Features**:
  - **Search Bar**: Allows users to search food items by name dynamically.
  - **Category Filter**: Filters food items based on selected categories (e.g., Burger, Pizza).
  - **Pagination**: Displays 6 items per page, with "Previous" and "Next" buttons.

c) Chefs Page

- **Description**: A listing page showing all chefs fetched from the Sanity CMS.
- **Features**:
    - Displays chef names, specialties, and profile images.
    - Links to individual chef detail pages.

d) ProductCard Component

- **Description**: A reusable component to display food or chef data.
- **Features**:
    - Displays name, category/specialty, price, and images.
    - Includes a clickable image for navigating to detailed pages.

e) Food Detail Page

- **Description**: Shows detailed information about a selected food item.
- **Features**:
    - Displays name, category, description, and price.
    - Quantity selector and "Add to Cart" button.

f) Chef Detail Page

- **Description**: Shows detailed information about a selected chef.
- **Features**:
    - Displays chef name, specialty, and bio.
    - Includes responsive design for a great user experience.

3. Code Snippets of Key Components

a) Navbar Component

Navbar.tsx

src > app > components > Navbar.tsx > Navbar

```tsx
import Link from 'next/link';

export default function Navbar() {
  return (
    <nav className="bg-gray-800 text-white shadow-lg">
      <div className="max-w-7xl mx-auto px-4 py-3 flex justify-between items-center">
        <h1 className="text-2xl font-bold">
          <Link href="/">FoodTuck</Link>
        </h1>
        <div className="space-x-4">
          <Link href="/" className="hover:text-yellow-400">Home</Link>
          <Link href="/food" className="hover:text-yellow-400">Food</Link>
          <Link href="/chefs" className="hover:text-yellow-400">Chefs</Link>
          <Link href="/cart" className="hover:text-yellow-400">Cart</Link>
        </div>
      </div>
    </nav>
  );
}

```

b) SearchBar Component

SearchBar.tsx

src > app > components > SearchBar.tsx > ...

```tsx
export default function SearchBar({ onSearch }: { onSearch: (query: string) => void }) {
  return (
    <input
      type="text"
      placeholder="Search for food..."
      onChange={(e) => onSearch(e.target.value)}
      className="w-full p-2 border border-gray-300 rounded"
    />
  );
}

```

c) ProductCard Component

```tsx
import { urlFor } from '@/sanity/lib/client';
import Link from 'next/link';

export default function ProductCard({ food }: { food: any }) {
  return (
```

```
    <div className="bg-white shadow-md rounded-lg overflow-hidden">
      {/* Wrap the image in a Link */}
      <Link href={`/food/${food._id}`}>
        {food.image && (
          <img
            src={urlFor(food.image).url()}
            alt={food.name}
            className="w-full h-90 object-cover cursor-pointer hover:opacity-90 transition"
          />
        )}
      </Link>
      <div className="p-4">
        <h2 className="text-xl font-semibold">{food.name}</h2>
        <p className="text-gray-500">{food.category}</p>
        <p className="text-green-600 font-bold">${food.price.toFixed(2)}</p>
      </div>
    </div>
  );
}
```

d) Food Page (Main Logic)

```
'use client';

import { useState, useEffect } from 'react';
import Navbar from '../components/Navbar';
import ProductCard from '../components/ProductCard';
import SearchBar from '../components/SearchBar';
import client from '@/sanity/lib/client';

export default function FoodPage() {
  const [foods, setFoods] = useState<any[]>([]); // State to store all food items
  const [selectedCategory, setSelectedCategory] = useState<string | null>(null); // Category filter
  const [searchQuery, setSearchQuery] = useState<string>(''); // Search filter
  const [currentPage, setCurrentPage] = useState(1); // Pagination
  const itemsPerPage = 6; // Items per page

  // Fetch food data from Sanity CMS
  useEffect(() => {
    const fetchFoods = async () => {
      const data = await client.fetch('*[_type == "food"]');
      setFoods(data);
    };
    fetchFoods();
  }, []);

  // Filtered food items based on category and search query
  const filteredFoods = foods.filter((food) => {
    const matchesCategory = selectedCategory
      ? food.category?.toLowerCase() === selectedCategory.toLowerCase()
      : true;
    const matchesSearch = searchQuery
      ? food.name.toLowerCase().includes(searchQuery.toLowerCase())
      : true;
    return matchesCategory && matchesSearch;
  });
```

```
// Paginated food items
const startIndex = (currentPage - 1) * itemsPerPage;
const paginatedFoods = filteredFoods.slice(startIndex, startIndex + itemsPerPage);

// Total pages for pagination
const totalPages = Math.ceil(filteredFoods.length / itemsPerPage);

return (
  <div className="min-h-screen bg-gray-100">
    <Navbar />
    <div className="max-w-7xl mx-auto py-10 px-5">
      <h1 className="text-4xl font-bold text-gray-800 mb-8">Food Items</h1>

      {/* Search Bar */}
      <div className="mb-4">
        <SearchBar onSearch={(query) => setSearchQuery(query)} />
      </div>

      {/* Category Filter */}
      <div className="mb-8">
        <select
          value={selectedCategory || ''}
          onChange={(e) => setSelectedCategory(e.target.value || null)}
          className="p-2 border border-gray-300 rounded"
        >
          <option value="">All Categories</option>
          <option value="Burger">Burger</option>
          <option value="Pizza">Pizza</option>
          <option value="Drink">Drink</option>
          <option value="Dessert">Dessert</option>
          {/* Add more categories as needed */}
        </select>
      </div>

      {/* Food Grid */}
      <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-8">
        {paginatedFoods.map((food: any) => (
          <ProductCard key={food._id} food={food} />
        ))}
      </div>

      {/* Pagination */}
      <div className="flex justify-center mt-6">
        <button
          onClick={() => setCurrentPage((prev) => Math.max(prev - 1, 1))}
          disabled={currentPage === 1}
          className="px-4 py-2 bg-gray-300 text-gray-700 rounded hover:bg-gray-400 disabled:opacity-50"
        >
          Previous
        </button>
        <span className="mx-4 my-3">
          Page {currentPage} of {totalPages}
        </span>
        <button
          onClick={() => setCurrentPage((prev) => Math.min(prev + 1, totalPages))}
          disabled={currentPage === totalPages}
          className="px-4 py-2 bg-gray-300 text-gray-700 rounded hover:bg-gray-400 disabled:opacity-50"
```

```
          >
            Next
          </button>
        </div>
      </div>
    </div>
  );
}
```

e) Food Detail Page

```
'use client';

import { useState } from 'react';
import { useCart } from '../../context/CartContext';
import { urlFor } from '@/sanity/lib/client';

export default function FoodDetail({ food }: { food: any }) {
  const { addToCart } = useCart();
  const [quantity, setQuantity] = useState(1);

  return (
    <div className="max-w-4xl mx-auto py-10 px-5">
      <div className="bg-white shadow-md rounded-lg overflow-hidden">
        {food.image && (
          <img
            src={urlFor(food.image).url()}
            alt={food.name}
            className="w-full h-96 object-cover"
          />
        )}
        <div className="p-6">
          <h1 className="text-3xl font-bold">{food.name}</h1>
          <p className="text-gray-500">{food.category}</p>
          <p className="text-green-600 text-2xl font-bold">${food.price.toFixed(2)}</p>
          {food.description && (
            <p className="mt-4 text-gray-700">{food.description}</p>
          )}

          {/* Quantity Selector */}
          <div className="flex items-center mt-6">
            <label className="mr-3">Quantity:</label>
            <input
              type="number"
              value={quantity}
              onChange={(e) =>
                setQuantity(Math.max(1, parseInt(e.target.value) || 1))
              }
              className="w-16 p-2 border border-gray-300 rounded"
            />
          </div>

          {/* Add to Cart */}
          <button
            onClick={() => addToCart({ ...food, quantity })}
            className="mt-6 bg-yellow-500 text-white px-6 py-3 rounded hover:bg-yellow-600"
```

```
          >
            Add to Cart
          </button>
        </div>
      </div>
    </div>
  );
}
```

f) Chef Page

```
import Navbar from '../components/Navbar';
import client, { urlFor } from '@/sanity/lib/client';
export default async function ChefsPage() {
  const chefs = await client.fetch('*[_type == "chef"]');

  return (
    <div className="min-h-screen bg-gray-100">
      <Navbar />
      <div className="max-w-7xl mx-auto py-10 px-5">
        <h1 className="text-4xl font-bold text-gray-800 mb-8">Our Chefs</h1>
        <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-8">
          {chefs.map((chef: any) => (
            <div
              key={chef._id}
              className="bg-white shadow-md rounded-lg overflow-hidden"
            >
              {chef.image && (
                <img
                  src={urlFor(chef.image).url()}
                  alt={chef.name}
                  className="w-full h-62 object-cover"
                />
              )}
              <div className="p-5">
                <h2 className="text-xl font-semibold">{chef.name}</h2>
                <p className="text-gray-500">{chef.specialty}</p>
                <a
                  href={`/chefs/${chef._id}`}
                  className="text-blue-500 hover:underline mt-3 block"
                >
                  View Details
                </a>
              </div>
            </div>
          ))}
        </div>
      </div>
    </div>
  );
}
```

g) Chef Detail Page

```
import Navbar from '../../components/Navbar';
import client, { urlFor } from '@/sanity/lib/client';

export default async function ChefDetailPage({ params }: { params: { id: string } }) {
  const chef = await client.fetch(`*[_type == "chef" && _id == "${params.id}"][0]`);

  if (!chef) {
    return <div>Chef not found</div>;
  }

  return (
    <div className="min-h-screen bg-gray-100">
      <Navbar />
      <div className="max-w-4xl mx-auto py-10 px-5">
        <div className="bg-white shadow-md rounded-lg overflow-hidden">
          {chef.image && (
            <img
              src={urlFor(chef.image).url()}
              alt={chef.name}
              className="w-full h-62 object-cover"
            />
          )}
          <div className="p-6">
          <h2 className="text-xl font-semibold">{chef.name}</h2>
            <p className="text-green-600 font-bold">{chef.position}</p>
            <p className="text-green-600 font-bold mt-1">{chef.specialty}</p>
            <p className="text-sm text-gray-700 font-bold mt-1">Experience: {chef.experience} years</p>
            <p className="text-sm text-gray-700 mt-2">{chef.description}</p>
          </div>
        </div>
      </div>
    </div>
  );
}
```

h) Cart Page

```
'use client';

import Navbar from '../components/Navbar';
import { useCart } from '../context/CartContext';

export default function CartPage() {
  const { cart, removeFromCart, clearCart } = useCart();

  const totalPrice = cart.reduce((acc: any, item: { price: any; }) => acc + item.price, 0);

  return (
    <div className="min-h-screen bg-gray-100">
      <Navbar />
      <div className="max-w-7xl mx-auto py-10 px-5">
        <h1 className="text-4xl font-bold text-gray-800 mb-8">Your Cart</h1>
```

```
        {cart.length > 0 ? (
          <>
            <ul>
              {cart.map((item: any) => (
                <li key={item._id} className="flex items-center justify-between bg-white p-4 shadow mb-4
rounded-lg">
                  <div>
                    <h2 className="text-lg font-semibold">{item.name}</h2>
                    <p>${item.price.toFixed(2)}</p>
                  </div>
                  <button
                    onClick={() => removeFromCart(item._id)}
                    className="bg-red-500 text-white px-4 py-2 rounded hover:bg-red-600"
                  >
                    Remove
                  </button>
                </li>
              ))}
            </ul>
            <div className="mt-6">
              <h2 className="text-2xl font-bold">Total: ${totalPrice.toFixed(2)}</h2>
              <button
                onClick={clearCart}
                className="mt-3 bg-red-500 text-white px-4 py-2 rounded hover:bg-red-600"
              >
                Clear Cart
              </button>
            </div>
          </>
        ) : (
          <p>Your cart is empty.</p>
        )}
      </div>
    </div>
  );
}
```

i) Context/CartContext.tsx

```
'use client';

import { createContext, useContext, useState } from 'react';

const CartContext = createContext<any>(null);

export function CartProvider({ children }: { children: React.ReactNode }) {
  const [cart, setCart] = useState<any[]>([]);

  const addToCart = (item: any) => {
    setCart((prev) => [...prev, item]);
  };

  const removeFromCart = (id: string) => {
    setCart((prev) => prev.filter((item) => item._id !== id));
  };
```

```
  const clearCart = () => {
    setCart([]);
  };

  return (
    <CartContext.Provider value={{ cart, addToCart, removeFromCart, clearCart }}>
      {children}
    </CartContext.Provider>
  );
}

export const useCart = () => useContext(CartContext);
```

4. Challenges Faced

1. **Search Bar**:
    a. Initially, the search bar was not functional. This was resolved by passing a callback function to filter and render food items dynamically.
2. **Pagination**:
    a. Adjusting the logic to ensure seamless navigation across pages.
3. **Dynamic Routing**:
    a. Implementing and testing detail pages for both food and chefs.

5. Conclusion

All tasks for Day 4 were completed successfully:

- Dynamic components were created and integrated into the website.
- Features like search, filter, pagination, and dynamic routing were implemented and tested.
- Both food and chef data were displayed effectively with a professional and responsive UI.

Prepared by: Mahnoor M. Ayub