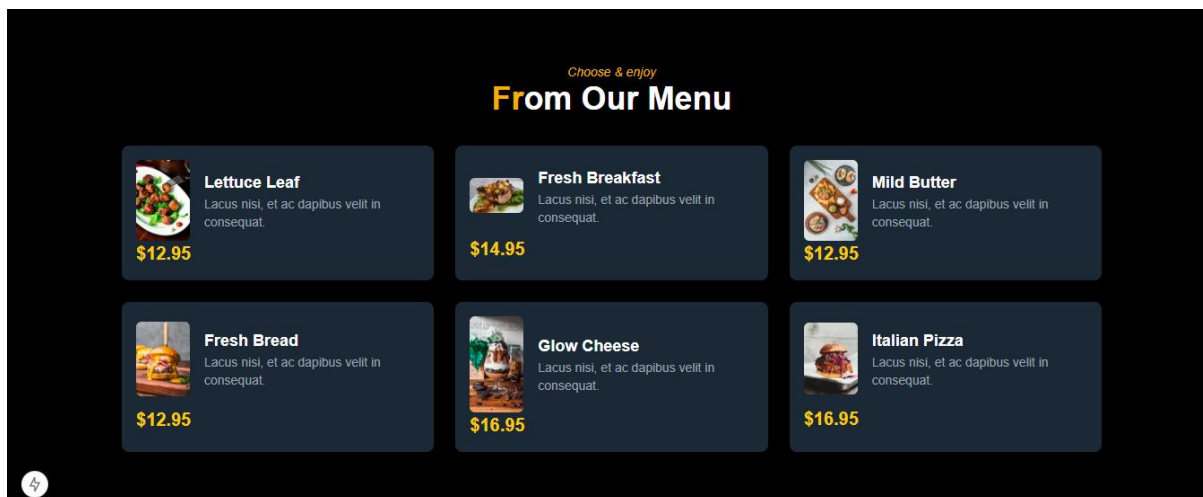
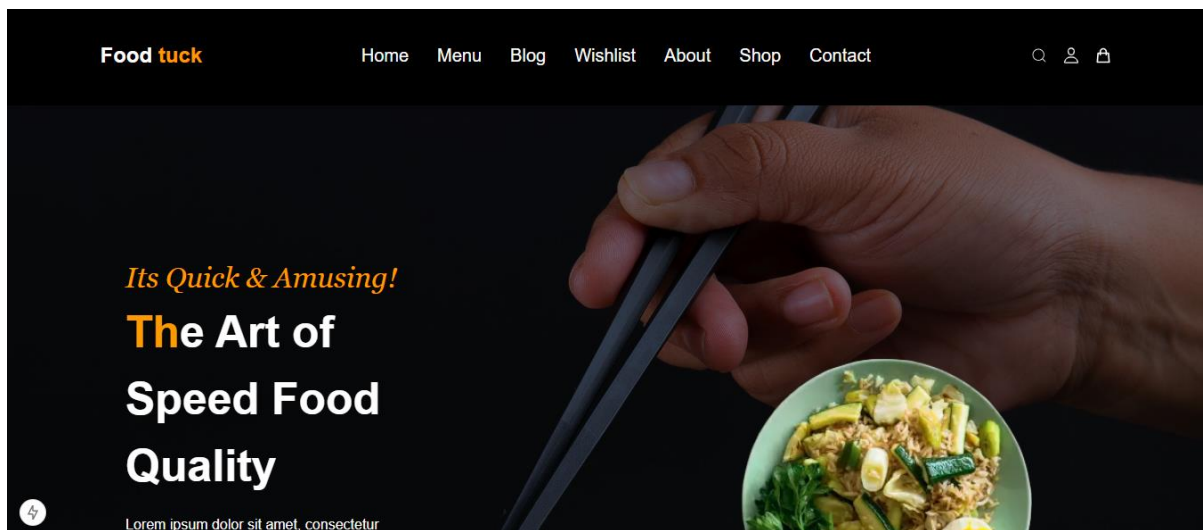
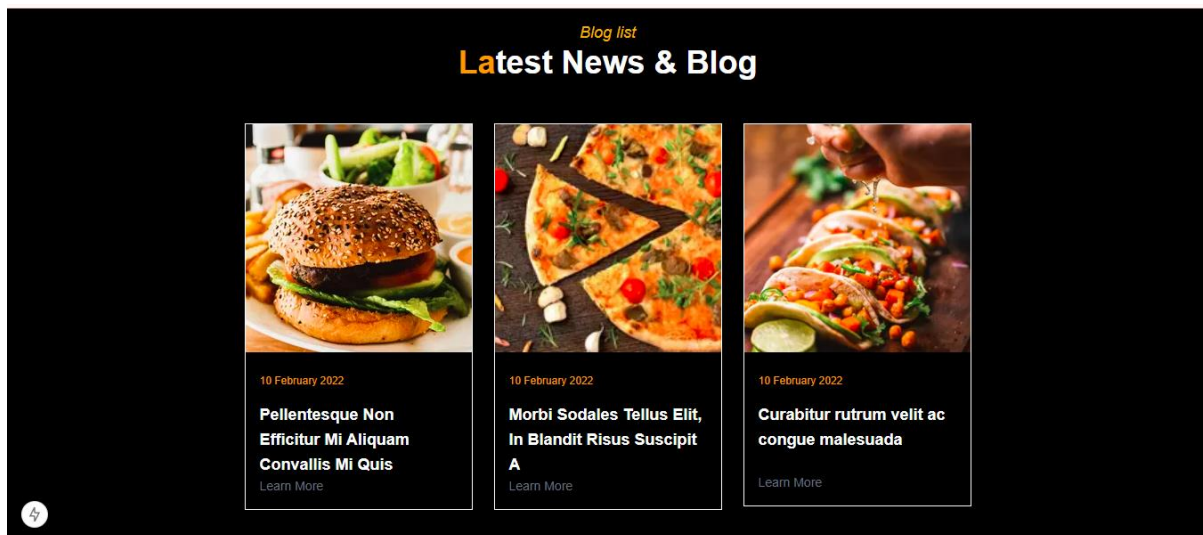


Day 4

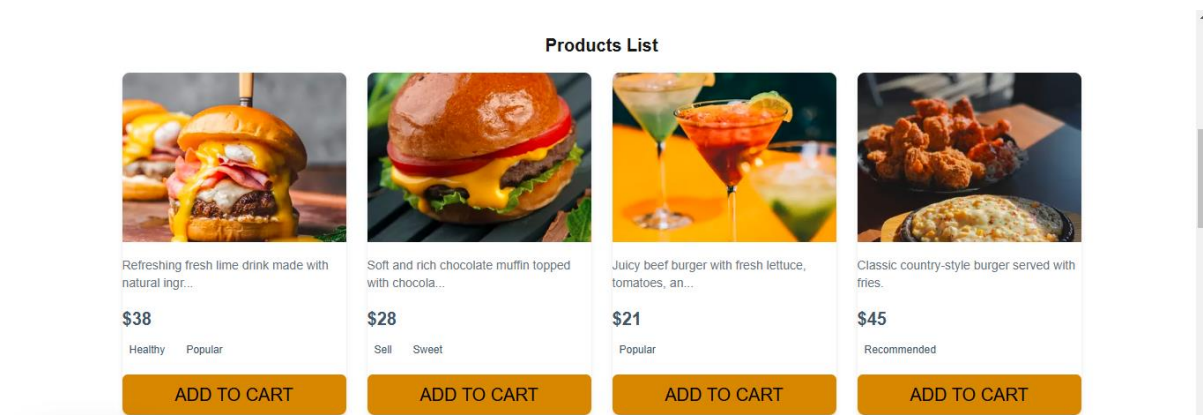
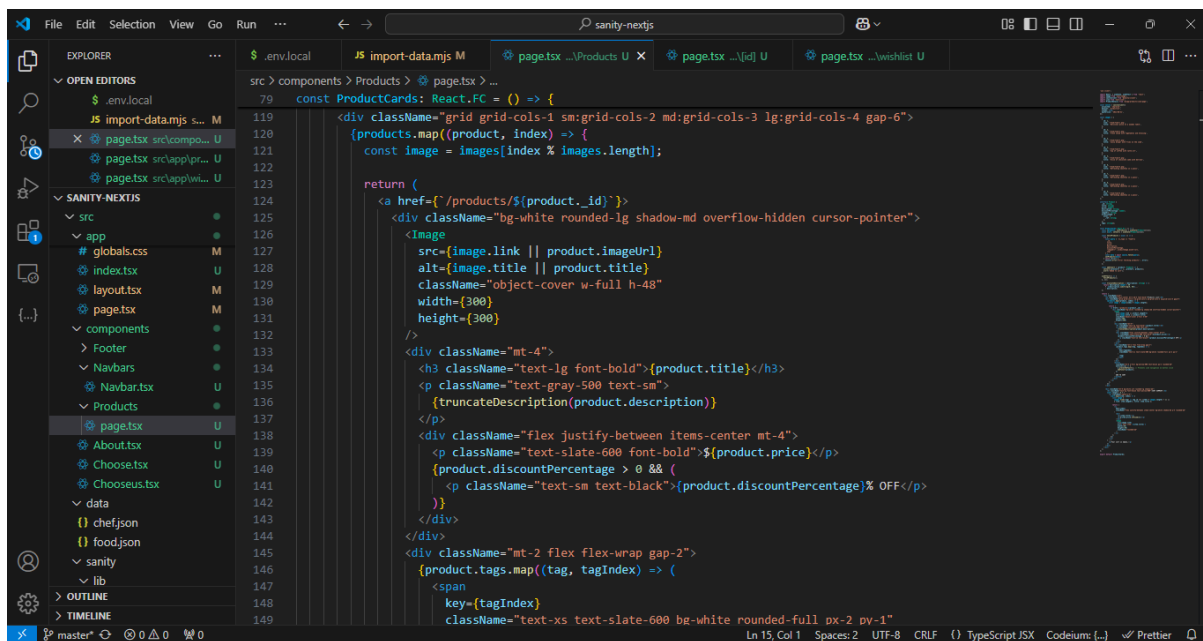
Dynamic Frontend Components

Frontend Data:



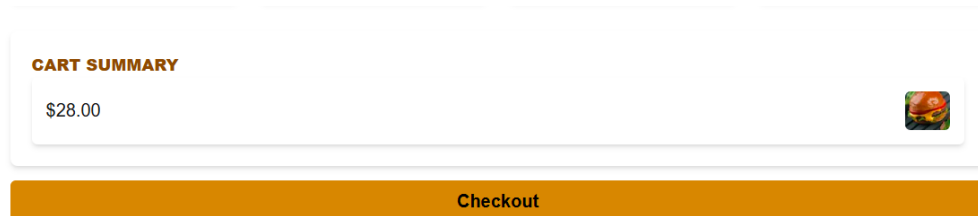


Product Listing:




ADD TO CART:

```
page.tsx U x  Chef.tsx U  NavBar.tsx U  Choose.tsx U  Cards.tsx U
src > components > Products > page.tsx > ...
33  const ProductCards: React.FC = () => {
129
130    <div className="mt-8 bg-white p-6 rounded-lg shadow-md">
131      <h2 className="text-lg font-black text-yellow-800">CART SUMMARY</h2>
132      {cart.length > 0 ? (
133        <ul className="space-y-4">
134          {cart.map((item, index) => {
135
136            return (
137              <li
138                key={index}
139                className="flex justify-between items-center bg-white shadow-md p-4 rounded-md"
140              >
141                <div>
142                  <p>{item.title}</p>
143                  <p>${item.price.toFixed(2)}</p>
144                </div>
145                <Image
146                  src={item.imageUrl}
147                  alt={item.title}
148                  width={50}
149                  height={50}
150                  className="rounded-md"
151                />
152              </li>
153            );
154          })}
155        </ul>
156      ) : (
157        <p>Your cart is empty.</p>
158      )}
159    </div>
```



Product Details (DYNAMIC ROUTING):

```
File Edit Selection View Go Run ...
sanity-nextjs
EXPLORER
src
  app
    menu
      Drinks.tsx
      Hero.tsx
      Maincourse.tsx
      page.tsx
      Starter.tsx
    products
      [id]
        page.tsx
      shopdetails
      shoplist
      Hero.tsx
      page.tsx
      Product.tsx
      shopping
      Cart.tsx
      Hero.tsx
      page.tsx
      signin
      signup
      studio
      [...tool]
      page.tsx
      wishlist
  OUTLINE
  TIMELINE
page.tsx
src > app > products > [id] > page.tsx > ...
27
Codeium: Refactor | Explain | Generate.JSDoc | X
const ProductDetails = ({ params }: { params: { id: string } }) => {
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
useEffect(() => {
  Codeium: Refactor | Explain | Generate.JSDoc | X
  const fetchProduct = async () => {
    try {
      const query = `[_type == "food" && _id == ${id}][0]{
        _id,
        title,
        price,
        description,
        discountPercentage,
        "imageUrl": image.asset->url,
        tags
      }`;
      const data = await sanity.fetch(query, { id: params.id });
      setProduct(data);
    } catch (error) {
      console.error("Error fetching product:", error);
    }
  };
  fetchProduct();
}, [params.id]);
if (!product) {
  return <p>Loading...</p>;
}
```



FOOD TRUCK

★★★★★

Soft and rich chocolate muffin topped with chocolate chips.

Tags:

Sell Sweet


\$28

Add to Cart

Food tuck

Home Menu Blog Wishlist About Shop Contact

🔍 👤 🛒



FOOD TRUCK

★★★★★

Juicy beef burger with fresh lettuce, tomatoes, and cheese.

Tags:


Popular

\$21

Add to Cart

Food truck

HomeMenuBlogWishlistAboutShopContact



FOOD TRUCK

★★★★★

Juicy beef burger with fresh lettuce, tomatoes, and cheese.

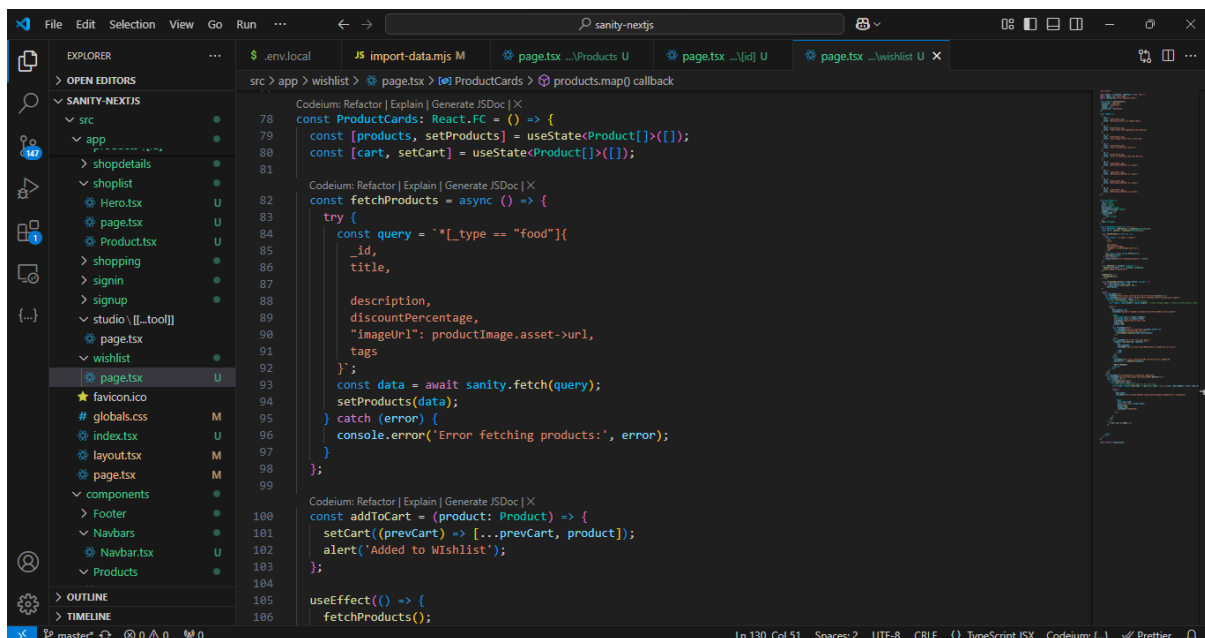
Tags:

Popular


\$21

Add to Cart

ADD TO WISHLIST:




Wishlist




Refreshing fresh lime drink made with natural ingredients.
Healthy Popular

Add to Wishlist




Soft and rich chocolate muffin topped with chocolate chips.
Salty Sweet

Add to Wishlist



Juicy beef burger with fresh lettuce, tomatoes, and cheese.
Popular

Add to Wishlist



Classic country-style burger served with fries.
Recommended

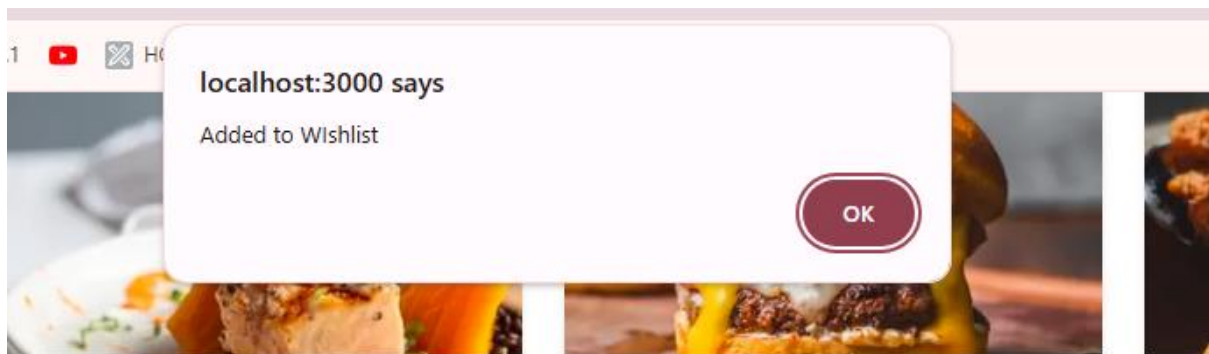
Add to Wishlist

WISHLIST

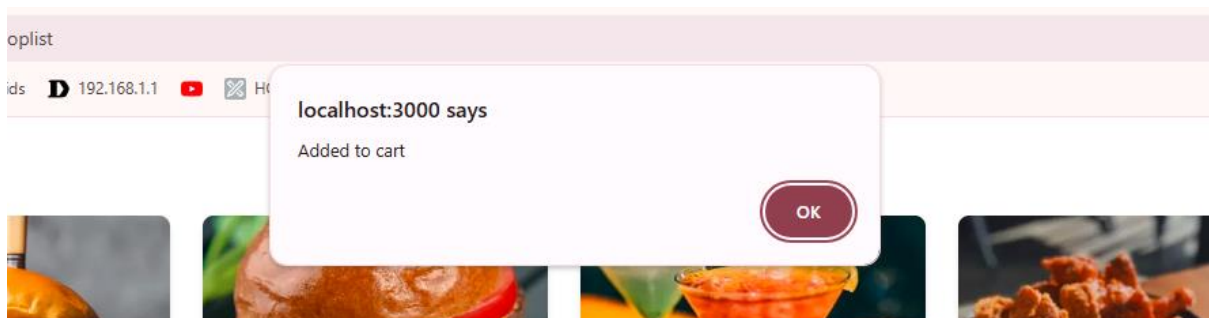


NOTIFICATIONS:

```
</div>
<button
  className="mt-4 w-full bg-yellow-600 text-black font-bold py-2 rounded-md"
  onClick={() => addToCart(product)}
>
  ADD TO WISHLIST
</button>
```



```
</div>
<button
  className="mt-4 w-full bg-yellow-600 text-black font-bold py-2 rounded-md"
  onClick={(e) => {
    e.preventDefault(); // Prevents Link navigation on button click
    addToCart(product);
  }}
>
  ADD TO CART
</button>
```



INTRODUCTION:

🔗 **Dynamic Content Management:** *The integration with Sanity CMS allows content to be created, updated, and deleted dynamically, with changes reflected instantly on the front end.*

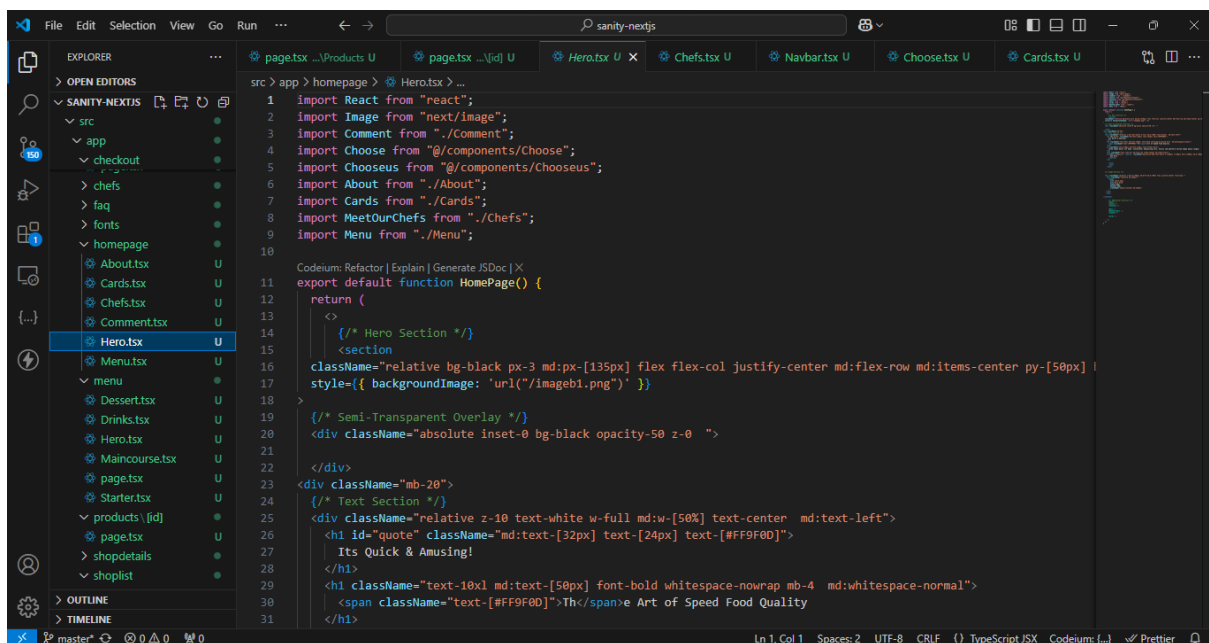
🔗 **Responsive Design:** *Built with modern design principles, the application is fully responsive, providing a seamless experience across devices.*

🔗 **Next.js Framework:** *Leverages server-side rendering (SSR) and static site generation (SSG) to deliver optimal performance and SEO benefits.*

🔗 **Image Handling:** *Displays high-quality images dynamically, with fallback options for missing content.*

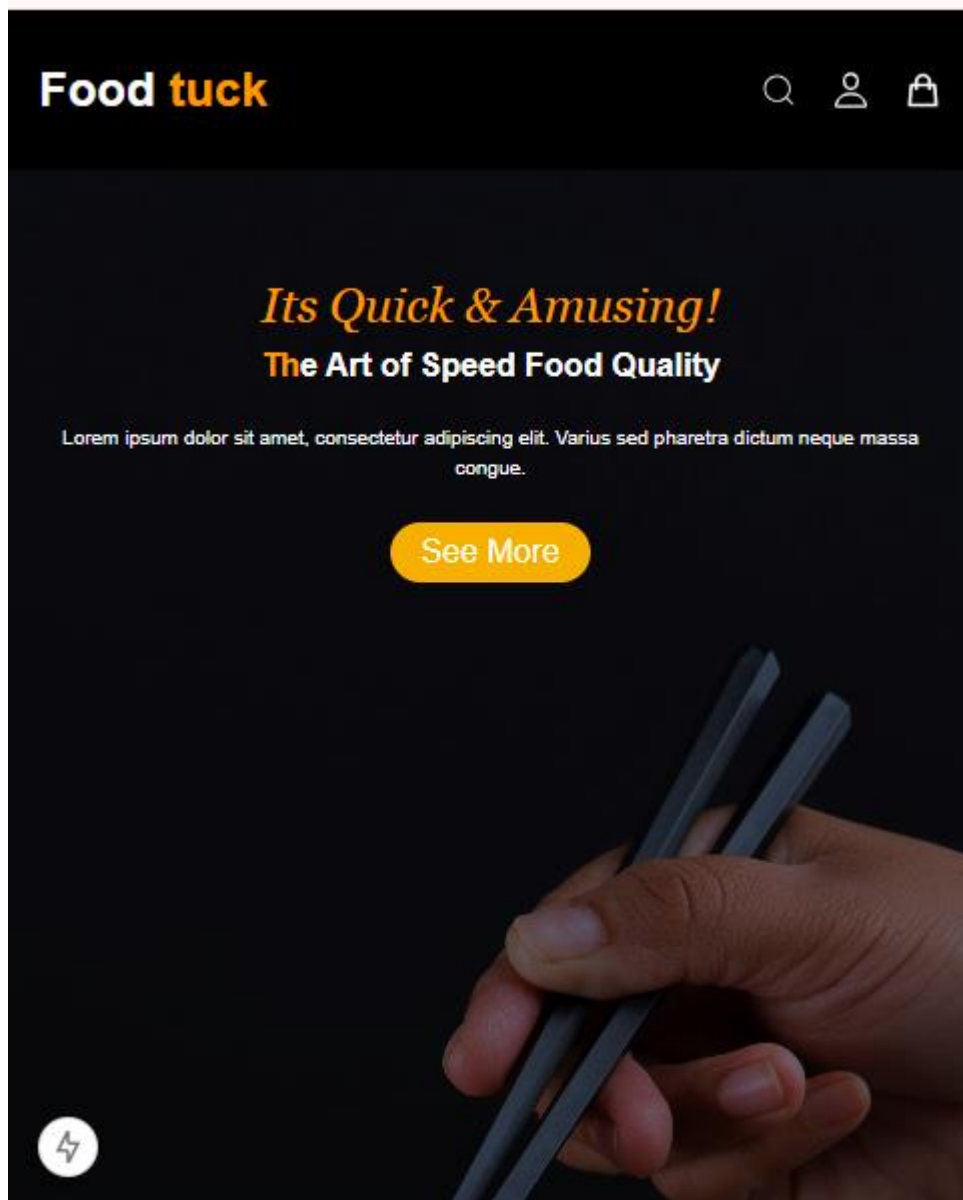
🔗 **Customizable UI:** *Designed with flexibility in mind, the user interface can easily be adjusted to fit the specific needs of the project.*

🔗 **Scalable Backend:** *Uses Sanity's GROQ query language to fetch data efficiently, ensuring the backend is scalable for future enhancements.*



The screenshot shows a VS Code editor interface with a dark theme. On the left, the Explorer sidebar displays a project structure for 'sanity-nextjs'. The 'src' directory is expanded, showing subdirectories like 'app', 'checkout', 'chefs', 'faq', 'fonts', 'homepage', 'menu', 'products', 'shopdetails', and 'shoplist'. The 'Hero.tsx' file is selected and highlighted in blue. The main editor area displays the code for 'Hero.tsx'. The code includes imports for 'React', 'Image', 'Comment', 'Choose', 'Chooseus', 'About', 'Cards', 'MeetOurChefs', and 'Menu'. It defines a 'HomePage' function that returns a JSX element. The JSX includes a hero section with a background image, a semi-transparent overlay, and a text section with a quote and a heading. The code is formatted with Prettier, as indicated by the 'Prettier' icon in the bottom right corner.

```
src > app > homepage > Hero.tsx > ...
1  import React from "react";
2  import Image from "next/image";
3  import Comment from "../Comment";
4  import Choose from "@components/Choose";
5  import Chooseus from "@components/Chooseus";
6  import About from "../About";
7  import Cards from "../Cards";
8  import MeetOurChefs from "../Chefs";
9  import Menu from "../Menu";
10
Codeium: Refactor | Explain | Generate JSDoc | X
11 export default function HomePage() {
12   return (
13     <>
14       </* Hero Section */>
15       <section
16         className="relative bg-black px-3 md:px-[135px] flex flex-col justify-center md:flex-row md:items-center py-[50px]
17         style={{ backgroundImage: 'url("/imageb1.png")' }}
18       >
19         </* Semi-Transparent Overlay */>
20         <div className="absolute inset-0 bg-black opacity-50 z-0">
21
22         </div>
23         <div className="mb-20">
24           </* Text Section */>
25           <div className="relative z-10 text-white w-full md:w-[50%] text-center md:text-left">
26             <h1 id="quote" className="md:text-[32px] text-[24px] text-[#FF9F0D]">
27               Its Quick & Amusing!
28             </h1>
29             <h1 className="text-[10xl] md:text-[50px] font-bold whitespace-normal mb-4 md:whitespace-normal">
30               <span className="text-[#FF9F0D]">Th</span>e Art of Speed Food Quality
31             </h1>
```



STEPS FOR IMPLEMENTATION:

🔧 Setup:

Integrated Next.js with Sanity CMS to enable fast rendering and dynamic content management. Configured APIs and utilized Sanity's GROQ queries for fetching data efficiently. Tested API requests to ensure smooth and reliable data retrieval.

🔧 Component Building:

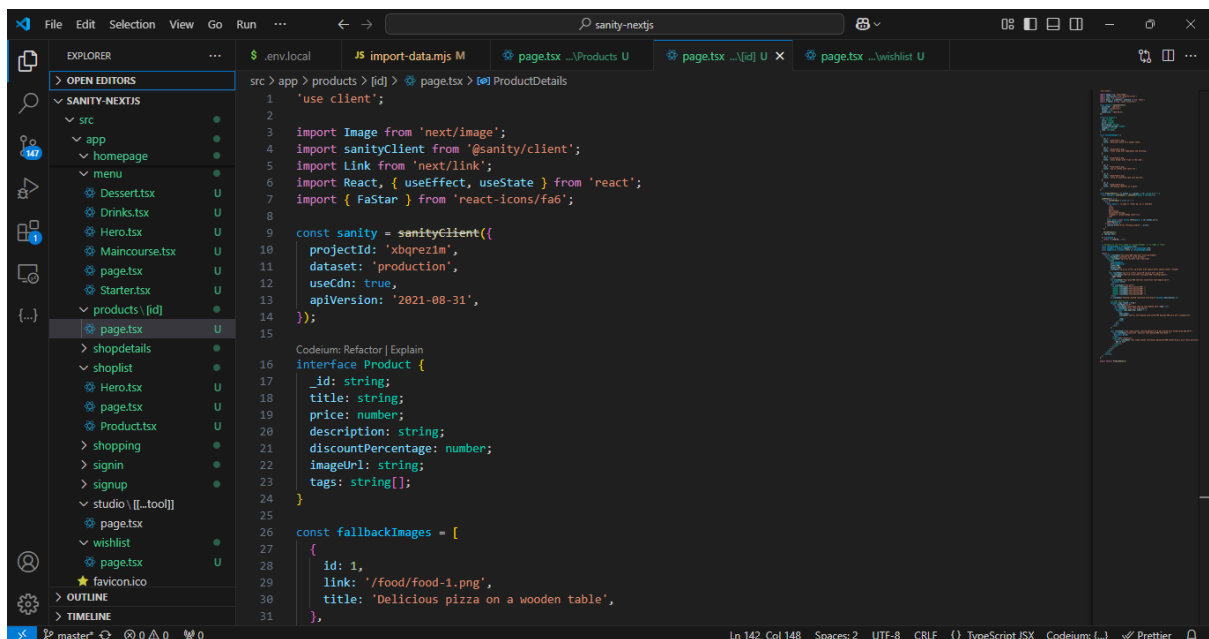
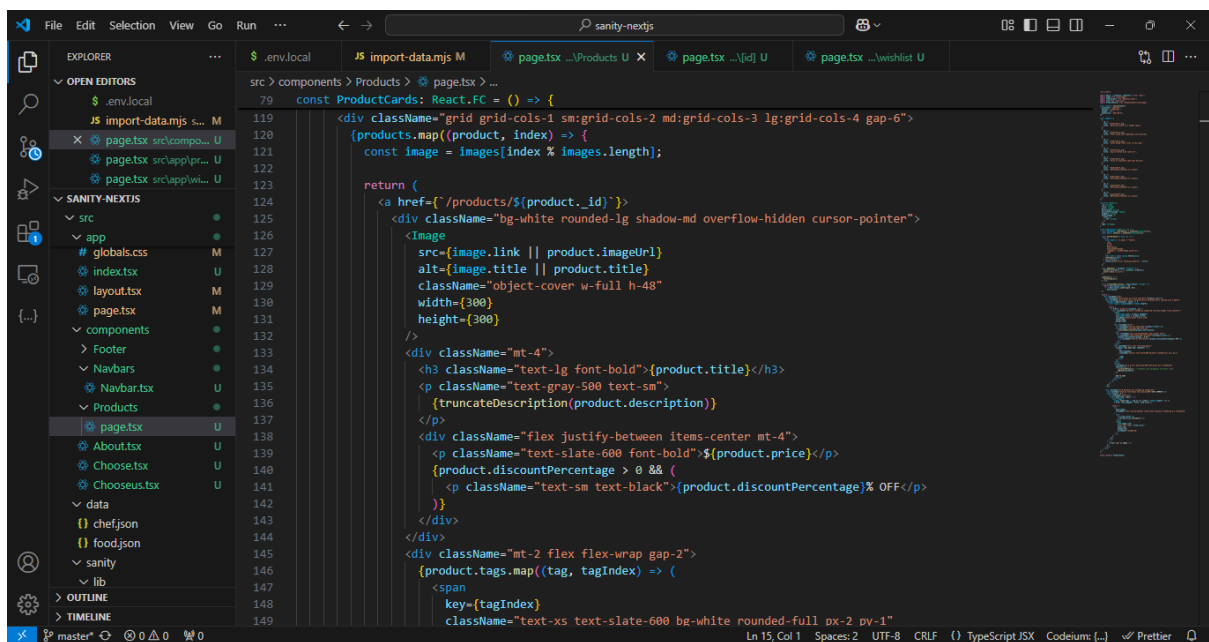
Developed modular and reusable components like Product Details and Cart for a scalable structure. Used Tailwind CSS to design a clean, responsive UI and included optional support for styled-components to allow flexible and custom designs.

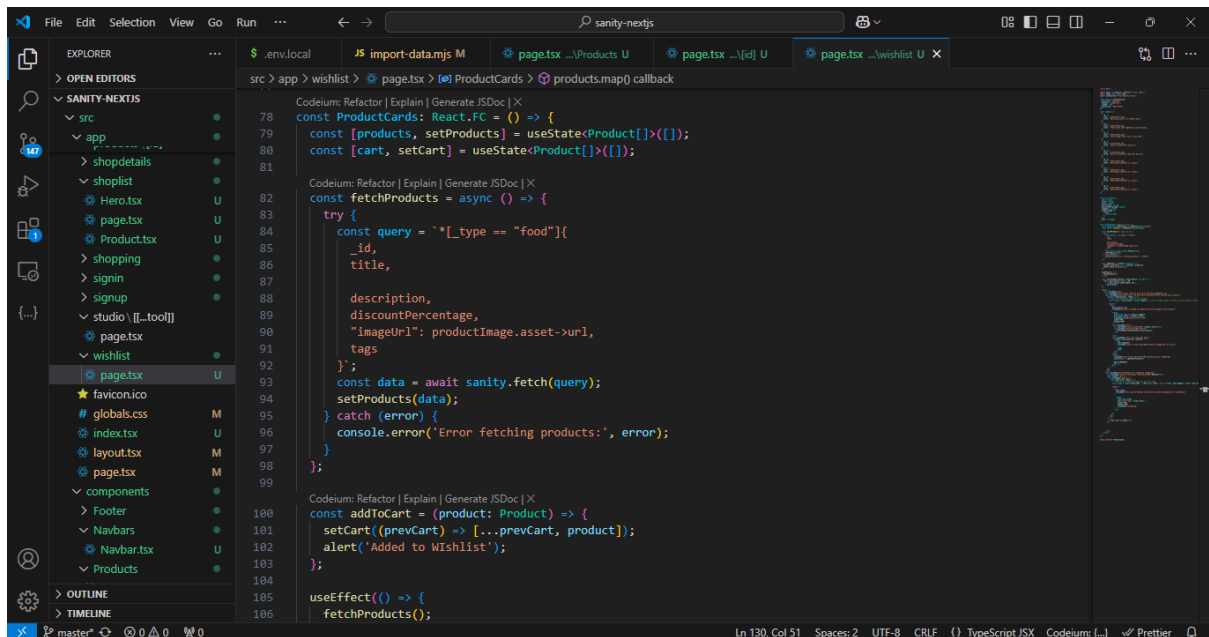
State Management:

Leveraged `useState` and `useEffect` for managing component-level states effectively. Implemented `useContext` to manage global state and streamline data flow across the application. Optimized state handling for seamless user interactions and API calls.

KEY COMPONENTS:

1. Product Card
2. Product Detail
3. Wish list





BACKEND DEVOPLMENT:

Backend development will involve creating a robust server-side infrastructure to handle user requests, process data, and interact with the database. We'll focus on building secure APIs, managing user authentication, and ensuring seamless data flow between the frontend and backend using efficient technologies. This will provide a smooth and reliable experience for users.

PAYMENT GATEWAY:

We offer a convenient cash-on-delivery option, allowing customers to pay in cash when their order is delivered. This provides an added layer of flexibility and trust for those who prefer to pay upon receiving their items.

CONCULOSION:

In conclusion, our website is designed with both user convenience and functionality in mind. The backend development ensures a secure and efficient experience, while the cash-on-delivery option provides flexibility for customers. Together, these elements create a seamless and reliable service that prioritizes ease of use and customer satisfaction.

THE END

