

Day 4 - Dynamic Frontend Components–AS foods

Functional Deliverables:

ScreenShot of:

***Product Listing Component:**



Chicken Chup
Crispy fried chicken bites serv...

\$15 ~~**\$12**~~ 20% OFF

[View Product](#)



Fresh Lime
Refreshing fresh lime drink ma...

\$45 ~~**\$38**~~ 16% OFF

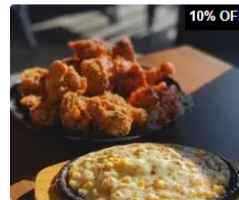
[View Product](#)



Chocolate Muffin
Soft and rich chocolate muffin...

\$30 ~~**\$28**~~ 7% OFF

[View Product](#)



Country Burger
Classic country-style burger...

\$50 ~~**\$46**~~ 10% OFF

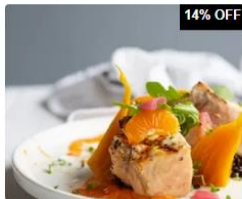
[View Product](#)



Burger
Juicy beef burger with fresh...

\$45 ~~**\$21**~~ 53% OFF

[View Product](#)



Pizza
Delicious vegetarian pizza...

\$50 ~~**\$43**~~ 14% OFF

[View Product](#)

*** Category Component:**



By Ghaniya Khan

Sandwichs



Country Burger
Classic country-style burger served...

\$50

\$45 10% OFF

[View Product](#)



Burger
Juicy beef burger with fresh lettuce,...

\$45

\$24 53% OFF

[View Product](#)

*Product Detail Component:



In Stock

Chicken Chup

Crispy fried chicken bites served with dipping sauce.

\$15 ~~\$42~~ 20% OFF

★★★★★

Rating

Review

- +

[Add to Cart](#)

[Add to Wishlist](#) [Compare](#)

Category: Appetizer

Tags: Sell, Crispy

Share: [Instagram](#) [Twitter](#) [Facebook](#) [Pinterest](#)

Description

Reviews (22)

Lorem Nam tristique porta ligula, vel viverra sem eleifend nec. Nulla sed purus augue, eu euismod tellus. Nam mattis eros nec mi sagittis sagittis. Vestibulum suscipit cursus bibendum. Integer at justo eget sem auctor auctor eget vitae arcu. Nam tempor malesuada porttitor. Nulla quis dignissim ipsum. Aliquam pulvinar iaculis justo, sit amet interdum sem hendrerit vitae. Vivamus vel erat tortor. Nulla facilisi. In nulla quam, lacinia eu aliquam ac, aliquam in nisi.

Suspendisse cursus sodales placerat. Morbi eu lacinia ex. Curabitur blandit justo urna, id porttitor est dignissim nec. Pellentesque scelerisque hendrerit posuere. Sed at dolor quis nisi rutrum accumsan et sagittis massa. Aliquam aliquam accumsan lectus quis auctor. Curabitur rutrum massa at volutpat placerat. Duis sagittis vehicula fermentum. Integer eu vulputate justo. Aenean pretium odio vel tempor sodales. Suspendisse eu fringilla leo, non aliquet sem.

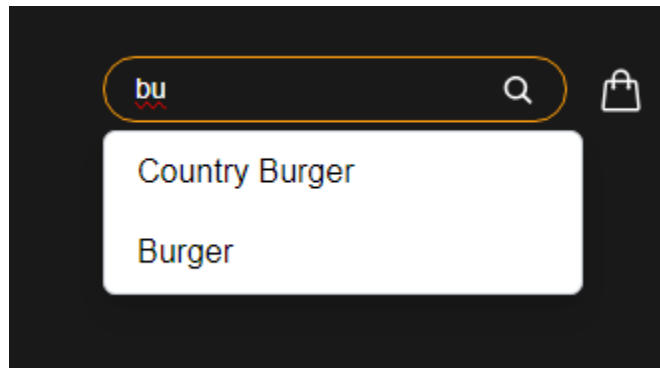
Key Benefits

- . Lorem ipsum dolor sit amet, consectetur adipiscing elit.
- . Maecenas ullamcorper est et massa mattis condimentum.
- . Vestibulum sed massa vel ipsum imperdiet malesuada id tempus nisi.
- . Etiam nec massa et lectus faucibus ornare congue in nunc.
- . Mauris eget diam magna, in blandit turpis.

By Ghaniya Khan

***Search Bar:**

This Screenshot is of the Nav bar .



This is from Shop page where user can find products through Search Bar



Chicken Chup

Crispy fried chicken bites serv...

\$15

\$42 20% OFF

[View Product](#)

***Similar Products:**

. Find this similar products by same tags through API.

By Ghaniya Khan

Similar Products



Code Deliverables:

Search bar Functionality:

- A search bar is implemented using the `SearchableProductList` component.
- The `handleSearch` function filters products based on the search query. It checks the `name` field of each product and shows them.
- The search bar dynamically updates the `filteredData` as the user types.

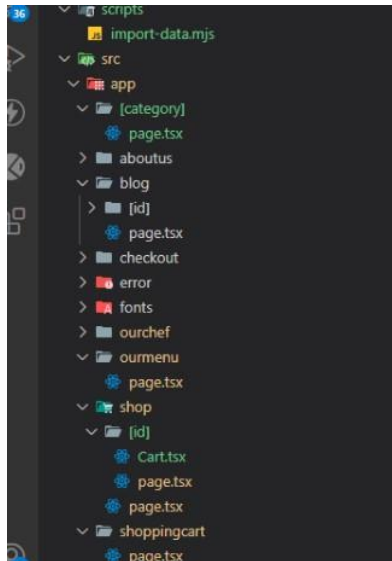
Code:

```
Searchbar.tsx | Nav.tsx | page.tsx | shop.tsx
src > components > layout > Nav.tsx > Nav
12 const Nav = () => {
48   <div className="hidden lg:flex justify-between items-center">
49     <ul className="list-none text-white flex gap-[10px] font-medium leading-[24px] text-[15px]">
50       <li href="/"><li className="w-[45px] h-[24px] font-medium leading-[24px]">Home</li></li>
51       <li href="/ourmenu"><li className="w-[45px] h-[24px] font-medium leading-[24px]">Menu</li></li>
52       <li href="/blog"><li className="w-[45px] h-[24px] font-medium leading-[24px]">Blog</li></li>
53       <li href="/ourchef"><li className="w-[45px] h-[24px] font-medium leading-[24px]">Chef</li></li>
54       <li href="/aboutus"><li className="w-[45px] h-[24px] font-medium leading-[24px]">About</li></li>
55       <li href="/shop"><li className="w-[45px] h-[24px] font-medium leading-[24px]">Shop</li></li>
56       <li href="/signin"><li className="w-[45px] h-[24px] font-medium leading-[24px]">Signin</li></li>
57     </ul>
58   </div>
59   <div className="flex items-center gap-[15px]">
60     <div>
61       <div className="flex items-center gap-[10px] px-[15px] py-[5px] border border-gray-300 rounded-2xl">
62         <input
63           type="text"
64           placeholder="Search..."
65           value={searchQuery}
66           onChange={handleSearch}
67           className="bg-transparent outline-none text-white text-[14px] placeholder:text-white w-full"
68         />
69         <div>
70           <div>
71             <div>
72               <div>
73                 <div>
74                   <div>
75                     <div>
76                       <div>
77                         <div>
78                           <div>
79                             <div>
80                               <div>
81                                 <div>
82                                   <div>
83                                     <div>
84                                       <div>
85                                         <div>
86                                           <div>
87                                             <div>
88                                               <div>
89                                                 <div>
90                                                   <div>
91                                                     <div>
92                                                       <div>
93                                                         <div>
94                                                           <div>
95                                                             <div>
```


Product Details (Dynamic Routes):

- Shows a product image, name, description, price (original and discounted), and a discount percentage for real-time.
- Includes a "View Product" button linking to the product's detailed page using its `_id`.

File Structure:

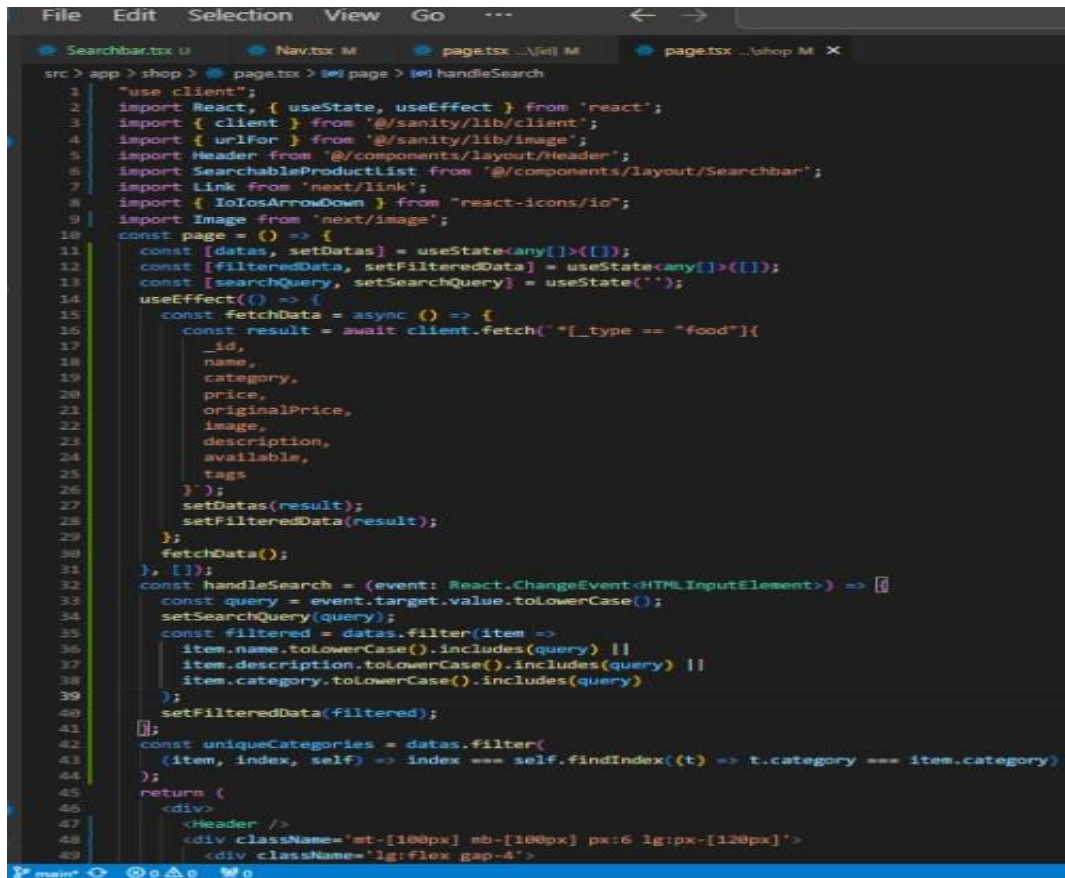


Code:

```
page.tsx M x
src > app > shop > [id] > page.tsx > Page
220
221 const Page = async ({ params }: { params: { id: string } }) => {
222   const datas = await client.fetch(
223     `*[_type == "food" && _id == "${params.id}"][0]{
224     name,
225     category,
226     price,
227     originalPrice,
228     image,
229     description,
230     available,
231     tags
232   }`
233   );
234   const relatedImages = await client.fetch(
235     `*[_type == "food" && "${datas.tags[0]}" in tags][0...3]{
236     "url": image.asset->url
237   }`
238   );
239   return (
240     <div>
241       <Header/>
242       <div className="mt-[100px] mb-[20px] container px-1 mx-auto">...
243     </div>
244     {shopdetail.map((datashop)=>{
245       return(
246         <div className="mt-[0px] mb-[100px] lg:w-4/5 px-12 mx-auto">
247           <div className="flex gap-10">
248             <button className="bg-bordercoloryellow text-whitetext p-2">Description</button><button>Rev
249           </div>
250           <div className="flex flex-col gap-4 text-[14px]">
251             <p>{datashop.description}</p>
252             <p>{datashop.senddespara}</p>
253           </div>
254         </div>
255       )
256     })}
257   )
258 }
```

Product List:

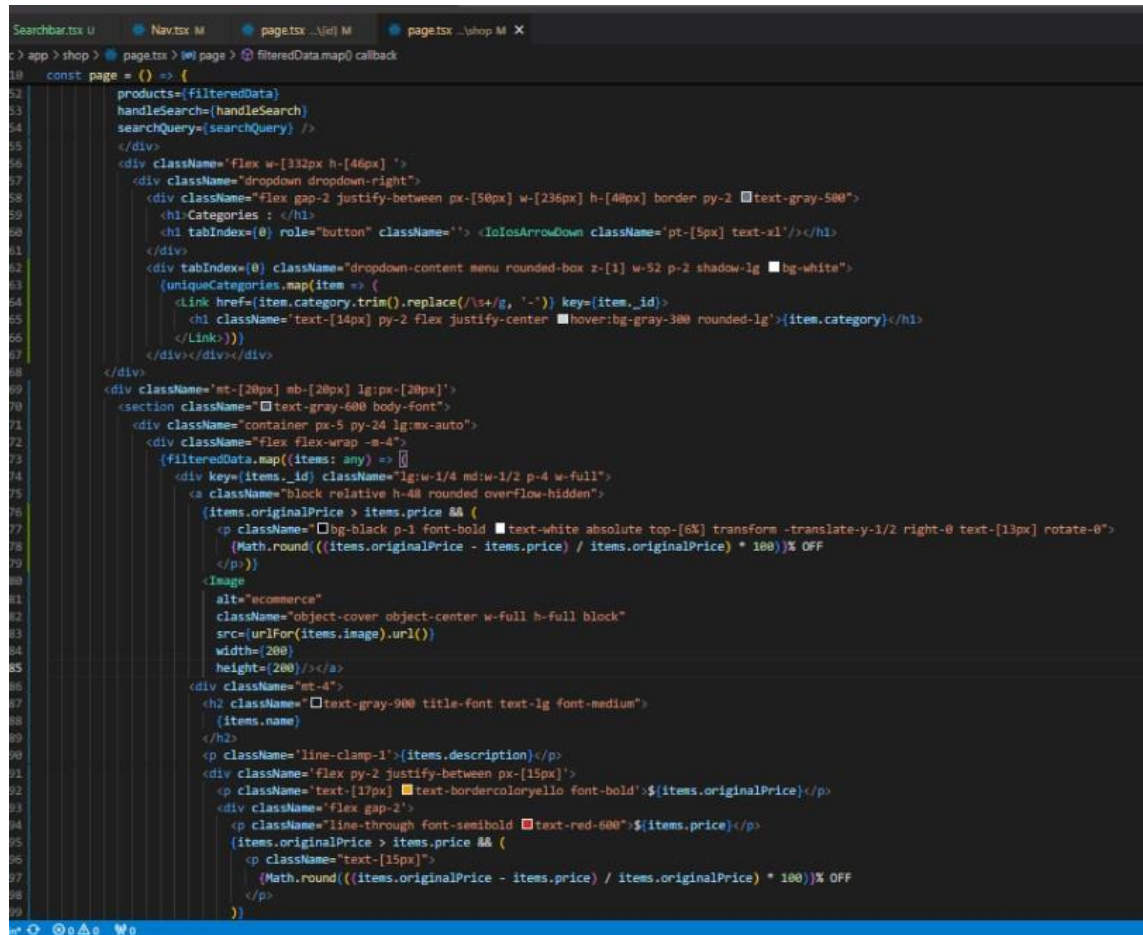
- Fetches data from a Sanity CMS backend for items of type "food".
- Displays each product in a card format, showing its name, description, price, discount, and an image.



```
File Edit Selection View Go ...  
Searchbar.tsx Nav.tsx M pagetss .../shop M X  
src > app > shop > page > handleSearch  
1 "use client";  
2 import React, { useState, useEffect } from 'react';  
3 import { client } from '@sanity/lib/client';  
4 import { urlFor } from '@sanity/lib/image';  
5 import Header from '@components/layout/Header';  
6 import SearchableProductList from '@components/layout/Searchbar';  
7 import Link from 'next/link';  
8 import { IoIosArrowDown } from 'react-icons/io';  
9 import Image from 'next/image';  
10 const page = () => {  
11   const [datas, setDatas] = useState<any[]>([]);  
12   const [filteredData, setFilteredData] = useState<any[]>([]);  
13   const [searchQuery, setSearchQuery] = useState('');  
14   useEffect(() => {  
15     const fetchData = async () => {  
16       const result = await client.fetch('*[_type == "food"]{  
17         _id,  
18         name,  
19         category,  
20         price,  
21         originalPrice,  
22         image,  
23         description,  
24         available,  
25         tags  
26       }');  
27       setDatas(result);  
28       setFilteredData(result);  
29     };  
30     fetchData();  
31   }, []);  
32   const handleSearch = (event: React.ChangeEvent<HTMLInputElement>) => {  
33     const query = event.target.value.toLowerCase();  
34     setSearchQuery(query);  
35     const filtered = datas.filter(item => {  
36       item.name.toLowerCase().includes(query) ||  
37       item.description.toLowerCase().includes(query) ||  
38       item.category.toLowerCase().includes(query)  
39     });  
40     setFilteredData(filtered);  
41   };  
42   const uniqueCategories = datas.filter(  
43     (item, index, self) => index === self.findIndex((t) => t.category === item.category)  
44   );  
45   return (  
46     <div>  
47       <Header />  
48       <div className='mt-[100px] mb-[100px] px:6 lg:px-[120px]`>  
49         <div className='lg:flex gap-4`>
```

Categories Dropdown:

- Filters unique categories from the fetched products.
- Created a dropdown menu with clickable category links.



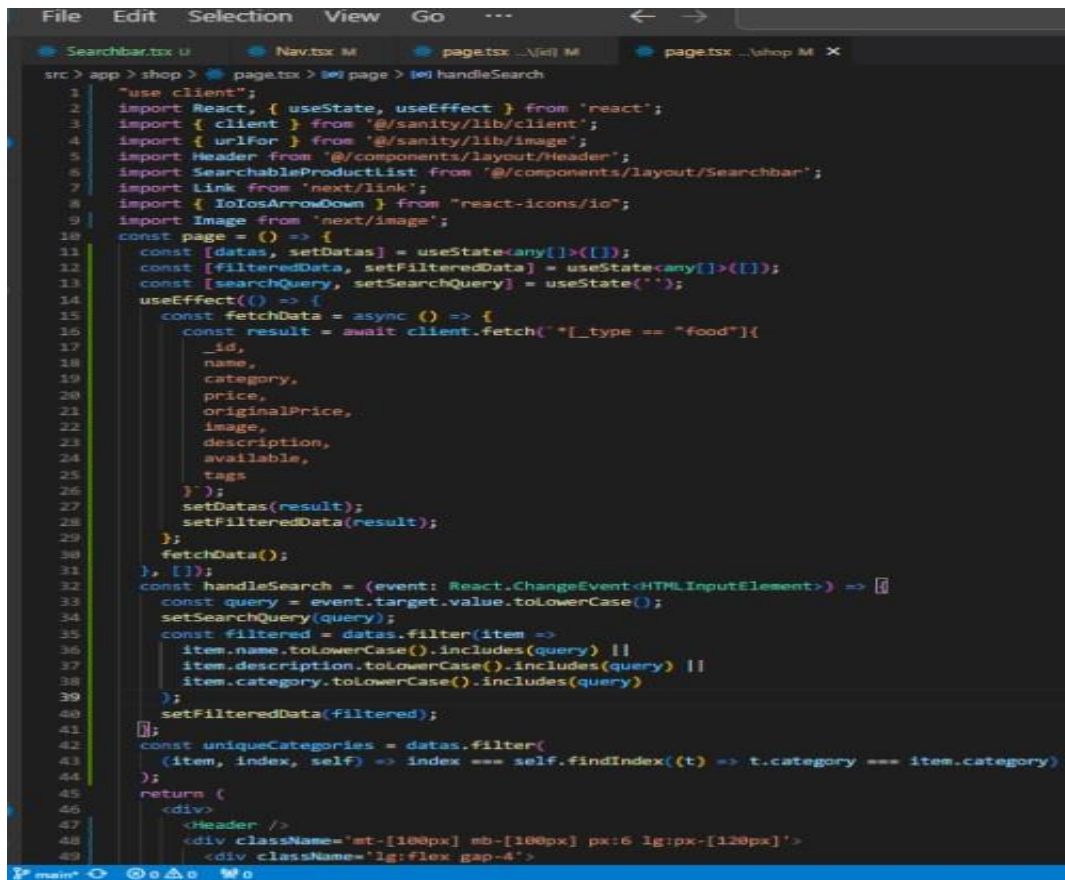
```
Searchbar.tsx U Nav.tsx M pagetsx ...[el] M pagetsx ...shop M X
c> app > shop > pagetsx > 100 page > filteredData.map() callback
10 const page = () => {
11   products=filteredData
12   handleSearch=handleSearch
13   searchQuery=searchQuery />
14 </div>
15 <div className='flex w-[332px] h-[46px]'>
16   <div className='dropdown dropdown-right'>
17     <div className='flex gap-2 justify-between px-[50px] w-[236px] h-[40px] border py-2 text-gray-500'>
18       <h1>Categories : </h1>
19       <h1 tabIndex={0} role='button' className=''><IoIosArrowDown className='pt-[5px] text-xl' /></h1>
20     </div>
21     <div tabIndex={0} className='dropdown-content menu rounded-box z-[1] w-52 p-2 shadow-lg bg-white'>
22       {uniqueCategories.map(item => (
23         <Link href={item.category.trim().replace(/\\/s+/g, '-')} key={item._id}>
24         <h1 className='text-[14px] py-2 flex justify-center hover:bg-gray-300 rounded-lg'>{item.category}</h1>
25       </Link>))}
26     </div></div></div>
27 </div>
28 <div className='mt-[20px] mb-[20px] lg:px-[20px]'>
29   <section className='text-gray-600 body-font'>
30     <div className='container px-5 py-24 lg:mx-auto'>
31       <div className='flex flex-wrap -m-4'>
32         {filteredData.map((items: any) => (
33           <div key={items._id} className='lg:w-1/4 md:w-1/2 p-4 w-full'>
34             <a className='block relative h-48 rounded overflow-hidden'>
35               {items.originalPrice > items.price && (
36                 <p className='bg-black p-1 font-bold text-white absolute top-[6px] transform -translate-y-1/2 right-0 text-[13px] rotate-0'>
37                   (Math.round(((items.originalPrice - items.price) / items.originalPrice) * 100))% OFF
38                 </p>
39               <Image
40                 alt='ecommerce'
41                 className='object-cover object-center w-full h-full block'
42                 src={urlFor(items.image).url()}
43                 width={200}
44                 height={200} /></a>
45             <div className='mt-4'>
46               <h2 className='text-gray-900 title-font text-lg font-medium'>
47                 {items.name}
48               </h2>
49               <p className='line-clamp-1'>{items.description}</p>
50               <div className='flex py-2 justify-between px-[15px]'>
51                 <p className='text-[17px] text-bordercoloryellow font-bold'>${items.originalPrice}</p>
52                 <div className='flex gap-2'>
53                   <p className='line-through font-semibold text-red-600'>${items.price}</p>
54                   {items.originalPrice > items.price && (
55                     <p className='text-[15px]'>
56                       (Math.round(((items.originalPrice - items.price) / items.originalPrice) * 100))% OFF
57                     </p>
58                   )}
59                 </div>
60               </div>
61             </div>
62           </div>
63         )}
64       </div>
65     </div>
66   </section>
67 </div>
68 )}
```

API Integration:

1. Sanity API Fetching Logic: Since you're using Sanity CMS, you need to ensure data fetching is properly managed.

- **Sanity Client Setup** (if not already set up): In your `lib` folder, configure the Sanity client:
- **Fetching Data in `useEffect`:**

By Ghaniya Khan



```
File Edit Selection View Go ...
Searchbar.tsx U Nav.tsx M pagetsx .../id M pagetsx .../shop M X
src > app > shop > pagetsx > page > handleSearch
1 "use client";
2 import React, { useState, useEffect } from "react";
3 import { client } from "@sanity/lib/client";
4 import { urlFor } from "@sanity/lib/image";
5 import Header from "@components/layout/Header";
6 import SearchableProductList from "@components/layout/Searchbar";
7 import Link from "next/link";
8 import { IoIosArrowDown } from "react-icons/io";
9 import Image from "next/image";
10 const page = () => {
11   const [datas, setDatas] = useState<any[]>([]);
12   const [filteredData, setFilteredData] = useState<any[]>([]);
13   const [searchQuery, setSearchQuery] = useState("");
14   useEffect(() => {
15     const fetchData = async () => {
16       const result = await client.fetch(`*[!_type == "food"]{
17         _id,
18         name,
19         category,
20         price,
21         originalPrice,
22         image,
23         description,
24         available,
25         tags
26       }`);
27       setDatas(result);
28       setFilteredData(result);
29     };
30     fetchData();
31   }, []);
32   const handleSearch = (event: React.ChangeEvent<HTMLInputElement>) => {
33     const query = event.target.value.toLowerCase();
34     setSearchQuery(query);
35     const filtered = datas.filter(item =>
36       item.name.toLowerCase().includes(query) ||
37       item.description.toLowerCase().includes(query) ||
38       item.category.toLowerCase().includes(query)
39     );
40     setFilteredData(filtered);
41   };
42   const uniqueCategories = datas.filter(
43     (item, index, self) => index === self.findIndex((t) => t.category === item.category)
44   );
45   return (
46     <div>
47       <Header />
48       <div className="mt-[100px] mb-[100px] px:6 lg:px-[120px]">
49         <div className="lg:flex gap-4">
```

Dynamic Routing

Dynamic routing is used to handle routes like `/shop/:id` for individual product pages.

1. Create Dynamic Route File:

In the pages directory, create a `[id].tsx` file inside the shop folder:

`pages/shop/[id].tsx`

2. Get Product Data Dynamically:


```

page.tsx M X
src > app > shop > [id] > page.tsx > [e] Page
220
221 const Page = async ({ params }: { params: { id: string } }) => {
222   const datas = await client.fetch(
223     `*_type == "food" && _id == "${params.id}"[0]{
224     name,
225     category,
226     price,
227     originalPrice,
228     image,
229     description,
230     available,
231     tags
232   }`
233   );
234   const relatedImages = await client.fetch(
235     `*_type == "food" && "${datas.tags[0]}" in tags[0...3]{
236     "url": image.asset->url
237   }`
238   );
239   return (
240     <div>
241       <Header/>
242       <div className="mt-[100px] mb-[20px] container px-1 mx-auto">...
243     </div>
244     {shopdetail.map((datashop)=>{
245       return(
246         <div className="mt-[0px] mb-[100px] lg:w-4/5 px-12 mx-auto">
247           <div className="flex gap-10">
248             <button className="bg-bordercoloryello text-whitetext p-2">Description</button><button>Rev
249           </div>
250           <div className="flex flex-col gap-4 text-[14px]">
251             <p>{datashop.description}</p>
252             <p>{datashop.senddespara}</p>
253           </div>
254         </div>
255       )
256     })}
257   )
258 }

```

Final CheckList

Frontend Component Development	Styling and Responsiveness	Code Quality	Documentation and Submission	Final Review
✓	✓	✓	✓	✓

By Ghaniya Khan