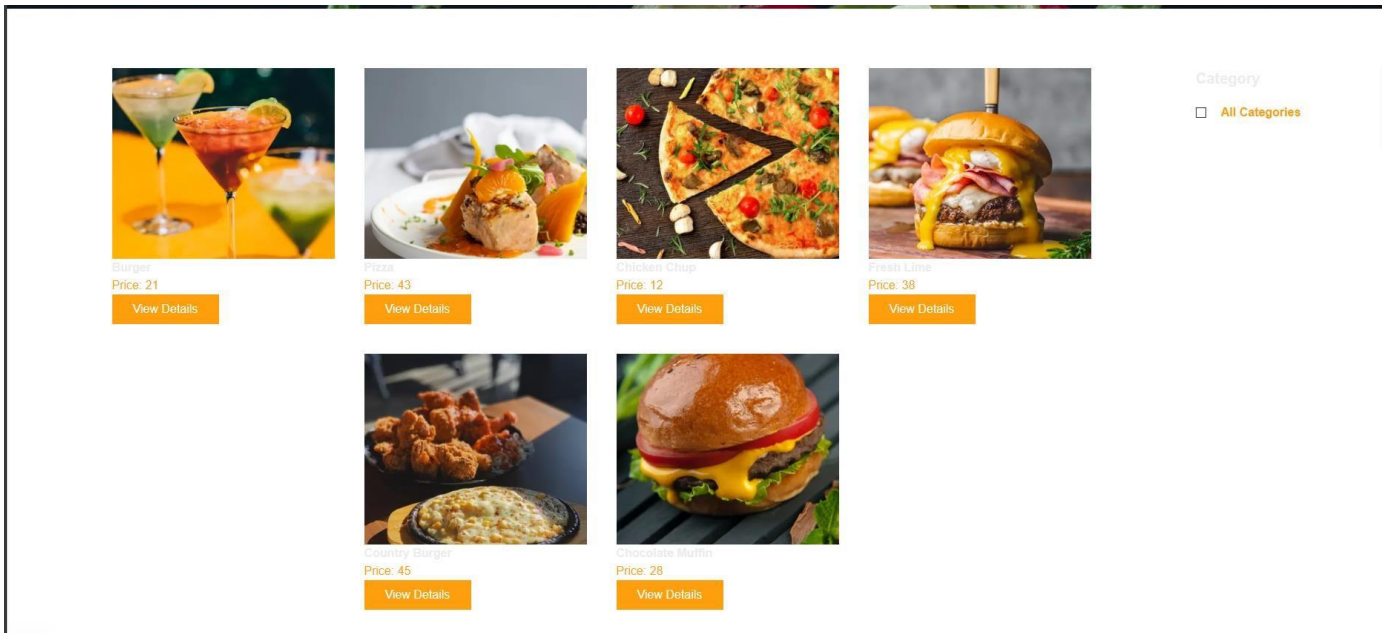


Day 4 - Dynamic Frontend Components–Temp-09

Functional Deliverables

Screenshots:

1. Product Listing Component:



2. Category Component:



By: Arman Zaman Khan (00050489)

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 nisl.

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 nisl.
- Etiam nec massa et lectus faucibus ornare congue in nunc.
- Mauris eget diam magna, in blandit turpis.

Search Bar:

By: Arman Zaman Khan (00050489)

This Screenshot is of the Search .



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



Chicken Chup

Crispy fried chicken bites serv...

\$15

\$12 20% OFF

[View Product](#)

- 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 show them.
- The search bar dynamically updates the filteredData as the user types.

By: Arman Zaman Khan (00050489)

Code:

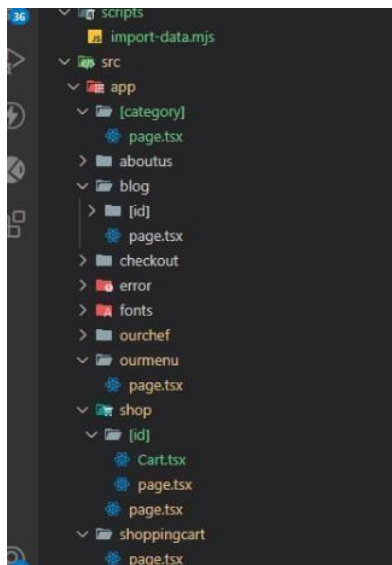
```
src > components > layout > Nav.jsx > @ Nav
12 const Nav = () => {
48   <div className="hidden lg:flex justify-between items-center">
49     <ul>
50       <li className="w-[45px] h-[24px] font-medium leading-[24px] text-[15px]">
51         <Link href="/"><li className="w-[45px] h-[24px] font-medium leading-[24px]">'.Home</li></Link>
52         <Link href="/ourmenu"><li className="w-[45px] h-[24px] font-medium leading-[24px]">'.Menu</li></Link>
53         <Link href="/blog"><li className="w-[45px] h-[24px] font-medium leading-[24px]">'.Blog</li></Link>
54         <Link href="/ourchef"><li className="w-[45px] h-[24px] font-medium leading-[24px]">'.Chef</li></Link>
55         <Link href="/aboutus"><li className="w-[45px] h-[24px] font-medium leading-[24px]">'.About</li></Link>
56         <Link href="/shop"><li className="w-[45px] h-[24px] font-medium leading-[24px]">'.Shop</li></Link>
57         <Link href="/signin"><li className="w-[45px] h-[24px] font-medium leading-[24px]">'.Signin</li></Link>
58       </ul>
59     <div className="flex items-center gap-[15px]">
60       <div>
61         <div className="flex items-center gap-[18px] px-[15px] py-[5px] border border-bordercoloryellow 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           <divSearch className="text-white w-[20px] h-[20px]" />
70         </div>
71         {searchQuery && filteredProducts.length > 0 && (
72           <div className="absolute bg-white w-[240px] mt-1 border border-gray-300 rounded-md shadow-lg z-10">
73             <ul>
74               {filteredProducts.map((product: any) => (
75                 <li key={product._id} className="px-4 py-2 text-black hover:bg-gray-200 cursor-pointer">
76                   <Link href="/shop/${product._id}">
77                     {product.name}
78                   </Link>
79                 </li>
80               ))}
81             </ul>
82           </div>
83         )}
84       </div>
85       <Link href="/shoppingcart">
86         <div className="relative">
87           <divOutlineShoppingBag className="text-white text-[24px] cursor-pointer" />
88           {cart.length > 0 && (
89             <span className="absolute -top-2 -right-2 bg-red-500 text-white text-xs font-bold rounded-full w-5 h-5 flex items-center justify-center">
90               {cart.length}
91             </span>
92           )}
93         </div>
94       </Link>
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.

By: Arman Zaman Khan (00050489)

File Structure:



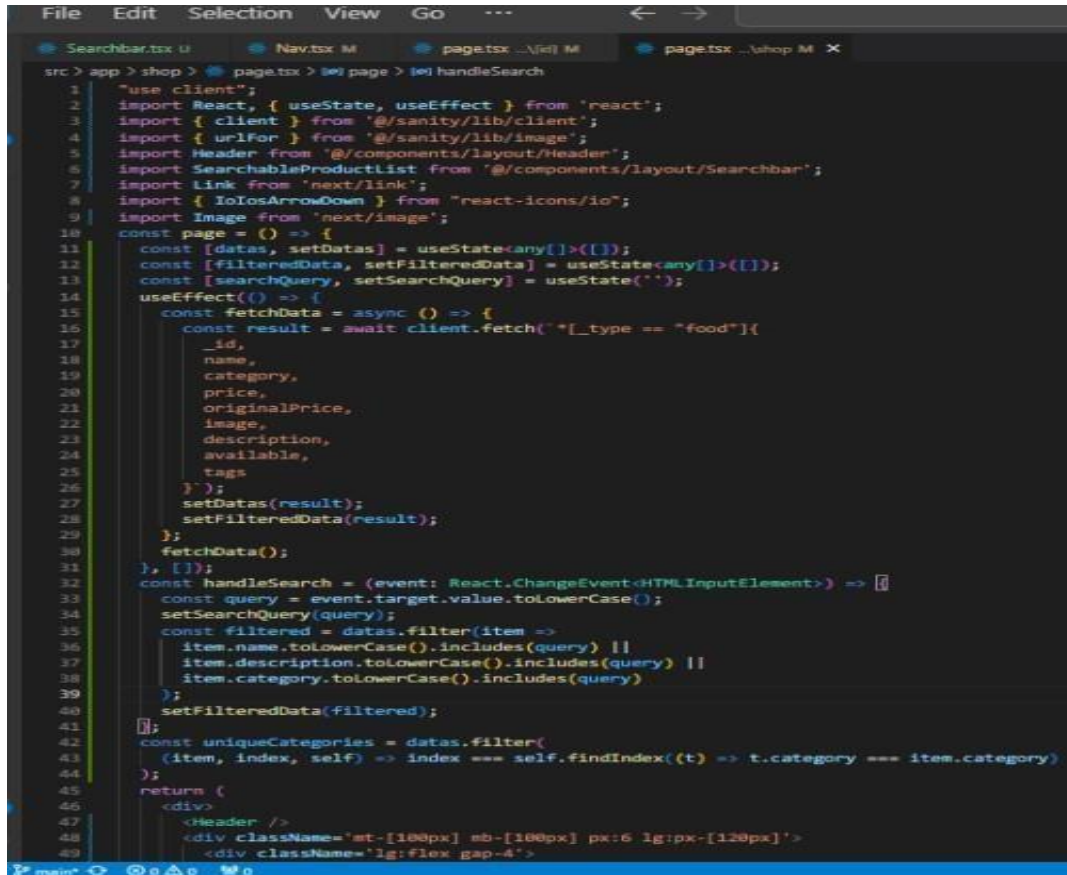
Code:

```
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 }
```

By: Arman Zaman Khan (00050489)

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 U Nav.tsx M page.tsx ...\file M page.tsx ...\shop M X  
src > app > shop > page.tsx > [edit] page > [edit] 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  
11 const page = () => {  
12   const [datas, setDatas] = useState<any[]>([]);  
13   const [filteredData, setFilteredData] = useState<any[]>([]);  
14   const [searchQuery, setSearchQuery] = useState('');  
15   useEffect(() => {  
16     const fetchData = async () => {  
17       const result = await client.fetch(`*[_type == "food"]{  
18         _id,  
19         name,  
20         category,  
21         price,  
22         originalPrice,  
23         image,  
24         description,  
25         available,  
26         tags  
27       }`);  
28       setDatas(result);  
29       setFilteredData(result);  
30     };  
31     fetchData();  
32   }, []);  
33   const handleSearch = (event: React.ChangeEvent<HTMLInputElement>) => {  
34     const query = event.target.value.toLowerCase();  
35     setSearchQuery(query);  
36     const filtered = datas.filter(item => {  
37       item.name.toLowerCase().includes(query) ||  
38       item.description.toLowerCase().includes(query) ||  
39       item.category.toLowerCase().includes(query)  
40     });  
41     setFilteredData(filtered);  
42   };  
43   const uniqueCategories = datas.filter(  
44     (item, index, self) => index === self.findIndex((t) => t.category === item.category)  
45   );  
46   return (  
47     <div>  
48       <Header />  
49       <div className="mt-[100px] mb-[100px] px:6 lg:px-[120px]">  
50         <div className="lg:flex gap-4">
```

Categories Dropdown:

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

By: Arman Zaman Khan (00050489)

By: Arman Zaman Khan (00050489)

```
File Edit Selection View Go ...
Searchbar.tsx U Nav.tsx M pagetx ...[id] M pagetx ...shop M X
src > app > shop > pagetx > src page > src 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:

By: Arman Zaman Khan (00050489)


```

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-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 }

```

Final Checklist

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

By: Arman Zaman Khan (00050489)