

# **Day 4**

## **Dynamic Frontend Components**

**Prepared by: Syed Nauman Navaid**

**Slot: Thursday morning 9 to 12**

## Dynamic Frontend Components

On Day 4, I focused on building five dynamic frontend components for my e-commerce marketplace. These components aimed to improve the user experience by enabling product filtering, searching, dynamic routing, product listing, and pagination. While all components were functional, I ensured they integrated seamlessly with my Sanity CMS backend. Below, I will explain each component in detail.

---

### 1. Category Filter Component

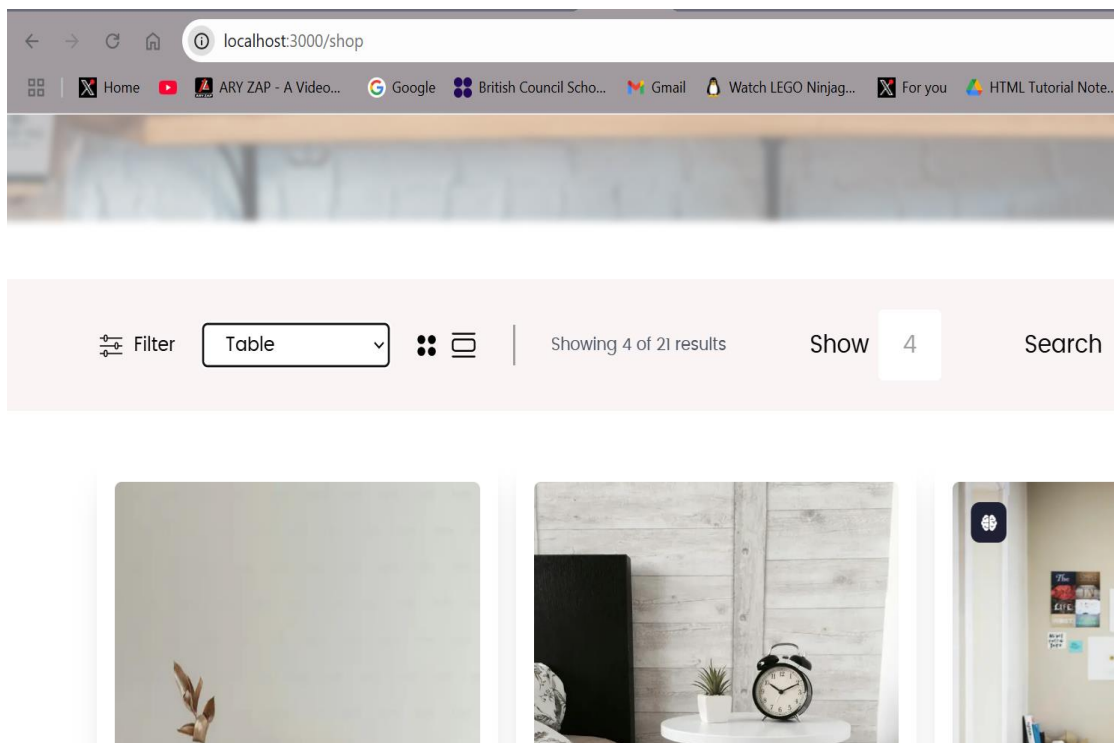
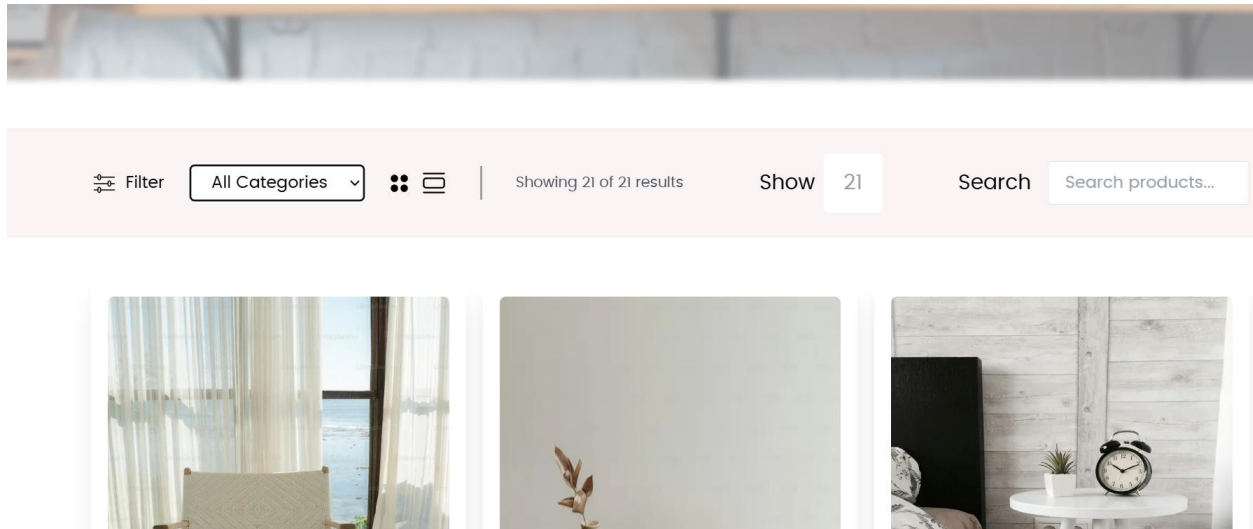
The **Category Filter Component** is designed to fetch and display products based on their category. Since my product schema in Sanity CMS includes a `category` field, I utilized it to create a filter bar on the `ShopPage`. Here's how I implemented it:

- **Sanity Query:** I used GROQ (Sanity's query language) to fetch all unique categories from the product schema.
- **Filter Logic:** Each category was displayed as a button in the filter bar. Clicking a button triggered a query to fetch products belonging to that category.
- **Dynamic Updates:** The product list dynamically updated whenever a category was selected.

For example:

- Categories like "Sofas," "Recliners," and "Loveseats" each fetched their respective products.
- This approach ensured users could quickly filter products by type.

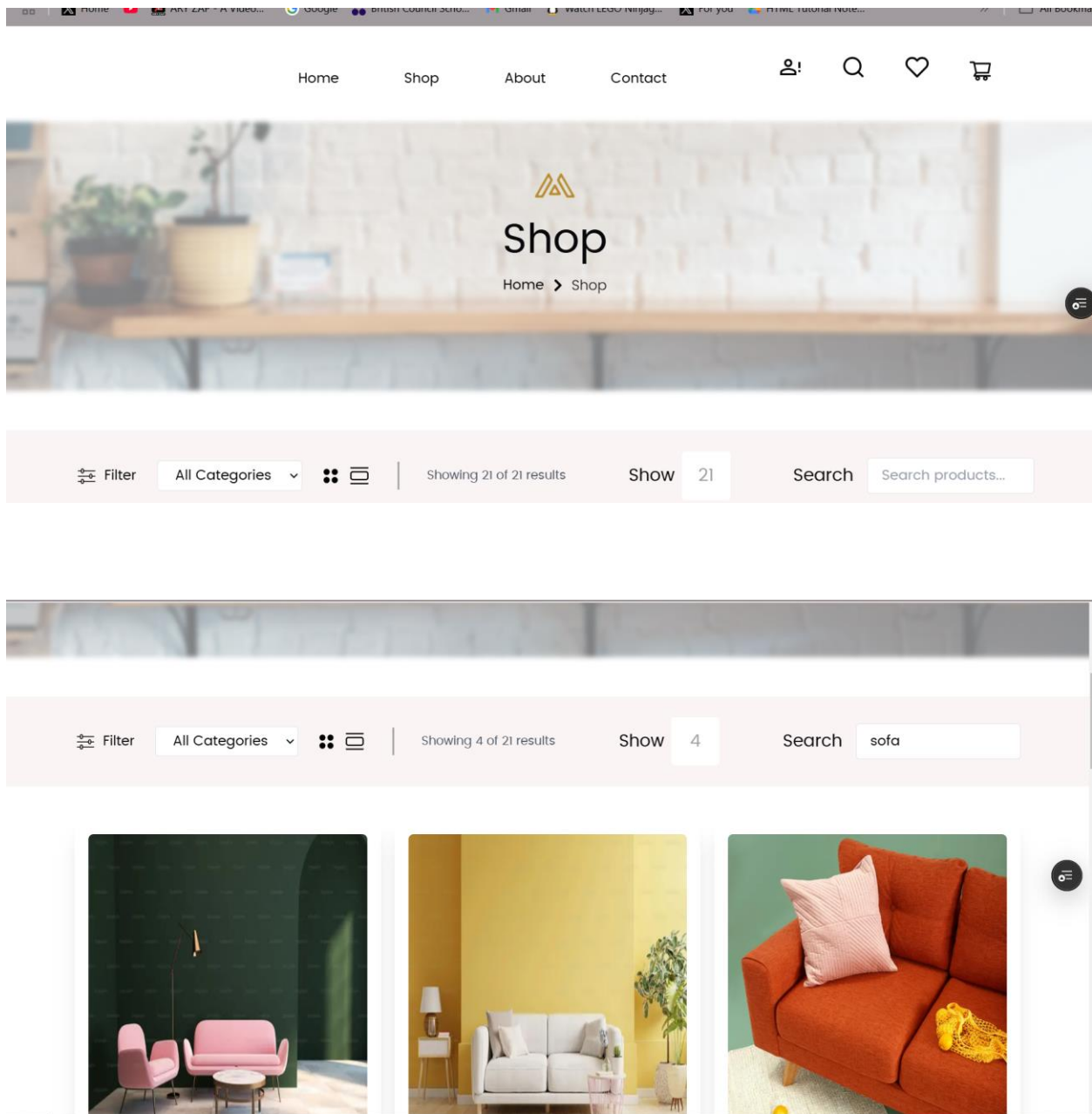
## Category Filter Component snippet:



## 2. Search Bar Component

The search bar allows users to search a specific product by its name and updates in real time as they are typing so they will be able to see what they want quick.

### Search bar snippet:



### 3. Pagination Component

The **Pagination Component** is designed to divide the product list into pages, making navigation easier for users, especially when dealing with a large number of products. Here's how I implemented it:

#### Implementation Details

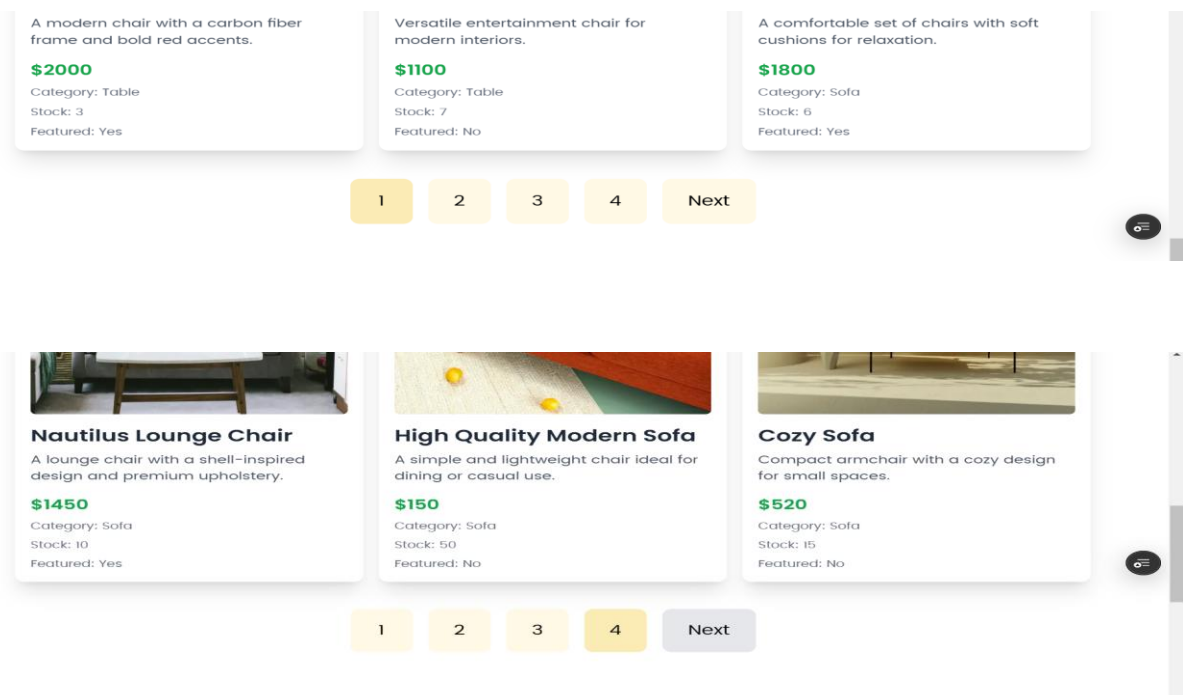
##### 1. Page Size

- The limit for each page is set to **6 products per page**.

##### 2. Product per Page Count Calculation:

- The total number of products fetched from Sanity CMS is divided by 6 to determine the total number of pages.
- Example:
  - **Total Products:** 21
  - **No of page:**  $21/6 = 4$
  - **Result:** the result of this is that on the first 3 page the number of products displayed are 6 and on the 4 remaining 3.
- Users can navigate between pages using:
  - **Next** and **1,2,3,4** buttons.
  - Direct page numbers displayed as clickable buttons.

#### Pagination component snippet:



## 4. Product listing component

In this component we have managed the product display on the shop page where we have fetched all the products from sanity using the query.

### Implementation:

#### Fetching products:

Below is the code for fetching products

```
const fetchProducts = async () => {  
  const products = await sanityClient.fetch(`  
    *[_type == "product"] | order(_createdAt desc)  
  `);  
  return products;  
};
```

#### Displaying products:

Map through the fetched products and display them in a grid or list format.

```
import { useState, useEffect } from "react";  
import sanityClient from "@sanityClient";  
  
const ProductList = () => {  
  const [products, setProducts] = useState([]);  
  
  const fetchProducts = async () => {  
    const productsData = await sanityClient.fetch(`  
      *[_type == "product"] | order(_createdAt desc)  
    `);  
    setProducts(productsData);  
  };  
  
  useEffect(() => {  
    fetchProducts();  
  }, []);  
  
  return (  

```


```

<div className="product-list grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3
gap-6">
  {products.map((product) => (
    <div key={product._id} className="product-item p-4 border rounded-lg">
      <img src={product.image} alt={product.name} className="mb-4 w-full h-48
object-cover" />
      <h3 className="text-lg font-semibold">{product.name}</h3>
      <p className="text-gray-500">{product.description}</p>
      <p className="text-blue-600 font-bold mt-2">${product.price}</p>
    </div>
  ))}
</div>
);
};

export default ProductList;

```

## Product Displayed on frontend:



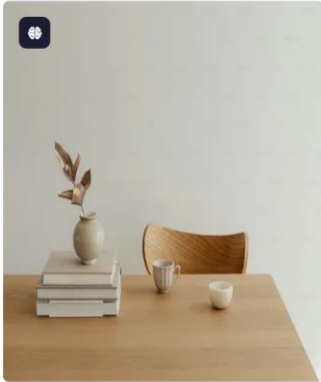
**Chair Wibe**

A sleek outdoor chair with natural wooden elements and a modern look.

**\$1200**

Category: Chair

Stock: 25




**Alpha Table**

A sturdy oak chair with a sleek and minimalist design.

**\$900**

Category: Table

Stock: 18



**Replica Table**

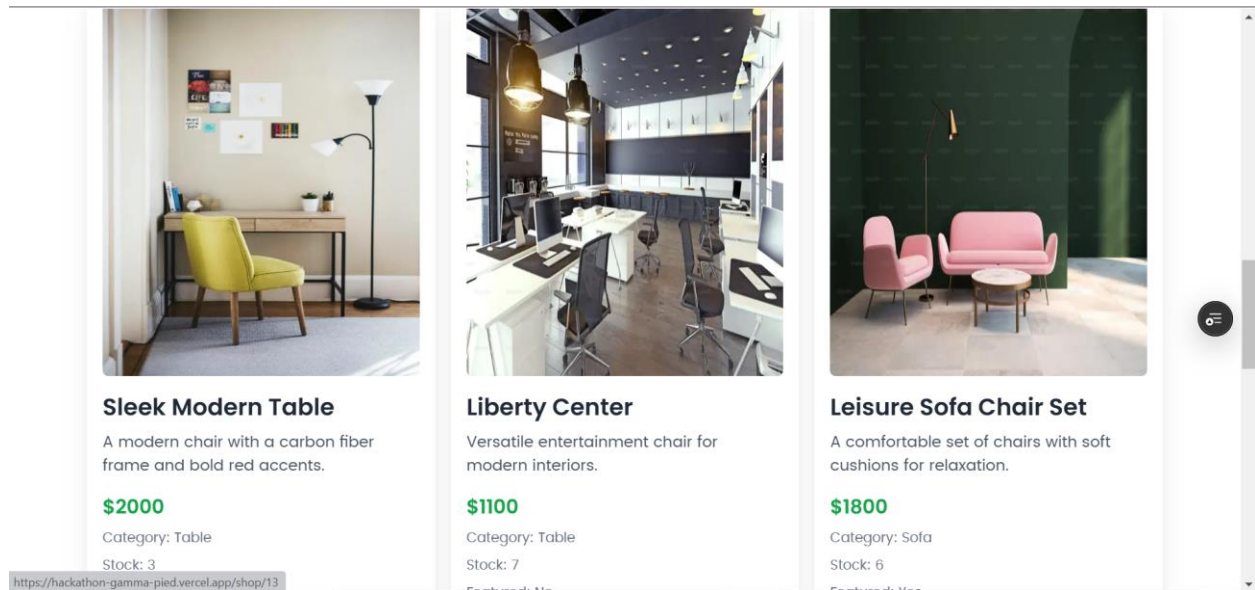
Classic wishbone chair with a dark walnut frame and cord seat.

**\$750**

Category: Table

Stock: 25

<https://hackathon-ed.vercel.app/shop/10>



## 5. Dynamic routing:

In this component I made every product be displayed individually when clicked on it. Here's how I implemented it

### Implementation details:

- I created a dynamic route under shop route [id] in which I fetched products by their ids below is the code for it:

```
import { client } from '@sanity/lib/client';
import Image from 'next/image';
import { Button } from '@components/ui/button';

const getProduct = async (id: string) => {
  const query = `*[_type == "product" && _id == "${id}"]{
    _id,
    name,
    imagePath,
    price,
    description,
    discountPercentage,
    stockLevel,
    category
  }[0]`;
}
```



```

    return client.fetch(query);
  });

const Page = async (props: any) => {
  const { params } = props;
  const product = await getProduct(params.id);
  console.log(product);
  if (!product) {
    return <div>Product not found.</div>;
  }

  return (
    <div className="grid md:grid-cols-2 gap-10 md:mx-10 sm:mx-4 mt-10">
      <div className="items-center justify-center flex">
        <Image
          src={product.imagePath}
          alt={product.name}
          width={300}
          height={300}
          className="md:w-[500px] md:h-[500px] rounded-md"
        />
      </div>
      <div className="flex flex-col gap-3">
        <p className="font-bold text-3xl">{product.name}</p>
        <div className="flex gap-3 m-4 text-xl font-semibold">
          <p className="text-green-600">${product.price}</p>
          {product.discountPercentage > 0 && (
            <span>{product.discountPercentage}% off</span>
          )}
        </div>
        <p className="font-semibold text-lg">In Stock: {product.stockLevel}</p>
        <p className="font-light text-md text-opacity-50 text-black">
          Category: {product.category}
        </p>
        <p className="font-medium text-2xl">{product.description}</p>
        <Button className="w-64 bg-black text-white mt-5 transition duration-300
ease-in-out transform hover:scale-105">
          Add To Cart
        </Button>
      </div>
    </div>
  );
};

export default Page

```

## Dynamic data frontend:



### Chair Wibe

**\$1200** 10% off

**In Stock: 25**

Category: Chair

A sleek outdoor chair with natural wooden elements and a modern look.

Add To Cart