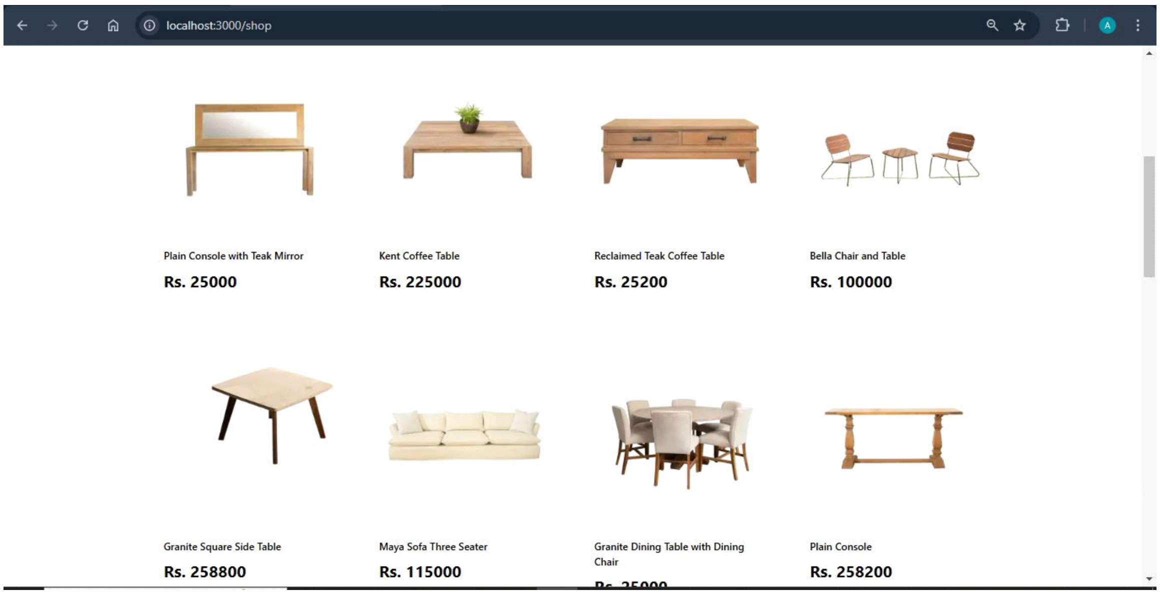
# Day 4 - Dynamic Frontend Components - FurniSphere

1. **Functional Deliverables:**

Product Listing page with dynamic data.


localhost:3000/shop

Plain Console with Teak Mirror
Rs. 25000

Kent Coffee Table
Rs. 225000

Reclaimed Teak Coffee Table
Rs. 25200

Bella Chair and Table
Rs. 100000

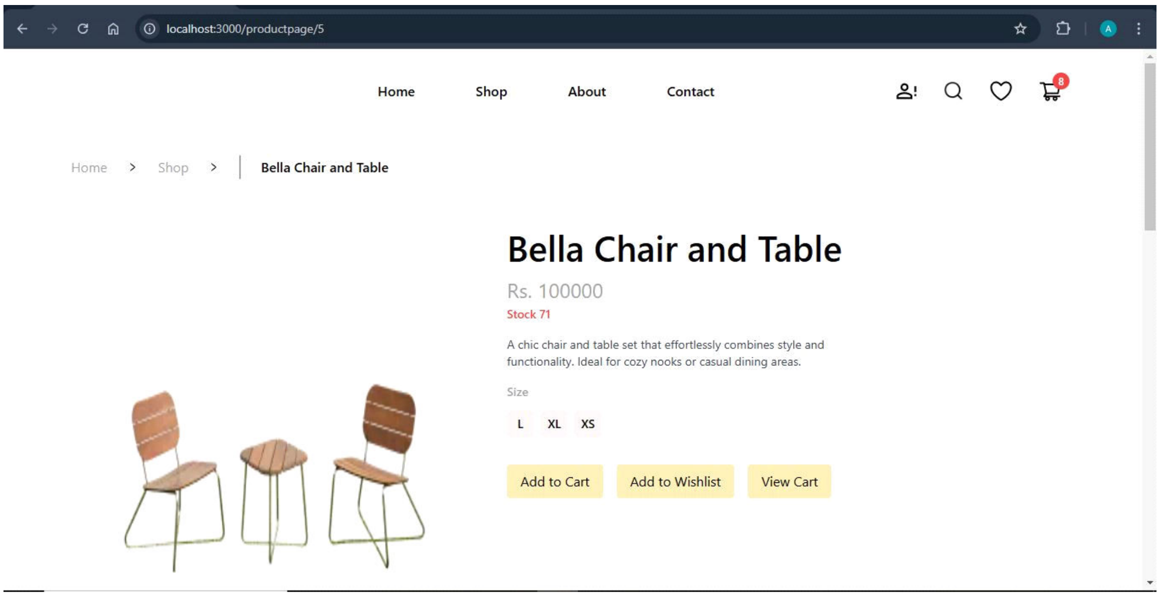
Granite Square Side Table
Rs. 258800

Maya Sofa Three Seater
Rs. 115000

Granite Dining Table with Dining Chair
Rs. 25000

Plain Console
Rs. 258200

Individual product detail pages with accurate routing and data rendering.


localhost:3000/productpage/5

Home > Shop > Bella Chair and Table

Bella Chair and Table

Rs. 100000

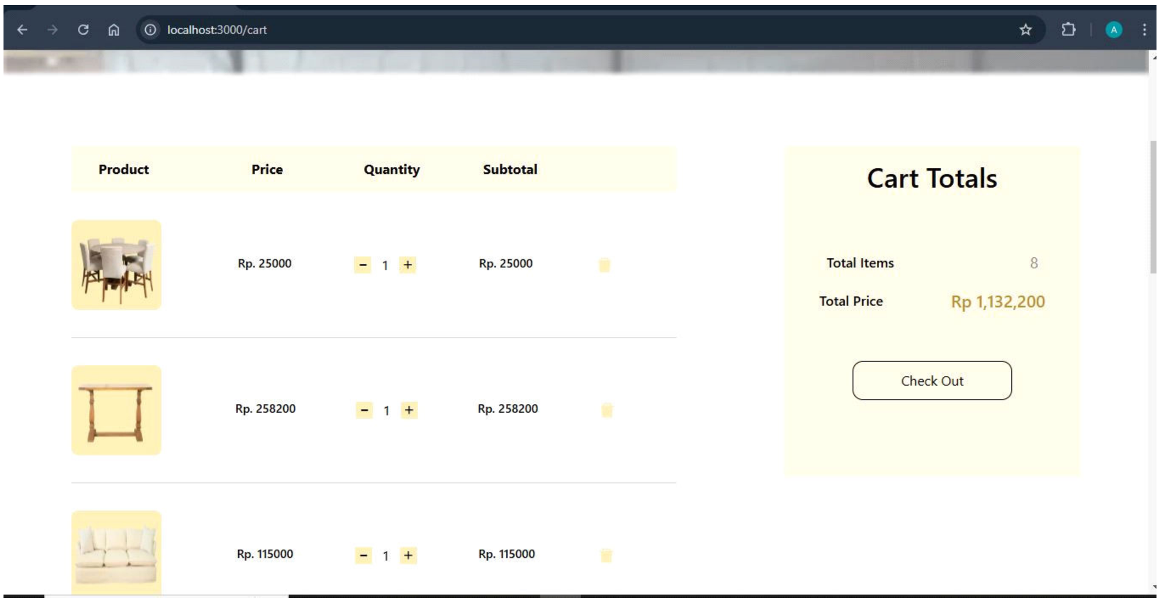
Stock 71

A chic chair and table set that effortlessly combines style and functionality. Ideal for cozy nooks or casual dining areas.

Size

L XL XS

Cart page component with dynamic prices and cart totals.



localhost:3000/cart

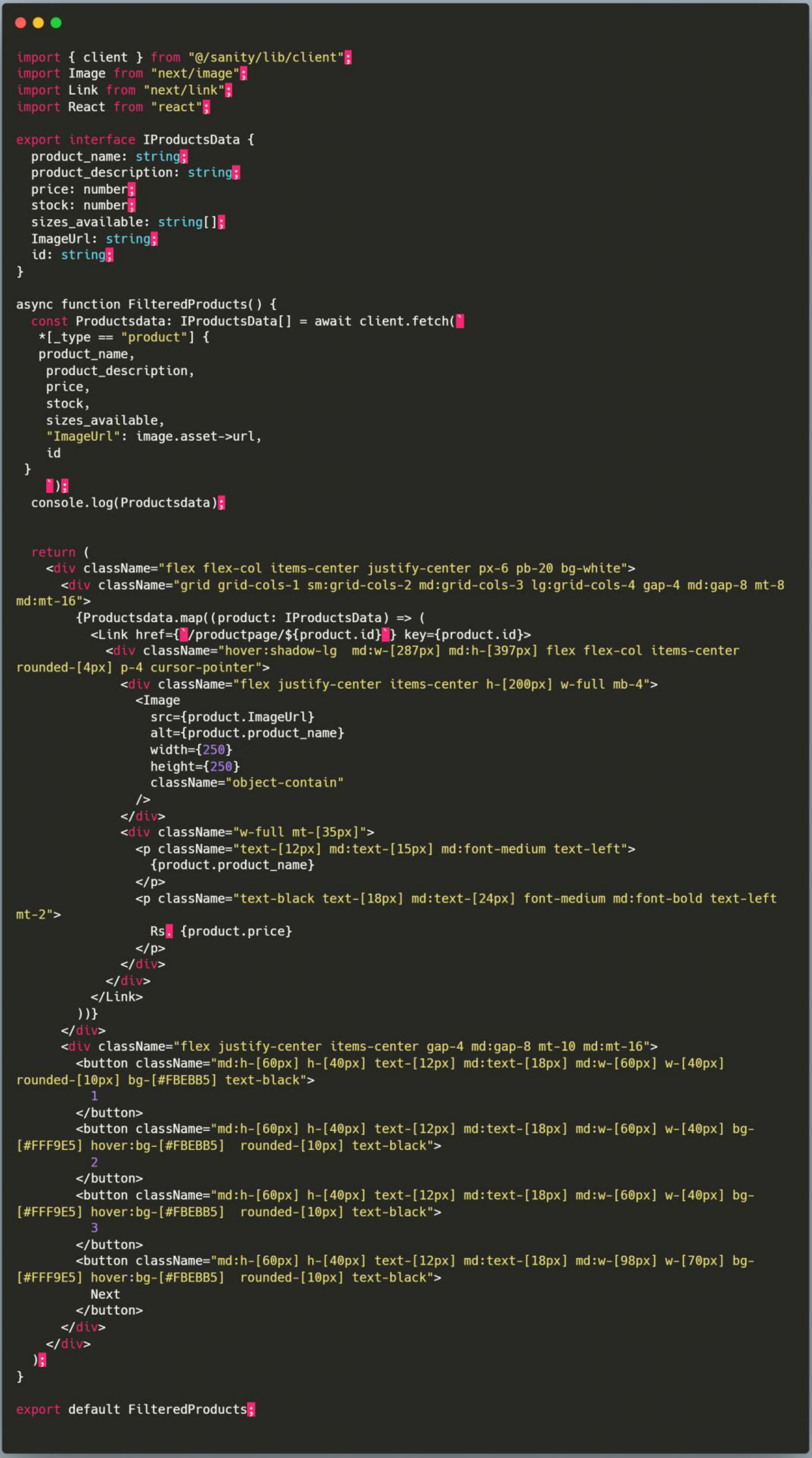
| Product | Price | Quantity | Subtotal |
|---|---|---|---|
|  | Rp. 25000 | − 1 + | Rp. 25000 |
|  | Rp. 258200 | − 1 + | Rp. 258200 |
|  | Rp. 115000 | − 1 + | Rp. 115000 |

Cart Totals

| | |
|---|---|
| Total Items | 8 |
| Total Price | Rp 1,132,200 |

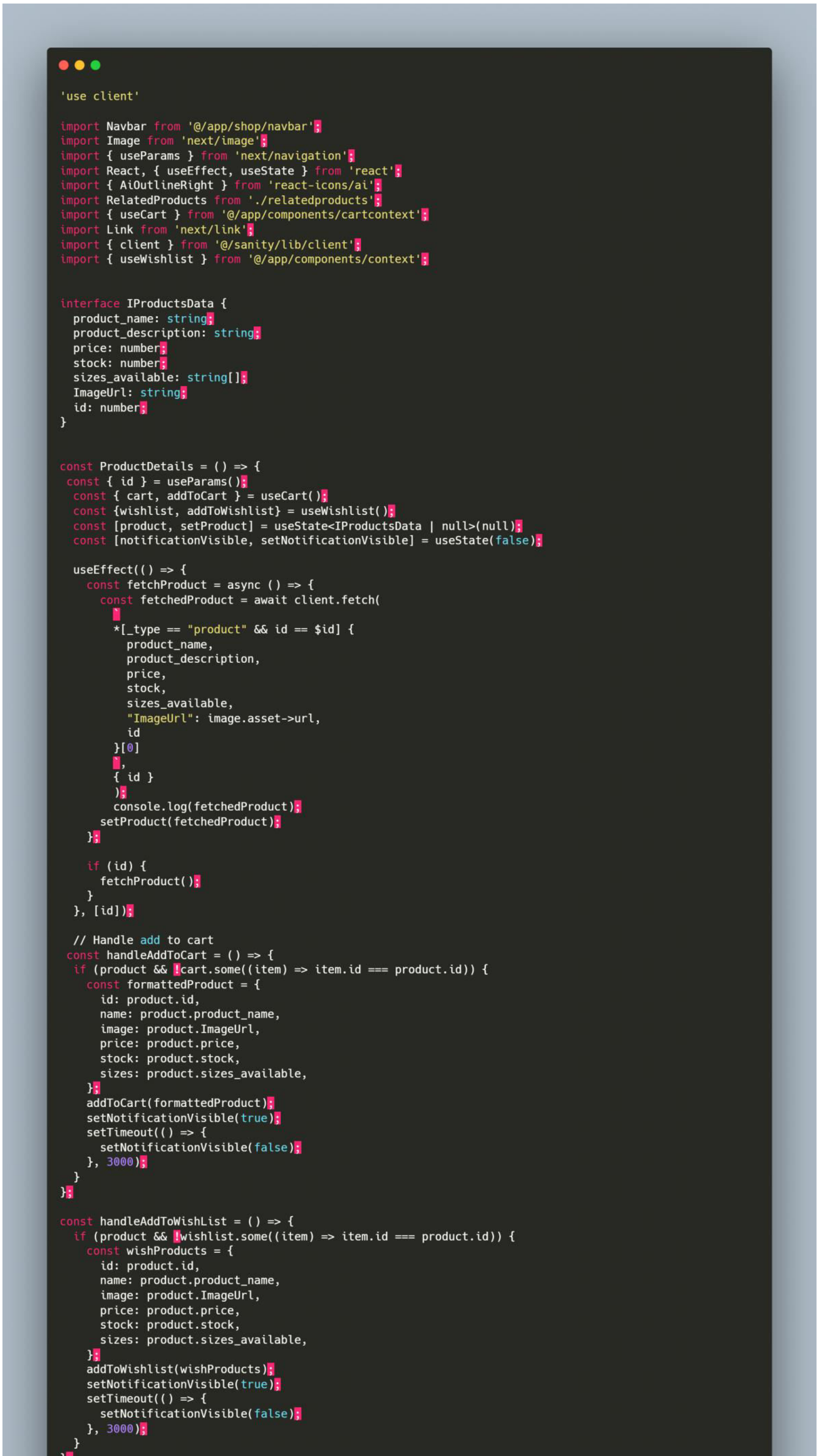
2. **Code Deliverables:**

Product listing page



```
import { client } from "@/sanity/lib/client";
import Image from "next/image";
import Link from "next/link";
import React from "react";

export interface IProductsData {
  product_name: string;
  product_description: string;
  price: number;
  stock: number;
  sizes_available: string[];
  ImageUrl: string;
  id: string;
}

async function FilteredProducts() {
  const Productsdata: IProductsData[] = await client.fetch(`
   *[_type == "product"] {
   product_name,
    product_description,
    price,
    stock,
    sizes_available,
    "ImageUrl": image.asset->url,
    id
 }
    `);
  console.log(Productsdata);


  return (
    <div className="flex flex-col items-center justify-center px-6 pb-20 bg-white">
      <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-4 md:gap-8 mt-8
md:mt-16">
        {Productsdata.map((product: IProductsData) => (
          <Link href={`/productpage/${product.id}`} key={product.id}>
            <div className="hover:shadow-lg  md:w-[287px] md:h-[397px] flex flex-col items-center
rounded-[4px] p-4 cursor-pointer">
              <div className="flex justify-center items-center h-[200px] w-full mb-4">
                <Image
                  src={product.ImageUrl}
                  alt={product.product_name}
                  width={250}
                  height={250}
                  className="object-contain"
                />
              </div>
              <div className="w-full mt-[35px]">
                <p className="text-[12px] md:text-[15px] md:font-medium text-left">
                  {product.product_name}
                </p>
                <p className="text-black text-[18px] md:text-[24px] font-medium md:font-bold text-left
mt-2">

                  Rs. {product.price}
                </p>
              </div>
            </div>
          </Link>
        ))}
      </div>
      <div className="flex justify-center items-center gap-4 md:gap-8 mt-10 md:mt-16">
        <button className="md:h-[60px] h-[40px] text-[12px] md:text-[18px] md:w-[60px] w-[40px]
rounded-[10px] bg-[#FBEBB5] text-black">
          1
        </button>
        <button className="md:h-[60px] h-[40px] text-[12px] md:text-[18px] md:w-[60px] w-[40px] bg-
[#FFF9E5] hover:bg-[#FBEBB5]  rounded-[10px] text-black">
          2
        </button>
        <button className="md:h-[60px] h-[40px] text-[12px] md:text-[18px] md:w-[60px] w-[40px] bg-
[#FFF9E5] hover:bg-[#FBEBB5]  rounded-[10px] text-black">
          3
        </button>
        <button className="md:h-[60px] h-[40px] text-[12px] md:text-[18px] md:w-[98px] w-[70px] bg-
[#FFF9E5] hover:bg-[#FBEBB5]  rounded-[10px] text-black">
          Next
        </button>
      </div>
    </div>
  );
}

export default FilteredProducts;
```

Product detail page



```
'use client'

import Navbar from '@/app/shop/navbar';
import Image from 'next/image';
import { useParams } from 'next/navigation';
import React, { useEffect, useState } from 'react';
import { AiOutlineRight } from 'react-icons/ai';
import RelatedProducts from './relatedproducts';
import { useCart } from '@/app/components/cartcontext';
import Link from 'next/link';
import { client } from '@/sanity/lib/client';
import { useWishlist } from '@/app/components/context';


interface IProductsData {
  product_name: string;
  product_description: string;
  price: number;
  stock: number;
  sizes_available: string[];
  ImageUrl: string;
  id: number;
}


const ProductDetails = () => {
 const { id } = useParams();
  const { cart, addToCart } = useCart();
  const {wishlist, addToWishlist} = useWishlist();
  const [product, setProduct] = useState<IProductsData | null>(null);
  const [notificationVisible, setNotificationVisible] = useState(false);

  useEffect(() => {
    const fetchProduct = async () => {
      const fetchedProduct = await client.fetch(
        `
        *[_type == "product" && id == $id] {
          product_name,
          product_description,
          price,
          stock,
          sizes_available,
          "ImageUrl": image.asset->url,
          id
        }[0]
        `,
        { id }
        );
        console.log(fetchedProduct);
      setProduct(fetchedProduct);
    };

    if (id) {
      fetchProduct();
    }
  }, [id]);

  // Handle add to cart
 const handleAddToCart = () => {
  if (product && !cart.some((item) => item.id === product.id)) {
    const formattedProduct = {
      id: product.id,
      name: product.product_name,
      image: product.ImageUrl,
      price: product.price,
      stock: product.stock,
      sizes: product.sizes_available,
    };
    addToCart(formattedProduct);
    setNotificationVisible(true);
    setTimeout(() => {
      setNotificationVisible(false);
    }, 3000);
  }
};

const handleAddToWishList = () => {
  if (product && !wishlist.some((item) => item.id === product.id)) {
    const wishProducts = {
      id: product.id,
      name: product.product_name,
      image: product.ImageUrl,
      price: product.price,
      stock: product.stock,
      sizes: product.sizes_available,
    };
    addToWishlist(wishProducts);
    setNotificationVisible(true);
    setTimeout(() => {
      setNotificationVisible(false);
    }, 3000);
  }
```

## 3. Report:

### Steps Taken

- Created reusable components (e.g., ProductCard, ProductList, SearchBar, Pagination).
- Integrated API to fetch product data dynamically from sanity CMS.
- Implemented dynamic routing for individual product detail pages using [id].

### Best Practices

- Followed component reusability principles for cleaner code.
- Used a modular folder structure for better scalability.
- Implemented lazy loading to enhance performance and reduce initial page load time.
- Added toast notifications for cart and wishlist for better user experience.

## Conclusion:

By integrating these best practices, the project achieved a well-structured and efficient frontend architecture. Component reusability and a modular folder structure enhanced maintainability and scalability, while performance optimizations like lazy loading and API caching improved user experience. These principles not only streamlined development but also laid a strong foundation for future enhancements.