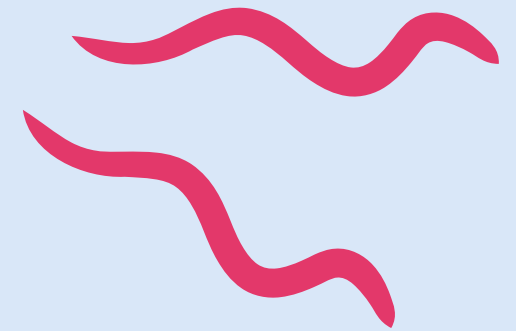


Day- 4

DYNAMIC FRONTEND COMPONENTS

shop.co





OVERVIEW

On Day 4, I implemented key features to enhance the functionality of my e-commerce website, Shop.co, ensuring a dynamic and user-friendly experience. Below are the details of the work completed



Features Implemented

Product Listing Page:
Product Detail Page:
Add to Cart Functionality :
Responsive Design with
Tailwind cs

Technical Work Flow

Product Listing Page



Fetches dynamic data from Sanity CMS to populate the product grid. Integrated category filters, a search bar, to allow users to browse and locate products seamlessly.

Category Page:



Dynamically fetches products based on their categories using Sanity CMS. Renders category-specific product listings for better user navigation.

Technical Work Flow

Product Detail Page:



Implemented individual product detail pages with accurate dynamic routing using Next.js. Each detail page renders data dynamically from Sanity CMS, including

Add to cart functionality



Developed a Add To cart Functionality With Local storage to update cart

Technical Work Flow

Responsive design:



Thoroughly tested and adjusted the layout for different screen sizes, ensuring consistent functionality and aesthetics.



BEST PRACTICES FOLLOWED

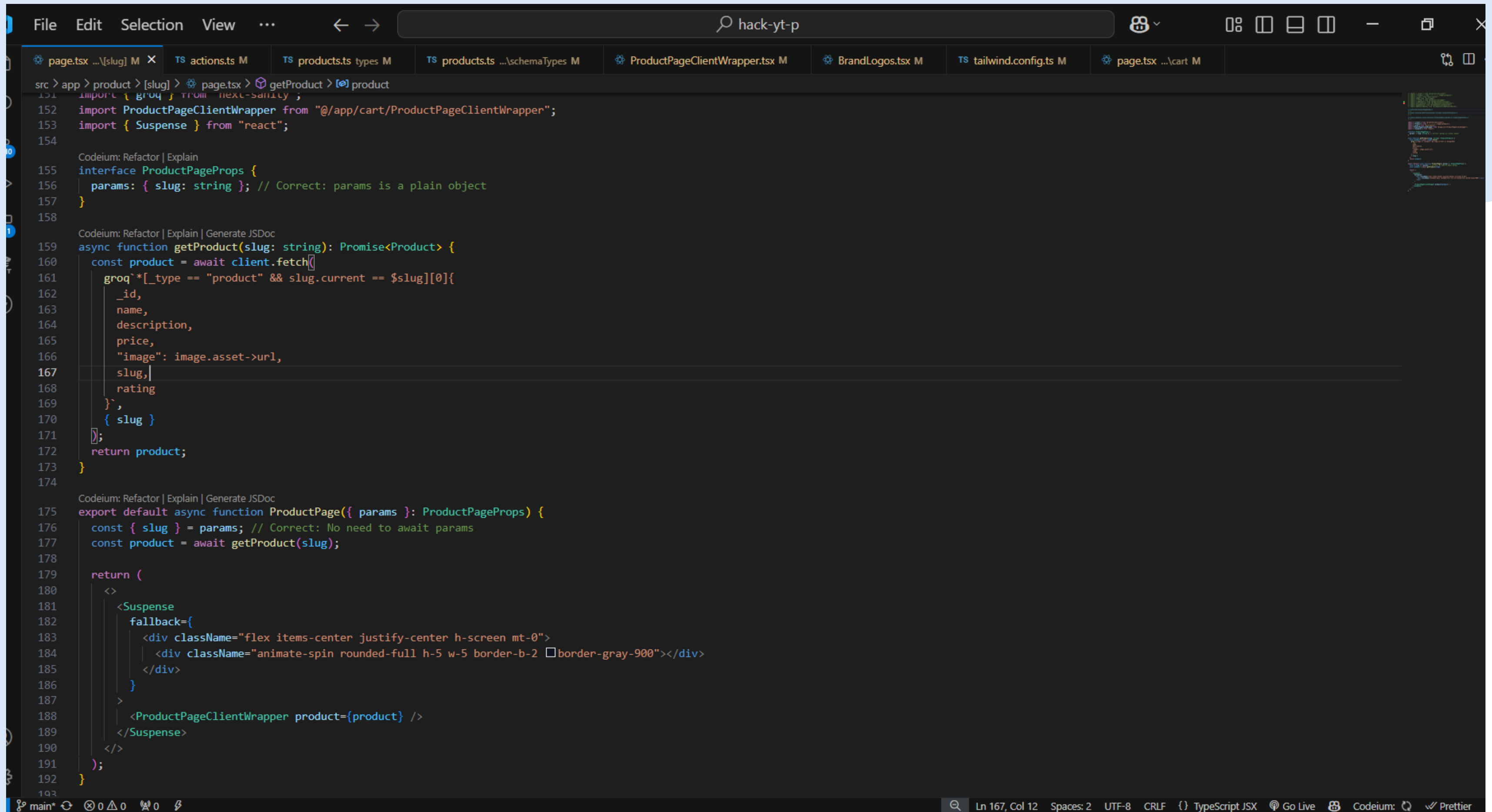
Ensured code modularity by separating concerns into dedicated folders and files. Followed clean code principles, making the project easy to maintain and extend. Implemented efficient state management using Redux Toolkit and redux-persist. Tested the application thoroughly to ensure responsiveness and functionality across devices

Product Listing Page

```
1 "use client"
2 import { Button } from "@/components/ui/button";
3 import { client } from "@/sanity/lib/client";
4 import { urlFor } from "@/sanity/lib/image";
5 import Image from "next/image";
6 import Link from "next/link";
7 import { FaStar } from "react-icons/fa";
8 import { useEffect, useState } from "react";
9
10 // Adding key prop in star array
11 let star = [
12   <FaStar key={1} />,
13   <FaStar key={2} />,
14   <FaStar key={3} />,
15   <FaStar key={4} />,
16   <FaStar key={5} />,
17 ];
18
19 interface Iproducts {
20   image: string;
21   discountPercent: number;
22   isNew: boolean;
23   name: string;
24   description: string;
25   price: number;
26   _id: string;
27 }
28
29 export default function Product() {
30   const [products, setProducts] = useState<Iproducts[]>([]);
31
32   useEffect(() => {
33     const fetchProducts = async () => {
34       try {
35         const fetchedProducts = await client.fetch(
36           `*[_type == 'products' && category == 'tshirt']{
37             "image": image.asset->url,
38             category,
39             discountPercent,
40             isNew,
41             name,
42             description,
43             price,
44             _id
45           }[0...4]`
46         );
47         setProducts(fetchedProducts);
48       } catch (error) {
49         console.error("Error fetching products:", error);
50       }
51     };
52   });
```

```
44     _id
45   }[0...4]`
46   );
47   setProducts(fetchedProducts);
48 } catch (error) {
49   console.error("Error fetching products:", error);
50 }
51 };
52
53 fetchProducts();
54 }, []);
55
56 return (
57   <>
58     <div className="w-full h-full mt-10 lg:mt-36 max-w-screen-xl mx-auto">
59       <h1 className="text-3xl md:text-4xl font-bold text-center">NEW ARRIVALS</h1>
60       <div className="relative mt-10 overflow-x-auto flex space-x-5 px-8">
61         {products.length > 0 ? (
62           products.map((data) => {
63             return (
64               <div key={data._id} className="flex-shrink-0">
65                 <Link href={`/product/${data._id}`}>
66                   <div className="w-[200px] md:w-[283px] h-[200px] md:h-[290px] bg-[#f0f0f0] rounded-[20px]">
67                     {data.image && (
68                       <Image
69                         src={urlFor(data.image).url()}
70                         alt={data.name}
71                         className="w-full h-full rounded-[20px]"
72                         width={100}
73                         height={100}
74                       />
75                     )}
76                   </div>
77                 </Link>
78                 <div className="pl-2">
79                   <p className="text-lg mt-2 font-bold">{data.name}</p>
80                   <div className="flex text-yellow-400">
81                     {star.map((icon, index) => (
82                       <span key={index}>{icon}</span>
83                     ))}
84                   </div>
85                   <p className="font-bold mt-1">
86                     {data.price}{" "}
87                     <span className="text-gray-400 font-bold line-through">
88                       {data.discountPercent}
89                     </span>
90                   </p>
91                 </div>
92               </div>
93             );
94           })
95         ) : (
96           <p className="text-center text-gray-500 w-full">
97             No products available at the moment. Please try again later.
98           </p>
99         )}
100       </div>
101     </div>
102     <div className="flex justify-center items-start mt-5">
103       <Link href="/casual">
104         <Button
105           variant={"outline"}
106           className="sm:mt-0 w-[80%] sm:w-[200px] rounded-[20px]"
107         >
108           View all
109         </Button>
110       </Link>
111     </div>
112   </>
113 );
114 }
115 }
```

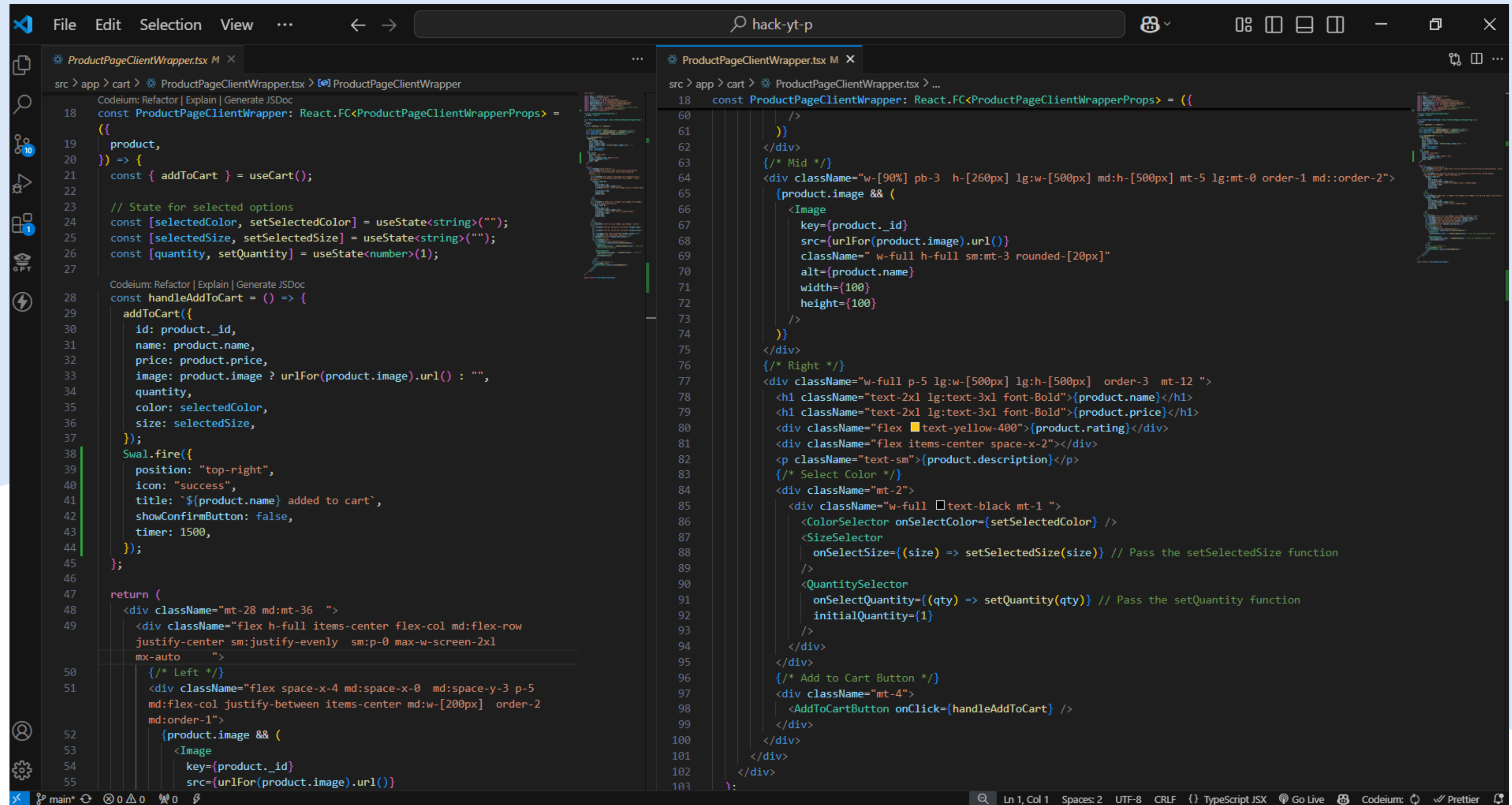

Dynamic Routing Page



```
File Edit Selection View ... hack-yt-p
page.tsx ...[slug] M TS actions.ts M TS products.ts types M TS products.ts ...schemaTypes M ProductPageClientWrapper.tsx M BrandLogos.tsx M TS tailwind.config.ts M page.tsx ...[cart] M

src > app > product > [slug] > page.tsx > getProduct > product
151 import { groq } from "next-sanity";
152 import ProductPageClientWrapper from "@app/cart/ProductPageClientWrapper";
153 import { Suspense } from "react";
154
Codeium: Refactor | Explain
155 interface ProductPageProps {
156   params: { slug: string }; // Correct: params is a plain object
157 }
158
Codeium: Refactor | Explain | Generate JSDoc
159 async function getProduct(slug: string): Promise<Product> {
160   const product = await client.fetch<
161     groq`*[_type == "product" && slug.current == $slug][0]{
162       _id,
163       name,
164       description,
165       price,
166       "image": image.asset->url,
167       slug,
168       rating
169     }`,
170     { slug }
171   >;
172   return product;
173 }
174
Codeium: Refactor | Explain | Generate JSDoc
175 export default async function ProductPage({ params }: ProductPageProps) {
176   const { slug } = params; // Correct: No need to await params
177   const product = await getProduct(slug);
178
179   return (
180     <>
181       <Suspense
182         fallback={
183           <div className="flex items-center justify-center h-screen mt-0">
184             <div className="animate-spin rounded-full h-5 w-5 border-b-2 border-gray-900"></div>
185           </div>
186         >
187       <ProductPageClientWrapper product={product} />
188     </Suspense>
189   </>
190 );
191 }
192
193
main* 0 0 0 0 Ln 167, Col 12 Spaces: 2 UTF-8 CRLF {} TypeScript JSX Go Live Codeium: Prettier
```

cart slice



```
src > app > cart > ProductPageClientWrapper.tsx > ProductPageClientWrapper
Codeium: Refactor | Explain | Generate JSDoc
18 const ProductPageClientWrapper: React.FC<ProductPageClientWrapperProps> =
19 ({
20   product,
21 }) => {
22   const { addToCart } = useCart();
23
24   // State for selected options
25   const [selectedColor, setSelectedColor] = useState<string>("");
26   const [selectedSize, setSelectedSize] = useState<string>("");
27   const [quantity, setQuantity] = useState<number>(1);
28
29   Codeium: Refactor | Explain | Generate JSDoc
30   const handleAddToCart = () => {
31     addToCart({
32       id: product._id,
33       name: product.name,
34       price: product.price,
35       image: product.image ? urlFor(product.image).url() : "",
36       quantity,
37       color: selectedColor,
38       size: selectedSize,
39     });
40     Swal.fire({
41       position: "top-right",
42       icon: "success",
43       title: `${product.name} added to cart`,
44       showConfirmButton: false,
45       timer: 1500,
46     });
47   };
48   return (
49     <div className="mt-28 md:mt-36">
50       <div className="flex h-full items-center flex-col md:flex-row
51         justify-center sm:justify-evenly sm:p-0 max-w-screen-2xl
52         mx-auto">
53         { /* Left */ }
54         <div className="flex space-x-4 md:space-x-0 md:space-y-3 p-5
55           md:flex-col justify-between items-center md:w-[200px] order-2
56           md:order-1">
57           {product.image && (
58             <Image
59               key={product._id}
60               src={urlFor(product.image).url()}
61             />
62           )}
63         </div>
64       </div>
65     </div>
66   );
67 }
68
69 export default ProductPageClientWrapper;
```

```
src > app > cart > ProductPageClientWrapper.tsx > ...
18 const ProductPageClientWrapper: React.FC<ProductPageClientWrapperProps> = ({
19   product,
20 }) => {
21   <div className="w-[90%] pb-3 h-[260px] lg:w-[500px] md:h-[500px] mt-5 lg:mt-0 order-1 md::order-2">
22     {product.image && (
23       <Image
24         key={product._id}
25         src={urlFor(product.image).url()}
26         className="w-full h-full sm:mt-3 rounded-[20px]"
27         alt={product.name}
28         width={100}
29         height={100}
30       />
31     )}
32   </div>
33   { /* Mid */ }
34   <div className="w-full p-5 lg:w-[500px] lg:h-[500px] order-3 mt-12">
35     <h1 className="text-2xl lg:text-3xl font-Bold">{product.name}</h1>
36     <h1 className="text-2xl lg:text-3xl font-Bold">{product.price}</h1>
37     <div className="flex text-yellow-400">{product.rating}</div>
38     <div className="flex items-center space-x-2"></div>
39     <p className="text-sm">{product.description}</p>
40     { /* Select Color */ }
41     <div className="mt-2">
42       <div className="w-full text-black mt-1">
43         <ColorSelector onSelectColor={setSelectedColor} />
44         <SizeSelector
45           onSelectSize={(size) => setSelectedSize(size)} // Pass the setSelectedSize function
46         />
47         <QuantitySelector
48           onSelectQuantity={(qty) => setQuantity(qty)} // Pass the setQuantity function
49           initialQuantity={1}
50         />
51       </div>
52     </div>
53     { /* Add to Cart Button */ }
54     <div className="mt-4">
55       <AddToCartButton onClick={handleAddToCart} />
56     </div>
57   </div>
58 }
59
60 export default ProductPageClientWrapper;
```

Product Listing

Apps

W3

Gmail

GA-RWD

WhatsApp

F

45

ipm/Home

eng

Chat


apna

GitHub

(Par2)

All Bookmarks


TOP SELLING



Casual Green Bomber Jacket

★★★★★


\$300.00 20%



COURAGE GRAPHIC T-SHIRT

★★★★★


\$145.00



Checkered Shirt

★★★★★

\$178.00





Beige Slim-Fit Jogger Pants


★★★★★


\$269.00 10%

View all










Beige Slim-Fit Jogger Pants


★★★★★


269 242:1 -10%


These beige jogger pants combine comfort and style with their relaxed yet modern fit. Designed with an elastic waistband and cuffed ankles, they provide a sleek, slim silhouette while offering a casual, sporty look. The soft fabric ensures all-day comfort, making them perfect for lounging or running errands. The neutral beige color adds versatility, allowing for easy pairing with a variety of tops. Whether you're pairing them with a t-shirt or a hoodie, these joggers are a must-have staple for a stylish, laid-back outfit.

Select Colors









Choose Size

M

L

S

XXL

- 1 +

Add to Cart

Thank
you