

Technical Report: Dynamic Routing and Modular Component Design in React and Next.js

Objective

This project demonstrates the implementation of a dynamic routing system for products in a Next.js application. The routing is based on product titles fetched from **Sanity CMS**, and the project emphasizes modularity and usability through reusable components such as **ProductCard**, **SearchBar**, and pagination handling in the main **Page** component.

Key Features and Design Decisions

1. Dynamic Routing for Product Details

Implementation

- The route for each product is dynamically generated using the product's **title** as a key. This ensures unique URLs for each product, improving SEO and user experience.

Code snippet for dynamic routing: (Appendix 1)

Key Functionality

- Fetching Data:** Uses Sanity's GROQ query to fetch product details based on the title parameter passed in the URL.
 - Fallback Handling:** Displays a **404** page if the product is not found, ensuring a user-friendly error state.
 - Dynamic Rendering:** Product details, such as title, description, price, tags, and discount percentage, are rendered dynamically based on the fetched data.
-

2. Reusable Components

ProductCard

- A reusable component designed to display a preview of product information in a grid layout.
- Highlights:
 - Displays product image, title, description, price, and discount.
 - Provides a "New" badge for new products.
 - Includes a modal that opens on clicking the card to show detailed product information.

Code snippet: (Appendix 2)

SearchBar

- A component to filter products based on the user's input.
- Highlights:
 - Filters products in real-time.
 - Returns filtered results to the parent component for rendering.

Code snippet: (Appendix 3)

3. Main Page with Pagination

- Displays a grid of **ProductCard** components along with pagination and other components such as Navbar and Footer.
- Highlights:
 - Fetches all products from Sanity CMS on load.
 - Implements pagination logic to divide the products into pages, each containing 8 items.
 - Integrates the **SearchBar** for filtering.

Code snippet: (Appendix 4)

Conclusion

This project successfully creates a scalable and user-friendly e-commerce interface. Dynamic routing ensures flexibility, while modular components like **ProductCard** and **SearchBar** make the codebase maintainable and extensible. The integration with Sanity CMS enables easy content management, making this implementation ideal for modern web applications.

Appendix - 1:

```
1 import { client } from "@sanity/lib/client";
2 import { notFound } from "next/navigation";
3
4 interface ProductDetailsProps {
5   params: {
6     title: string;
7   };
8 }
9
10 export default async function ProductDetails({ params }: ProductDetailsProps) {
11   const { title } = params;
12
13   const product = await client.fetch(
14     `*[_type == "product" && lower(title) == ${title}][0]{
15       title,
16       "imageUrl": productImage.asset->url,
17       description,
18       price,
19       tags,
20       discountPercentage,
21       isNew
22     }`,
23     { title: title.toLowerCase() }
24   );
25
26   if (!product) {
27     return notFound();
28   }
29
30   return (
31     <div className="container mx-auto p-4">
32       <div className="flex flex-col md:flex-row items-center">
33         <img
34           src={product.imageUrl || "/default-image.jpg"} // Use fallback image
35           alt={product.title}
36           className="w-full md:w-1/2 rounded-lg shadow-lg"
37         />
38         <div className="md:ml-8 mt-4 md:mt-0">
39           <h1 className="text-3xl font-bold">{product.title}</h1>
40           <p className="text-lg mt-2">{product.description}</p>
41           <p className="text-xl font-semibold mt-4">${product.price}</p>
42           {product.discountPercentage && (
43             <p className="text-green-500 mt-2">
44               Discount: {product.discountPercentage}%
45             </p>
46           )}
47           {product.isNew && <p className="text-blue-500 mt-2">New Arrival!</p>}
48           {product.tags?.length > 0 && (
49             <div className="mt-4">
50               <h4 className="font-semibold">Tags:</h4>
51               <ul className="flex space-x-2">
52                 {product.tags.map((tag: string, index: number) => (
53                   <li key={index} className="bg-gray-200 px-2 py-1 rounded-lg">
54                     {tag}
55                   </li>
56                 ))}
57               </ul>
58             </div>
59           )}
60         </div>
61       </div>
62     </div>
63   );
64 }
65
```

Appendix - 2:

```
1 import React, { useState } from "react";
2 import Image from "next/image";
3
4 interface ProductCardProps {
5   product: {
6     title: string;
7     description: string;
8     price: number;
9     productImage: {
10       asset?: {
11         id?: string;
12         url?: string;
13       };
14     };
15     discountPercentage?: number;
16     isNew?: boolean;
17   };
18 }
19
20 const ProductCard: React.FC<ProductCardProps> = ({ product }) => {
21   const [isModalOpen, setIsModalOpen] = useState(false);
22
23   const handleProductClick = () => {
24     setIsModalOpen(true);
25   };
26
27   const imageUrl =
28     product.productImage.asset?.url || "/image-4.png";
29
30   return (
31     <div
32       className="card border shadow-lg rounded-lg overflow-hidden cursor:pointer"
33       onClick={handleProductClick}
34     >
35       <img
36         src={imageUrl}
37         alt={product.title}
38         className="object-cover w-full h-[200px]"
39       />
40       <div className="card-body p-4">
41         <h3 className="card-title font-bold text-lg">{product.title}</h3>
42         <p className="card-text text-gray-600">
43           {product.description.substring(0, 80)}...
44         </p>
45         <div className="card-title font-bold text-lg mt-2 text-blue-500">
46           {product.discountPercentage ? (
47             <div>
48               Discounted: $
49               {
50                 ((product.price * (1 - product.discountPercentage / 100)).toFixed(2))
51               }
52               <span className="line-through text-gray-500">
53                 Original: ${product.price}
54               </span>
55             </div>
56           ) : (
57             Price: ${product.price}
58           )
59         </div>
60         {product.isNew && (
61           <span className="bg-green-500 text-white px-2 py-1 rounded-full inline-block mt-2">
62             New
63           </span>
64         )}
65       </div>
66     </div>
67   );
68
69   if (isModalOpen) {
70     <div
71       className="fixed inset-0 bg-black bg-opacity-50 flex items-center justify-center z-50"
72       >
73       <div
74         className="bg-white p-6 rounded shadow-lg overflow-y-auto max-h-screen relative w-[90%] md:w-[70%] lg:w-[50%]"
75       >
76         <button
77           className="absolute top-4 right-4 text-gray-500 hover:text-gray-700 bg-gray-200 rounded-full p-2"
78           onClick={() => setIsModalOpen(false)}
79         >
80           <times>
81         </button>
82         <h2 className="text-2xl font-bold mb-4">{product.title}</h2>
83         <img
84           alt={product.title}
85           src={imageUrl}
86           width={500}
87           height={400}
88           className="rounded mb-4 w-full h-[300px] object-cover"
89         />
90         <div className="mb-4">
91           <h3 className="text-lg font-semibold mb-1">Description:</h3>
92           <p>{product.description}</p>
93         </div>
94         <div className="mb-4">
95           <h3 className="text-lg font-semibold mb-1">Price:</h3>
96           <p>
97             {product.discountPercentage ? (
98               <div>
99                 Discounted Price: $
100                 {
101                   ((product.price * (1 - product.discountPercentage / 100)).toFixed(2))
102                 }
103                 <span>Original Price: ${product.price}</span>
104               </div>
105             ) : (
106               ${product.price}
107             )
108           </p>
109         </div>
110         {product.isNew && (
111           <div className="bg-green-500 text-white px-2 py-1 rounded-full inline-block">
112             New
113           </div>
114         )}
115       </div>
116     </div>
117   );
118 }
119
120 export default ProductCard;
```

Appendix - 3:

```
1 import React from "react";
2
3 interface SearchBarProps {
4   products: any[];
5   onSearchResults: (results: any[]) => void;
6 }
7
8 const SearchBar: React.FC<SearchBarProps> = ({ products, onSearchResults }) => {
9   const handleSearch = (query: string) => {
10     const results = products.filter((product) =>
11       product.title.toLowerCase().includes(query.toLowerCase())
12     );
13     onSearchResults(results);
14   };
15
16   return (
17     <div className="flex items-center justify-center w-full">
18       <div className="relative w-full max-w-xs">
19         <div className="absolute inset-y-0 left-0 flex items-center pl-3 pointer-events-none">
20           <svg className="w-4 h-4 text-gray-500 fill-none stroke-current" viewBox="0 0 24 24" xmlns="http://www.w3.org/2000/svg"><path stroke-linecap="round" stroke-join="round" stroke-width="2" d="M21 21-6-6a2 2 0 0 0 0 2.83 2 2 0 0 0 2.83 0l6 6a2 2 0 0 0 0 2.83 2 2 0 0 0 2.83 0l6-6a2 2 0 0 0 0 2.83 2 2 0 0 0 2.83 0l-6 6a2 2 0 0 0 0 2.83 2 2 0 0 0 2.83 0z"/></svg>
21         <input
22           type="text"
23           placeholder="Search..."
24           className="py-2 px-3 pl-10 rounded-md border border-gray-300 focus:ring-blue-500 focus:border-blue-500 w-full"
25           onChange={(e) => handleSearch(e.target.value)}
26         />
27       </div>
28     </div>
29   );
30 };
31
32 export default SearchBar;
```

Appendix - 4:

```
1 "use client";
2
3 import React, { useState, useEffect } from "react";
4 import { client } from "@/sanity/lib/client";
5 import Hero from "@/Components/Hero";
6 import Sec_Hero from "@/Components/Sec_Hero";
7 import ProductCard from "@/Components/ProductCard";
8 import SearchBar from "@/Components/SearchBar";
9 import Slide from "@/Components/Slide";
10 import ShareSec from "@/Components/ShareSec";
11
12 const Home = () => {
13   const [products, setProducts] = useState<any[]>([]);
14   const [filteredProducts, setFilteredProducts] = useState<any[]>([]);
15   const [currentPage, setCurrentPage] = useState(1);
16   const productsPerPage = 8;
17
18   useEffect(() => {
19     const fetchProducts = async () => {
20       const data = await client.fetch( `*[_type == "product"] ` );
21       setProducts(data);
22       setFilteredProducts(data);
23     };
24
25     fetchProducts();
26   }, []);
27
28   const handlePageChange = (page: number) => {
29     setCurrentPage(page);
30   };
31
32   const indexOfLastProduct = currentPage * productsPerPage;
33   const indexOfFirstProduct = indexOfLastProduct - productsPerPage;
34   const currentProducts = filteredProducts.slice(indexOfFirstProduct, indexOfLastProduct);
35
36   const totalPages = Math.ceil(filteredProducts.length / productsPerPage);
37
38   return (
39     <div>
40       <Hero />
41       <Sec_Hero />
42       <SearchBar products={products} onSearchResults={setFilteredProducts} />
43       <div className="container mx-auto mt-4">
44         <div className="grid grid-cols-3 sm:grid-cols-3 lg:grid-cols-4 gap-4">
45           {currentProducts.map((product) => (
46             <ProductCard key={product.title} product={product} />
47           ))}
48         </div>
49       </div>
50       <div className="container mx-auto mt-8">
51         <div className="flex justify-center mt-6 mb-6">
52           <button
53             onClick={() => handlePageChange(currentPage - 1)}
54             disabled={currentPage === 1}
55             className="px-4 py-2 bg-blue-500 text-white rounded disabled:bg-gray-400"
56           >
57             Previous
58           </button>
59           <span className="mx-4">
60             Page {currentPage} of {totalPages}
61           </span>
62           <button
63             onClick={() => handlePageChange(currentPage + 1)}
64             disabled={currentPage === totalPages}
65             className="px-4 py-2 bg-blue-500 text-white rounded disabled:bg-gray-400"
66           >
67             Next
68           </button>
69         </div>
70       </div>
71       <Slide />
72       <ShareSec />
73     </div>
74   );
75 };
76
77 export default Home;
```

