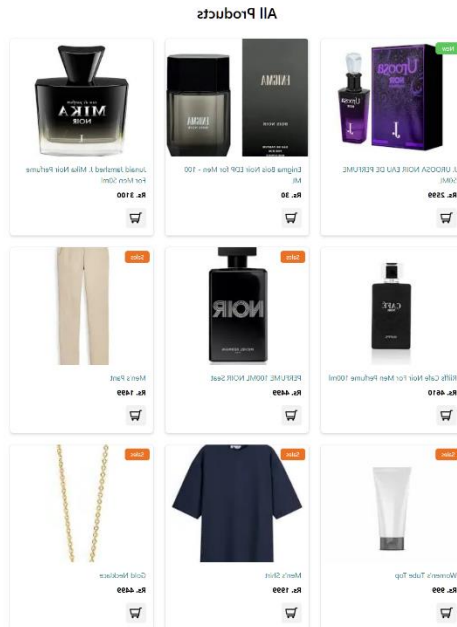
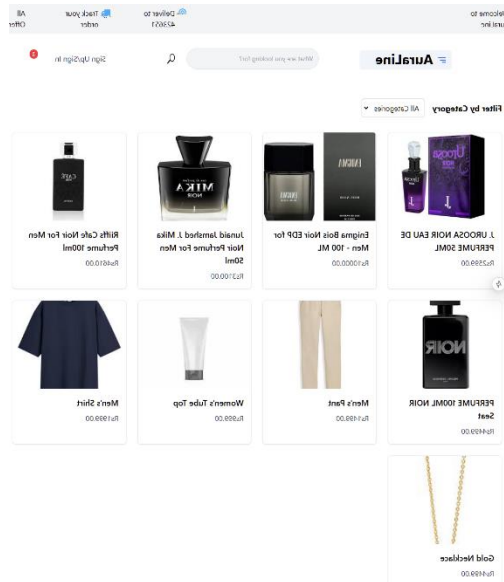
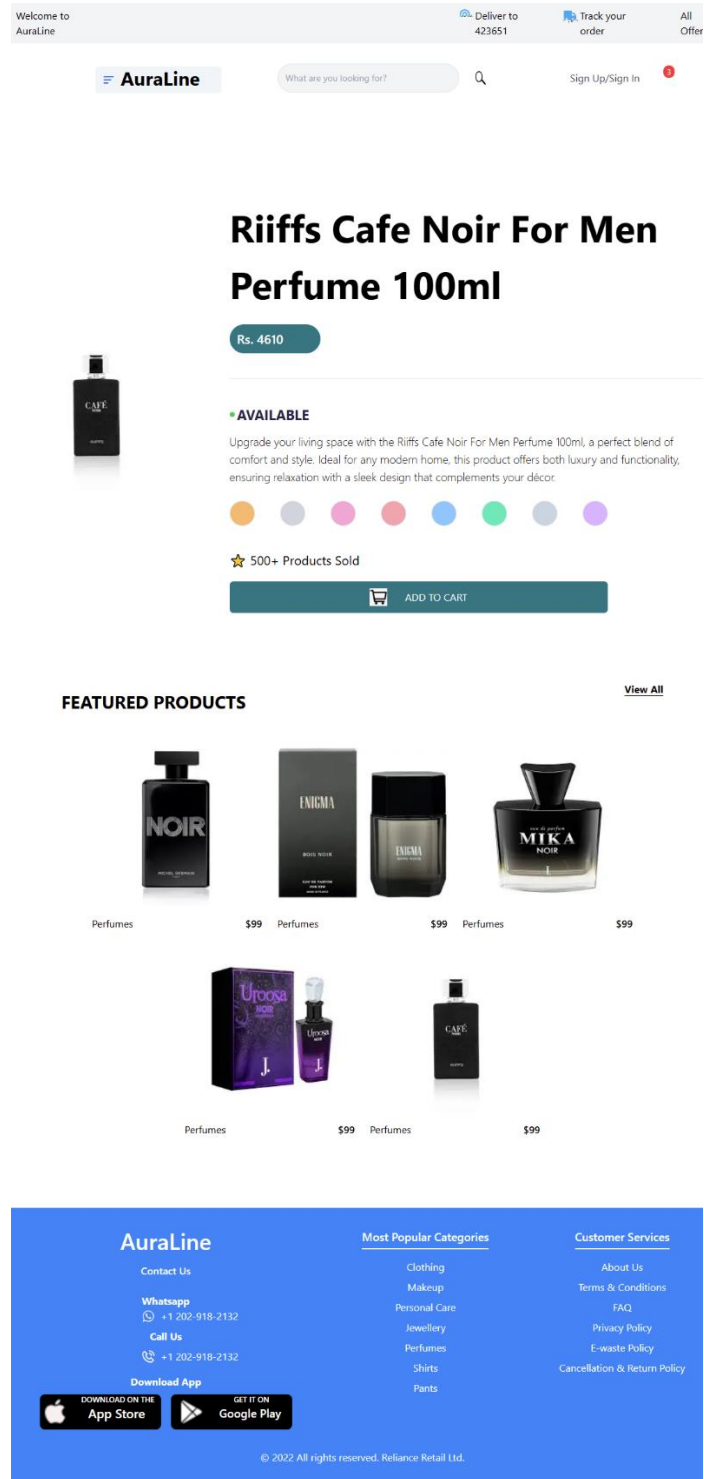


DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE-“ AURALINE”



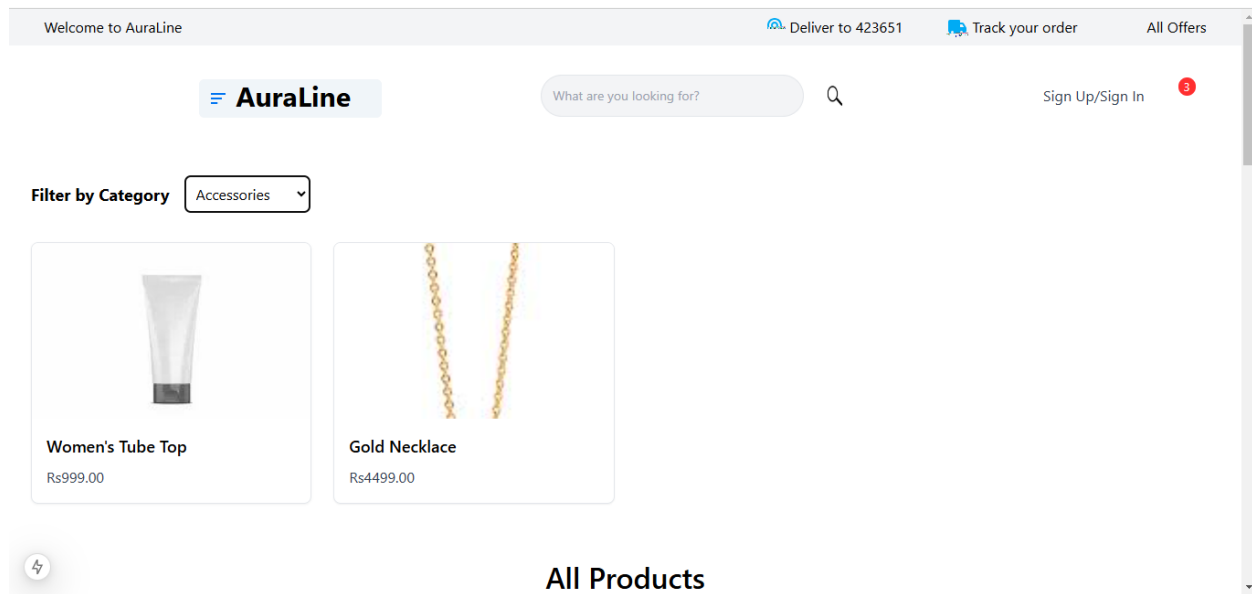
PRODUCT LISTING PAGE

DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE-“ AURALINE”

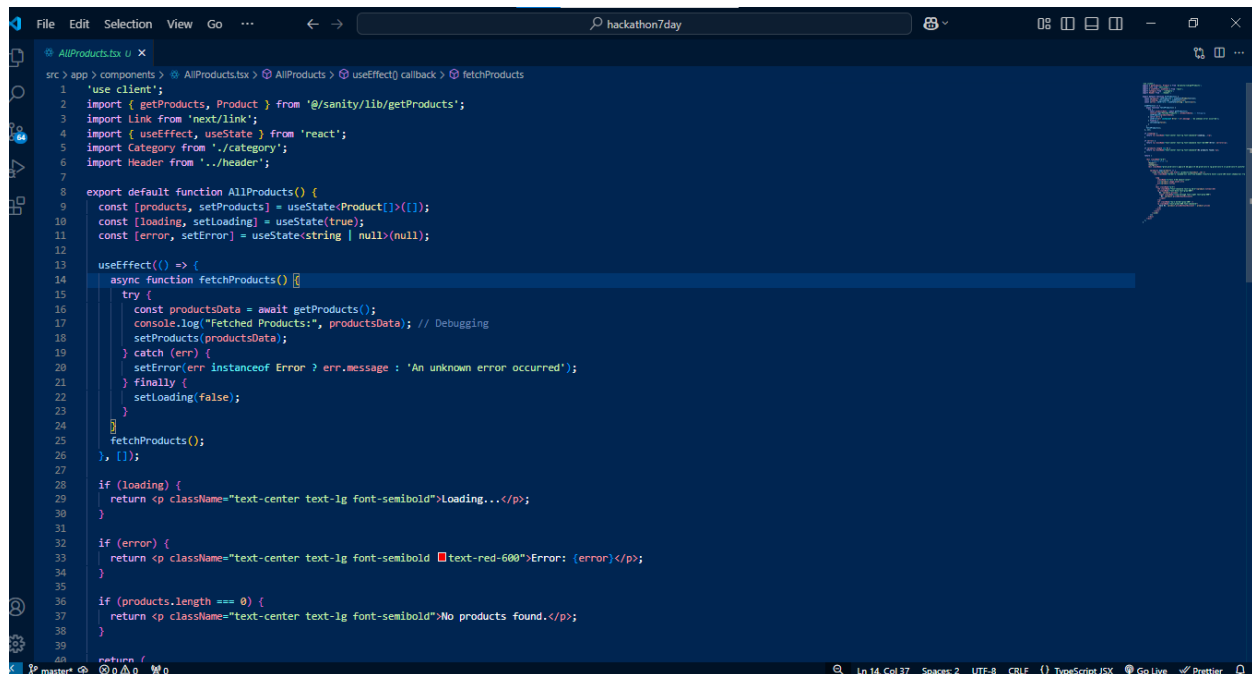


DYNAMIC PRODUCT RENDERING: <http://localhost:3000/productlisting/4>

DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE-“ AURALINE”



FILTERED BY CATEGORY



ALL PRODUCTS PAGE CODE

DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE-“ AURALINE”

```
File Edit Selection View Go ... hackathon7day
src > app > components > AllProducts.tsx > AllProducts
8 export default function AllProducts() {
40 return (
41   <div className="p-6">
42     /* Products Grid */
43     <Header/>
44     <Category/>
45     <div className="grid grid-cols-1 gap-y-6 md:gap-x-6 md:grid-cols-2 lg:grid-cols-3 xl:grid-cols-4 justify-around">
46       {products.map((product) => (
47         <Link key={product._id} href={`/products/${product._id}`}>
48           <div className="border-2 rounded-2xl overflow-hidden transform hover:scale-105 hover:shadow-2xl transition-transform duration-300 ease-in-out cursor-pointer">
49             <img
50               className="w-full h-48 object-cover"
51               src={product.image.asset.url}
52               alt={product.title}
53             />
54             <div className="p-4">
55               <h4 className="font-semibold text-lg mb-2">{product.title}</h4>
56               <p className="font-bold text-gray-800">
57                 Rs. {product.price}{' '}
58                 <span className="line-through font-light text-gray-500">
59                   Rs. {product.priceWithoutDiscount}
60                 </span>
61               </p>
62               <hr className="my-2 border-gray-200" />
63               <p className="text-green-600 font-semibold">
64                 Save Rs. {product.priceWithoutDiscount - product.price}
65               </p>
66             </div>
67           </Link>
68         </div>
69       ))}
70     </div>
71   </div>
72 );
73 }
74 }
75 }
76 }
```

ALL PRODUCTS PAGE CODE

```
File Edit Selection View Go ... hackathon7day
src > app > components > search.tsx > Product
1 "use client"
2
3 import { useState, useEffect } from "react";
4 import Link from "next/link"; // Import Link from Next.js
5
6 type Product = {
7   id: number;
8   name: string;
9   price: number;
10  imageUrl: string;
11  isNew: boolean;
12  salePrice: number;
13  label: string;
14  labelColor?: string;
15  category: string;
16 };
17
18 const Search = () => {
19   const [query, setQuery] = useState(""); // User's search input
20   const [results, setResults] = useState<Product[]>([]); // Search results
21   const [loading, setLoading] = useState(false); // Loading state
22
23   useEffect(() => {
24     const fetchResults = async () => {
25       if (!query) {
26         setResults([]); // Clear results when query is empty
27         return;
28       }
29       setLoading(true);
30       try {
31         const response = await fetch("/api/products");
32         const data: Product[] = await response.json();
33         console.log("Fetched products:", data);
34
35         // Filter products based on the query
36         const filteredResults = data.filter((product) =>
37           product.name.toLowerCase().includes(query.toLowerCase())
38         );
39       } catch (error) {
40         console.error("Error fetching products:", error);
41       }
42       setResults(filteredResults);
43       setLoading(false);
44     };
45     fetchResults();
46   }, [query]);
47
48   return (
49     <div>
50       <input type="text" value={query} onChange={setQuery} />
51       <button onClick={fetchResults}>Search</button>
52       <div>
53         {loading ? <div>Loading...</div> : results.map((product) => (
54           <Link href={`/products/${product._id}`}>
55             <div>
56               <img alt={product.imageUrl} />
57               <div>
58                 <h3>{product.name}</h3>
59                 <p>Rs. {product.price}</p>
60                 {product.isNew ? <p>New!</p> : <p>Sale!</p>}
61               </div>
62             </div>
63           </Link>
64         ))}
65       </div>
66     </div>
67   );
68 }
69
70 export default Search;
```

SEARCH COMPONENT CODE

DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE-“ AURALINE”

```
File Edit Selection View Go ... hackathon7day

src > app > components > search.tsx | use Product
18 const Search = () => {
23   useEffect(() => {
24     const fetchResults = async () => {
37       const filteredResults = data.filter(product =>
38         product.name.toLowerCase().includes(query.toLowerCase())
39       );
40       setResults(filteredResults);
41     } catch (error) {
42       console.error("Failed to fetch products:", error);
43     } finally {
44       setLoading(false);
45     }
46   });
47
48   const delayDebounce = setTimeout(() => {
49     fetchResults();
50   }, 300); // Debounce API calls by 300ms
51
52   return () => clearTimeout(delayDebounce); // Cleanup debounce
53 }, [query]);
54
55 return (
56   <div className="relative hidden md:flex items-center p-2 px-4 gap-x-5 text-gray-700 text-sm focus:outline-none border-none">
57     <input
58       type="text"
59       placeholder="What are you looking for?"
60       className="border rounded-full focus:outline-none md:w-72 p-3 bg-gray-100"
61       value={query}
62       onChange={(e) => setQuery(e.target.value)}
63     />
64     
65     <div className="absolute top-full left-0 right-0 loading...</div>
66     {results.length > 0 && (
67       <ul className="absolute top-full left-2 right-2 bg-white border shadow-lg z-50 p-2 px-2 w-52">
68         {results.map((product) => (
69           <li key={product.id} href={`/productlisting/${product.id}`} passHref>
70             <div className="p-3 hover:bg-gray-100 cursor-pointer z-10 text-[14px]>
71               {product.name}
72             </div>
73           </li>
74         ))}
75       </ul>
76     )}
77   </div>
78 );
79
80 export default Search;
```

SEARCH COMPONENT CODE

```
File Edit Selection View Go ... hackathon7day

src > app > productlisting > [id] > page.tsx | ProductDetail
1 "use client";
2
3 import { useEffect, useState } from "react";
4 import { useParams, notFound } from "next/navigation";
5 import Image from "next/image";
6 import Header from "@app/header";
7 import Footer from "@app/footer";
8
9 type Product = {
10   id: number;
11   name: string;
12   price: number | string;
13   imageUrl: string;
14   isSale: boolean;
15   salePrice?: number;
16   label?: string;
17   labelColor?: string;
18 };
19
20 export default function ProductDetail() {
21   const [product, setProduct] = useState<Product | null>(null);
22   const [loading, setLoading] = useState(true);
23   const [error, setError] = useState<string | null>(null);
24   const { id } = useParams();
25
26   useEffect(() => {
27     if (id) {
28       const fetchProduct = async () => {
29         try {
30           const response = await fetch("/api/products");
31           if (!response.ok) {
32             throw new Error("Failed to fetch products: " + response.status);
33           }
34           const products: Product[] = await response.json();
35           const foundProduct = products.find((p) => p.id.toString() === id);
36           if (!foundProduct) {
37             notFound();
38             return;
39           }
40           setProduct(foundProduct);
41         } catch (error) {
42           setError("Failed to fetch product. Please try again later.");
43         } finally {
44           setLoading(false);
45         }
46       };
47       fetchProduct();
48     }
49   });
```

DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE-“ AURALINE”

```
File Edit Selection View Go ... ↵ hackathon7day
```

```
< >
```

```
page.tsx | x
```

```
src > app > productListing [fd] > @ page.tsx > ProductDetail
20 export default function ProductDetail() {
25   useEffect(() => {
48     },
49     [, [id]]);
50
51   if (loading) {
52     return (
53       <div className="flex justify-center items-center h-screen">
54         <p className="text-lg font-semibold">Loading...</p>
55       </div>
56     );
57   }
58
59   if (error) {
60     return (
61       <div className="flex justify-center items-center h-screen">
62         <p className="text-lg font-semibold text-red-600">[error]</p>
63       </div>
64     );
65   }
66
67   if (!product) {
68     notFound();
69     return null;
70   }
71
72   type ColorKey = "orange" | "gray" | "pink" | "rose" | "blue" | "emerald" | "slate" | "purple";
73
74   const colorMap: Record<ColorKey, string> = {
75     orange: "bg-orange-300",
76     gray: "bg-gray-300",
77     pink: "bg-pink-300",
78     rose: "bg-rose-300",
79     blue: "bg-blue-300",
80     emerald: "bg-emerald-300",
81     slate: "bg-slate-300",
82     purple: "bg-purple-300",
83   };
84
85   return (
86     <main>
87       <header />
88       <div className="flex flex-col md:flex-row justify-center md:gap-x-20 at-32 px-4 md:px-8">
89         <div className="flex justify-center mb-10 md:mb-0">
90           <Image
91             src={product.imageUrl}
92             alt={product.name}
93             width={255}
```

```
Ln 178, Col 2 Spaces 2 UTF-8 CRLF TypeScript JSX Go Live Preview
```

```

src> app > production > [id] > @ pagetrix > @ ProductDetail
20 export default function ProductDetail() {
21   133     alt="Cart Icon"
22   134     width={28}
23   135     height={28}
24   136     className="md:w-7 md:h-7 transition-opacity duration-300 group-hover:opacity-0"
25   137   />
26   138   <Image
27   139     src="/cart.png"
28   140     alt="Cart icon"
29   141     width={28}
30   142     height={28}
31   143     className="md:w-7 md:h-7 transition-opacity duration-300 opacity-0 group-hover:opacity-100"
32   144   />
33   145   </div>
34   146   <p className=""">ADD TO CART</p>
35   147   </div>
36   148   </div>
37   149   </div>
38   150
39   151   <div className="flex items-center justify-between px-4 md:px-20 mt-20">
40   152     <h2 className="text-[24px] md:text-[28px] font-bold text-black mt-10">
41   153       FEATURED PRODUCTS
42   154     </h2>
43   155     <h3 className="font-bold underline underline-offset-4">View All</h3>
44   156   </div>
45   157   <div className="flex flex-wrap justify-center gap-6 mt-10 md:20 px-4 md:px-0">
46   158     <Array.from( length: 5 ).map((_, index) => {
47   159       <div key={index} className="w-full sm:w-1/2 md:w-auto">
48   160         <Image
49   161           src={`/img/${index + 1}.jpg`}
50   162           alt={`Featured Product ${index + 1}`}
51   163           width={270}
52   164           height={263}
53   165           className="w-full md:w-[270px] h-auto object-contain md:h-[263px]"
54   166         />
55   167         <div className="flex justify-between mt-4">
56   168           <p>Perfumes</p>
57   169           <p className="font-semibold">$99</p>
58   170         </div>
59   171       </div>
60   172     })
61   173   </div>
62   174
63   175   <Footer />
64   176   </main>
65   177   ;
66   178 }

```

DYNAMIC ROUTING FILE

DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE-“ AURALINE”

Steps taken to build and integrate components.

To build and integrate components, I first planned what each component should do and how it should look. I then created the structure using React and defined how it would handle data. I added logic for things like fetching data and filtering products, and styled the components using Tailwind CSS to make them look good and work on all screen sizes. After testing to make sure everything worked correctly, I added the components to the main app by importing them and passing the necessary data. I also optimized the components to make them faster and documented how to use them. Finally, I made sure the components could be reused in other parts of the app and kept them updated to fix any issues or add new features.

Challenges faced and solutions implemented.

While building the e-commerce website, I faced challenges like slow API data fetching, broken category filters, and poor mobile responsiveness. I fixed these by adding a loading spinner, error messages, and proper filtering logic. I used Tailwind CSS to make the site responsive and optimized performance with React.memo and useCallback. Finally, I added comments and documentation to make the code easier to understand. These solutions ensured a smooth and user-friendly website.

Best practices followed during development.

I followed best practices like using reusable components, managing state with React hooks, and making the site responsive with Tailwind CSS. I optimized performance with React.memo and useCallback, added error handling and loading states, and documented the code for clarity. These steps ensured a clean, efficient, and user-friendly e-commerce website.