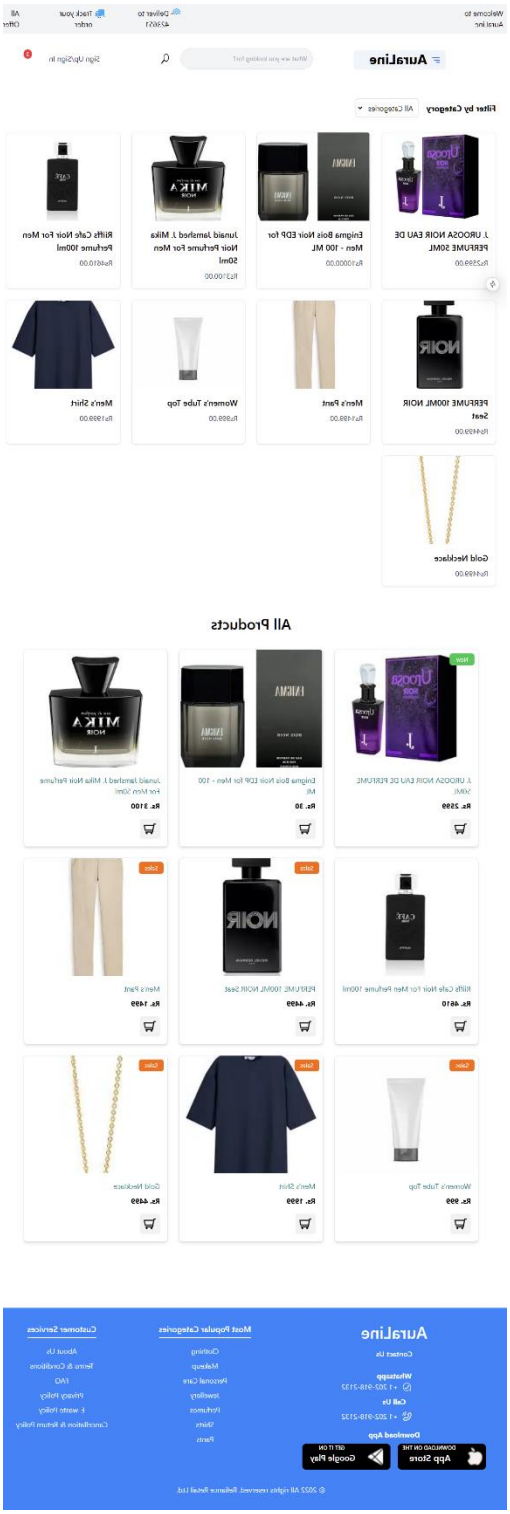
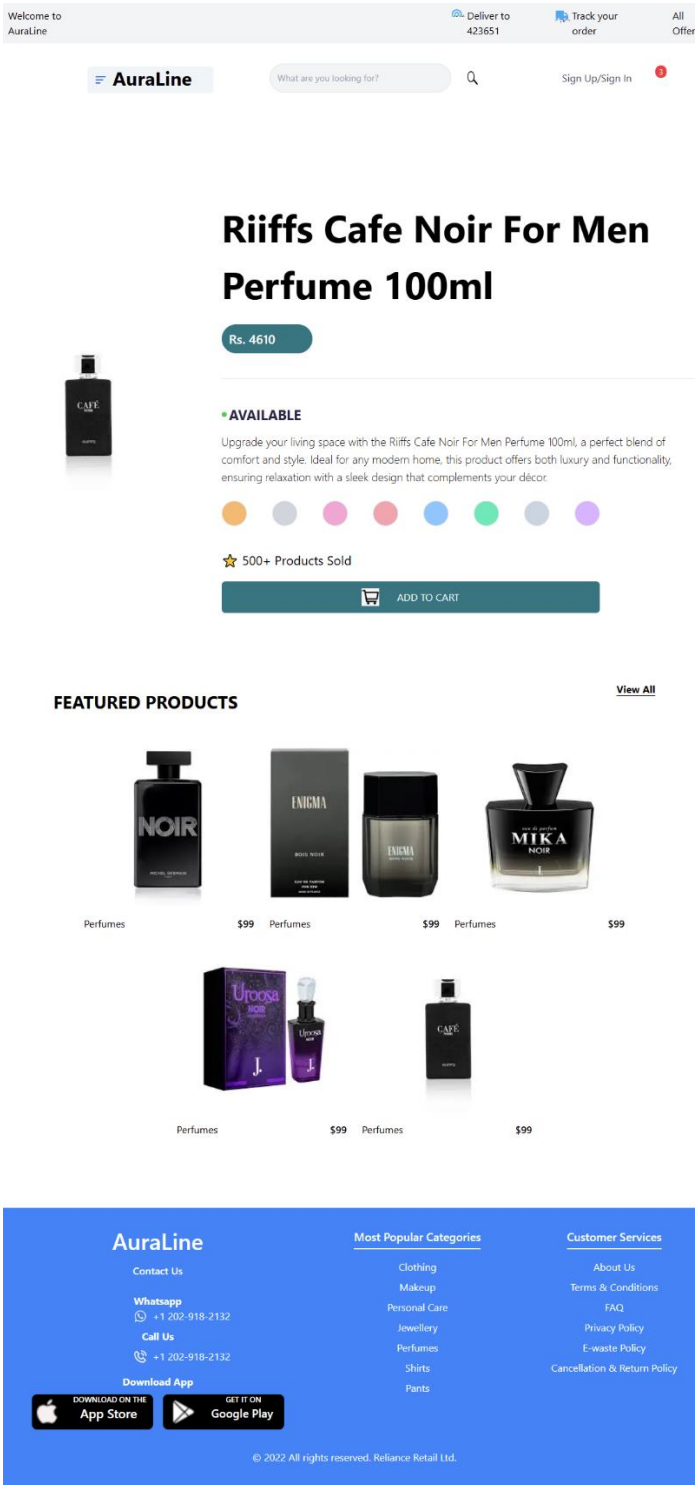
DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE-“ AURALINE”



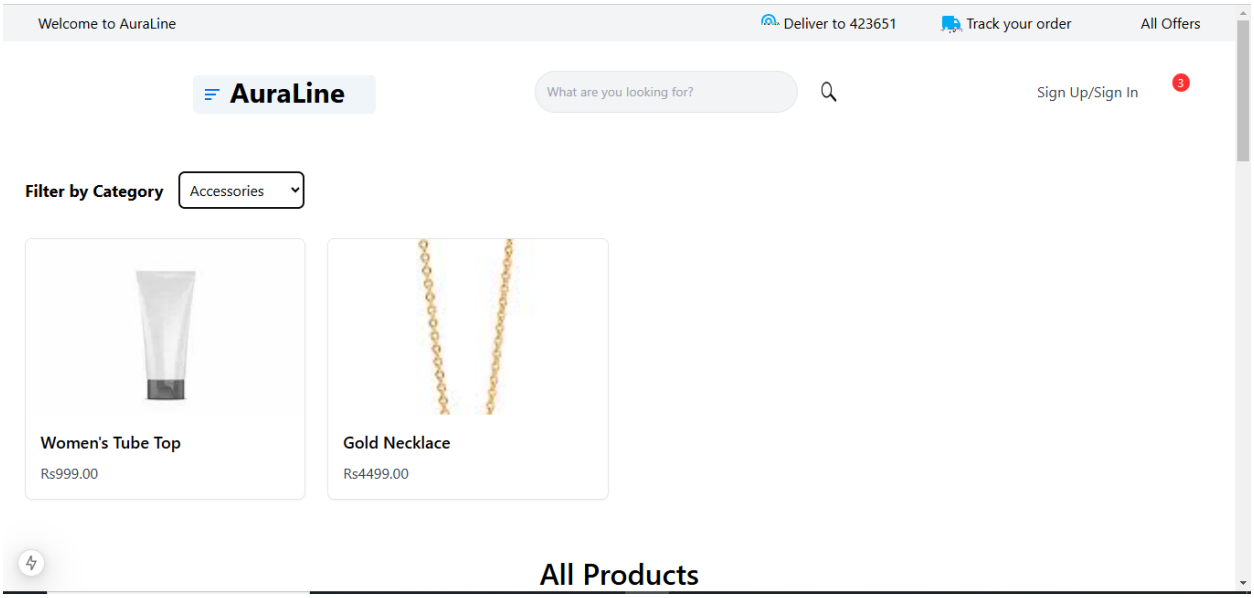
PRODUCT LISTING PAGE

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

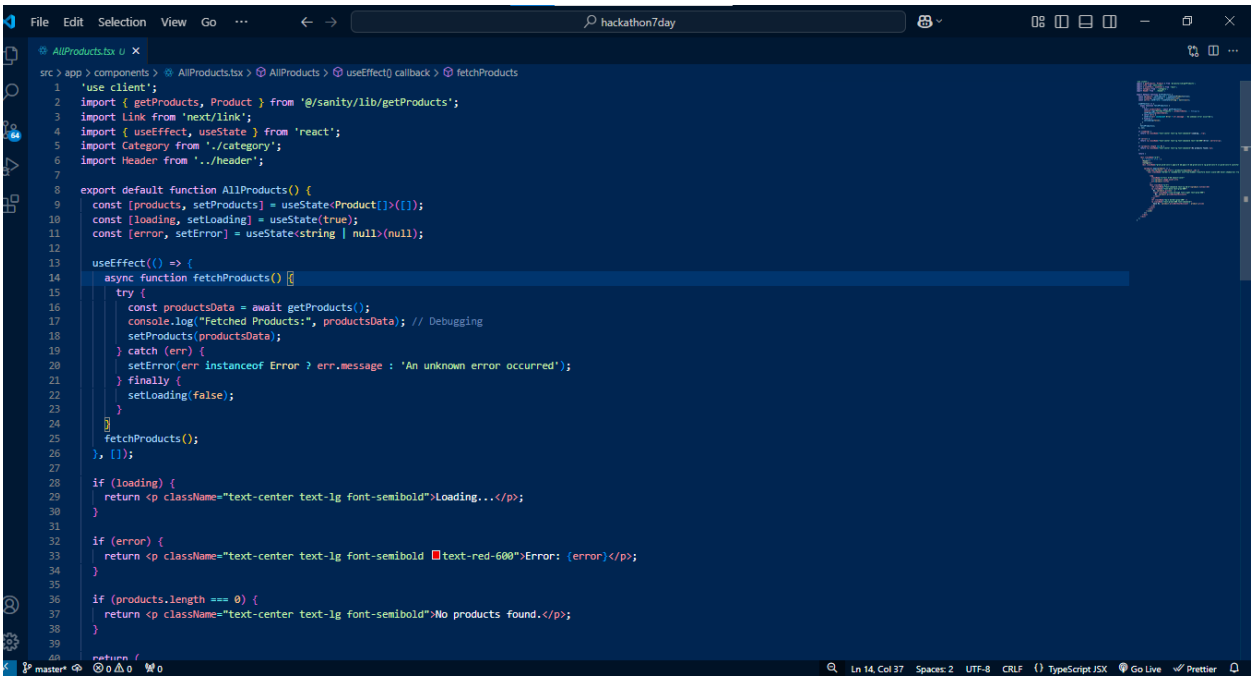


DYNAMIC PRODUCT RENDERING: http://localhost:3000/productlisting/4

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

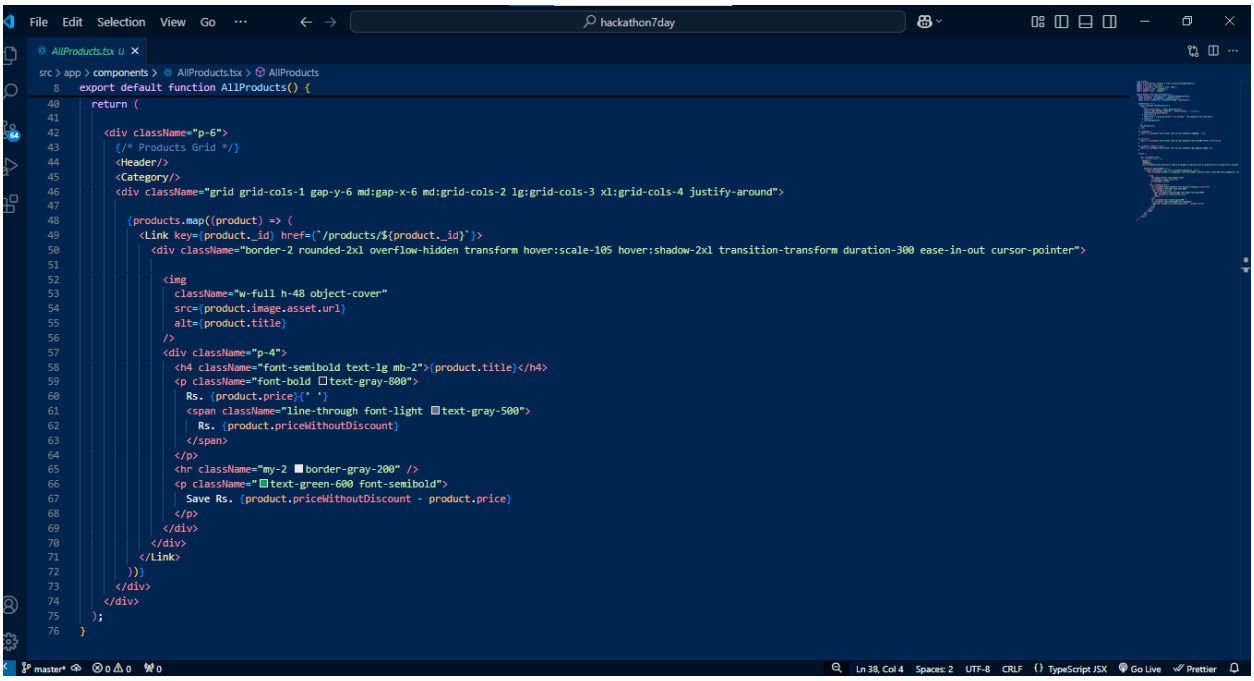


FILTERED BY CATEGORY



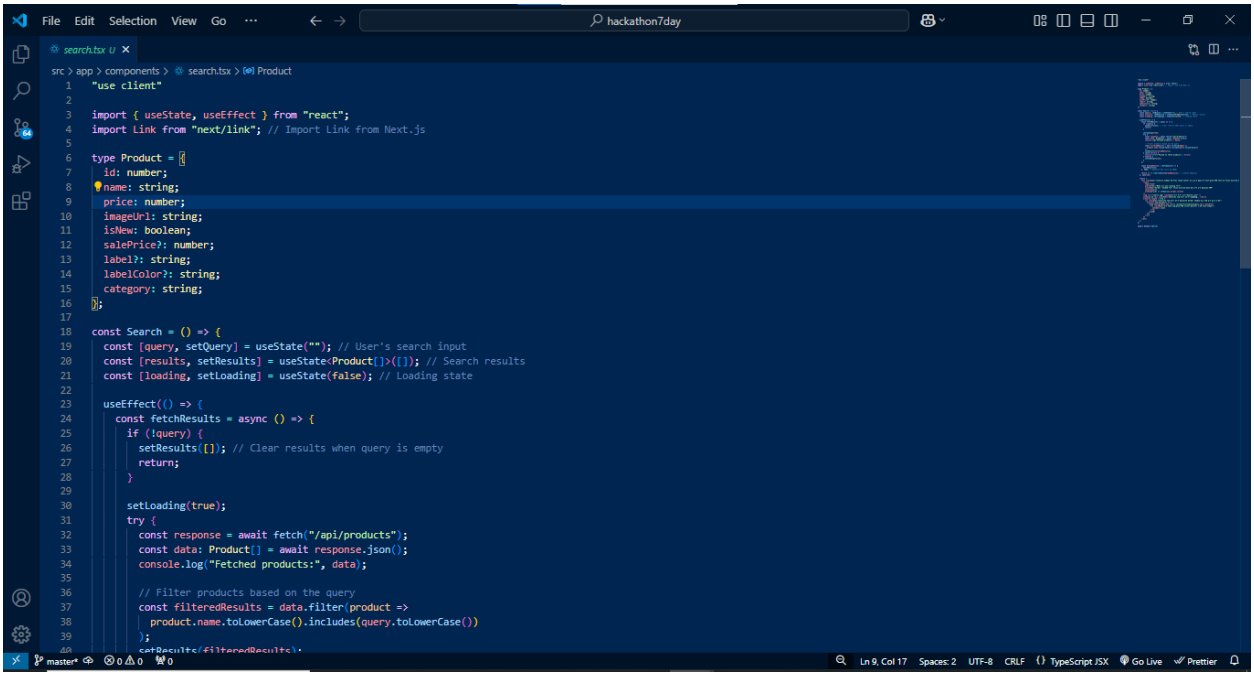
All PRODUCTS PAGE CODE

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



```tsx
export default function AllProducts() {
  return (
    <div className="p-6">
      {/* Products Grid */}
      <Header/>
      <Category/>
      <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">

        {products.map((product) => (
          <Link key={product._id} href={`/products/${product._id}`}>
            <div className="border-2 rounded-2xl overflow-hidden transform hover:scale-105 hover:shadow-2xl transition-transform duration-300 ease-in-out cursor-pointer">

              <img
                className="w-full h-48 object-cover"
                src={product.image.asset.url}
                alt={product.title}
              />
              <div className="p-4">
                <h4 className="font-semibold text-lg mb-2">{product.title}</h4>
                <p className="font-bold text-gray-800">
                  Rs. {product.price}{' '}
                  <span className="line-through font-light text-gray-500">
                    Rs. {product.priceWithoutDiscount}
                  </span>
                </p>
                <hr className="my-2 border-gray-200" />
                <p className="text-green-600 font-semibold">
                  Save Rs. {product.priceWithoutDiscount - product.price}
                </p>
              </div>
            </div>
          </Link>
        ))}
      </div>
    </div>
  );
}
```

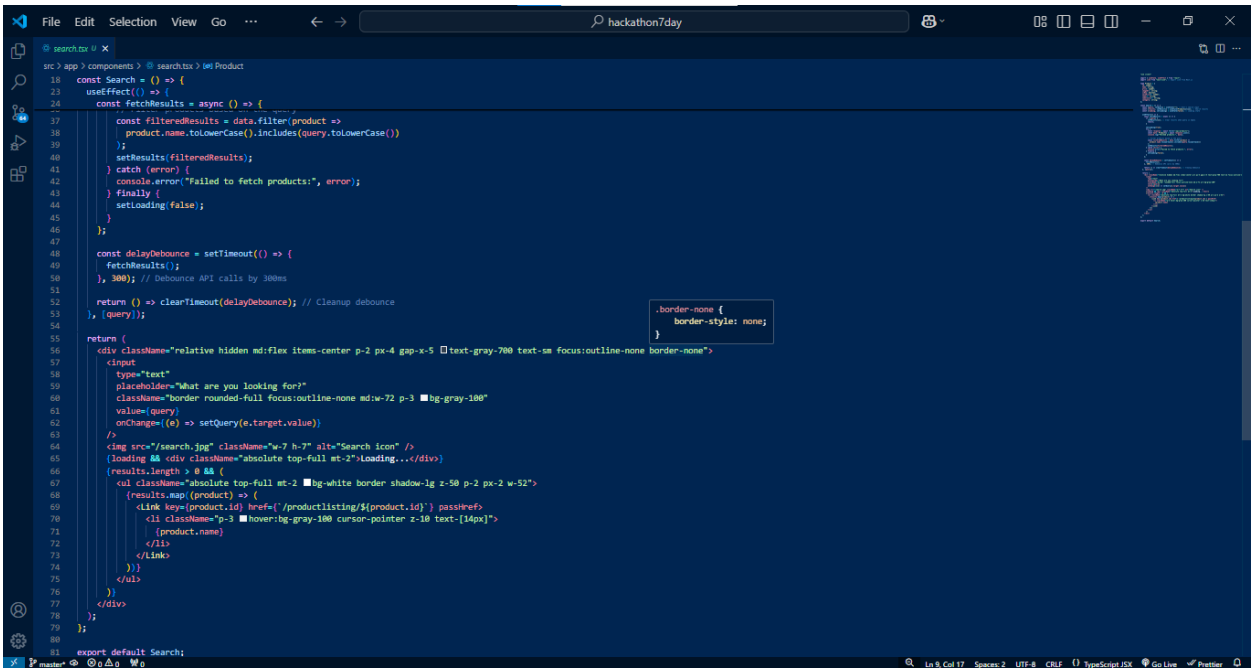
All PRODUCTS PAGE CODE



```tsx
"use client"

import { useState, useEffect } from "react";
import Link from "next/link"; // Import Link from Next.js

type Product = {
  id: number;
  name: string;
  price: number;
  imageUrl: string;
  isNew: boolean;
  salePrice?: number;
  label?: string;
  labelColor?: string;
  category: string;
};

const Search = () => {
  const [query, setQuery] = useState(""); // User's search input
  const [results, setResults] = useState<Product[]>([]); // Search results
  const [loading, setLoading] = useState(false); // Loading state

  useEffect(() => {
    const fetchResults = async () => {
      if (!query) {
        setResults([]); // Clear results when query is empty
        return;
      }

      setLoading(true);
      try {
        const response = await fetch("/api/products");
        const data: Product[] = await response.json();
        console.log("Fetched products:", data);

        // Filter products based on the query
        const filteredResults = data.filter(product =>
          product.name.toLowerCase().includes(query.toLowerCase())
        );
```

SEARCH COMPONENT CODE

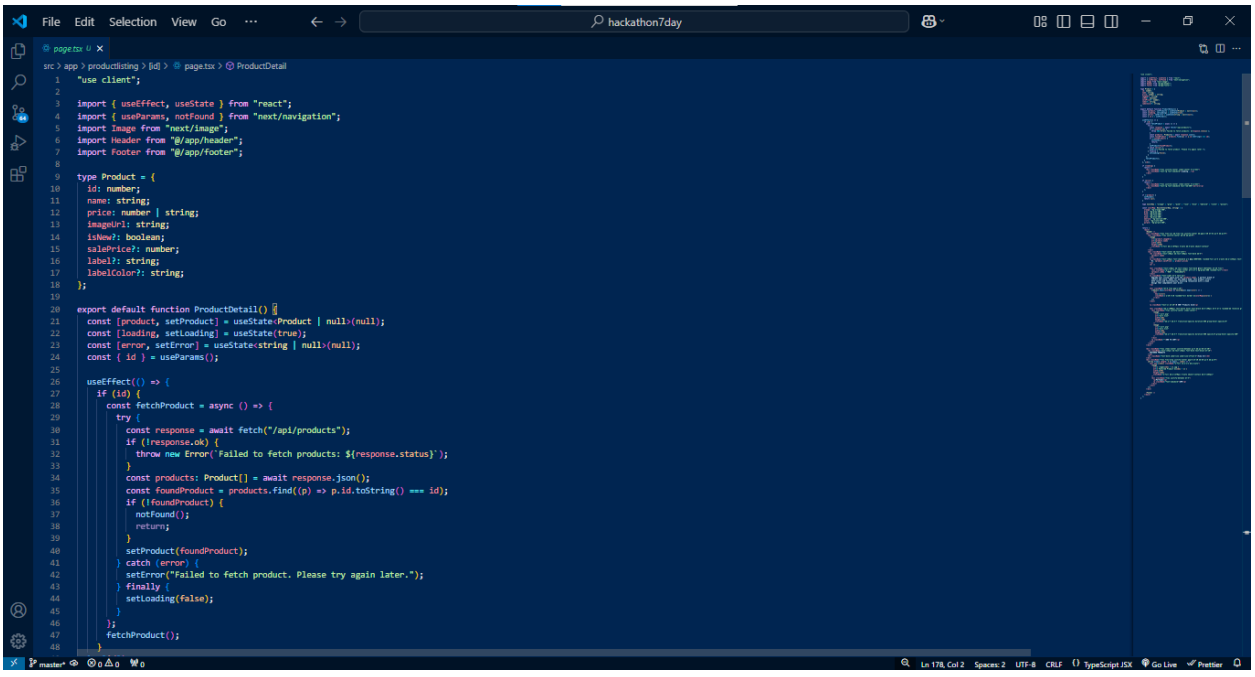
DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR YOUR MARKETPLACE-“ AURALINE”



```
const Search = () => {
  useEffect(() => {
    const fetchResults = async () => {
      const filteredResults = data.filter(product =>
        product.name.toLowerCase().includes(query.toLowerCase())
      );
      setResults(filteredResults);
    } catch (error) {
      console.error("Failed to fetch products:", error);
    } finally {
      setLoading(false);
    }
  };

  const delayDebounce = setTimeout(() => {
    fetchResults();
  }, 300); // Debounce API calls by 300ms

  return () => clearTimeout(delayDebounce); // Cleanup debounce
}, [query]);

return (
  <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">
    <input
      type="text"
      placeholder="What are you looking for?"
      className="border rounded-full focus:outline-none md:w-72 p-3 bg-gray-100"
      value={query}
      onChange={(e) => setQuery(e.target.value)}
    />
    <img src="/search.jpg" className="w-7 h-7" alt="Search icon" />
    {loading && <div className="absolute top-full mt-2">Loading...</div>}
    {results.length > 0 && (
      <ul className="absolute top-full mt-2 bg-white border shadow-lg z-50 p-2 px-2 w-52">
        {results.map((product) => (
          <Link key={product.id} href={`/productlisting/${product.id}`} passHref>
            <li className="p-3 hover:bg-gray-100 cursor-pointer z-10 text-[14px]">
              {product.name}
            </li>
          </Link>
        ))}
      </ul>
    )}
  </div>
);
};

export default Search;
```

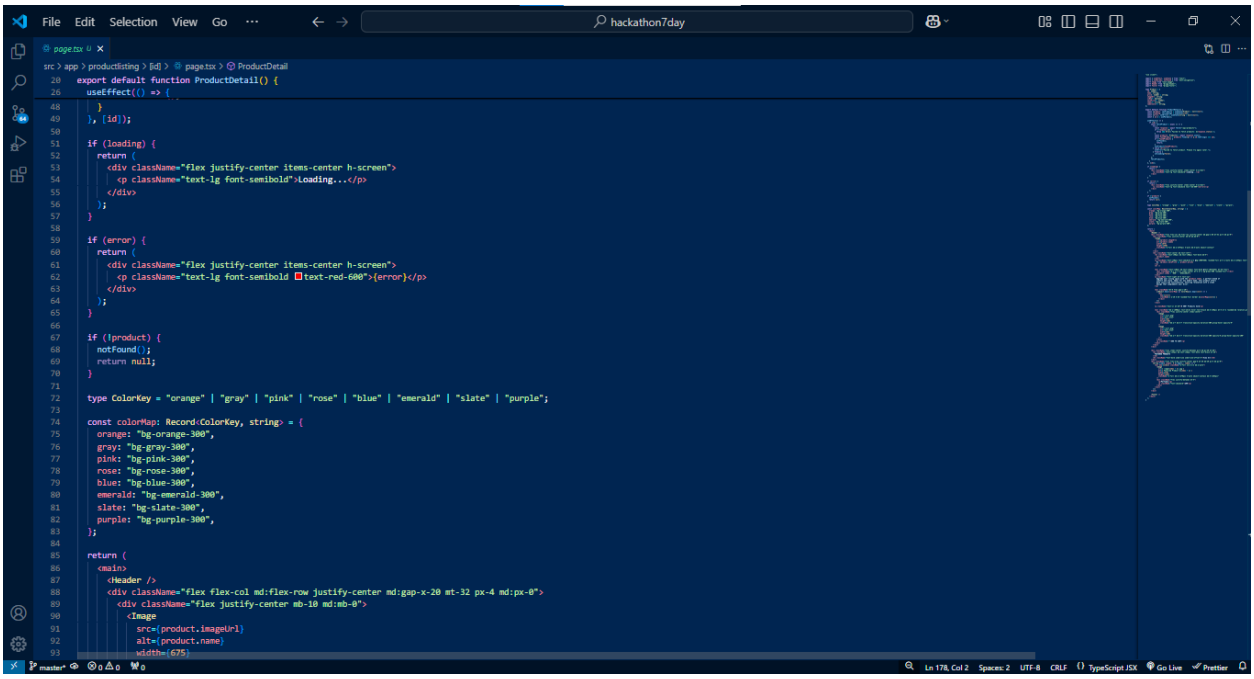
```
.border-none {
    border-style: none;
}
```

SEARCH COMPONENT CODE

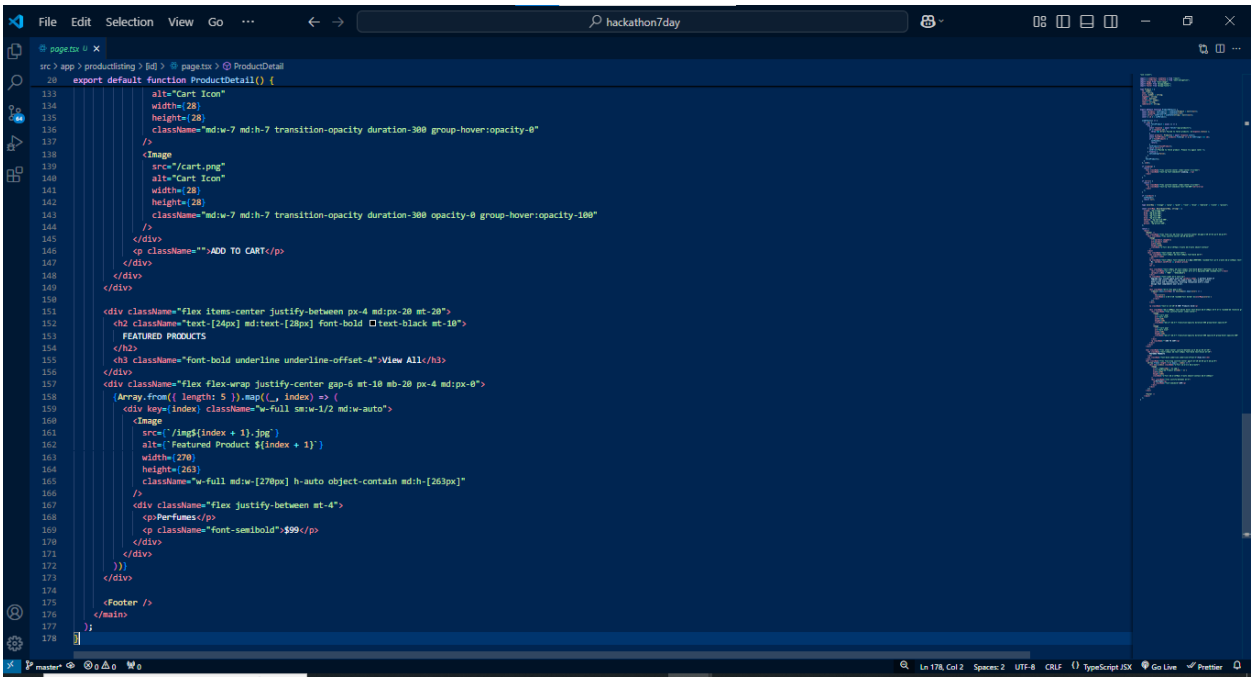


```
"use client";

import { useEffect, useState } from "react";
import { useParams, notFound } from "next/navigation";
import Image from "next/image";
import Header from "@/app/header";
import Footer from "@/app/footer";

type Product = {
  id: number;
  name: string;
  price: number | string;
  imageUrl: string;
  isNew?: boolean;
  salePrice?: number;
  label?: string;
  labelColor?: string;
};

export default function ProductDetail() {
  const [product, setProduct] = useState<Product | null>(null);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState<string | null>(null);
  const { id } = useParams();

  useEffect(() => {
    if (id) {
      const fetchProduct = async () => {
        try {
          const response = await fetch("/api/products");
          if (!response.ok) {
            throw new Error(`Failed to fetch products: ${response.status}`);
          }
          const products: Product[] = await response.json();
          const foundProduct = products.find((p) => p.id.toString() === id);
          if (!foundProduct) {
            notFound();
            return;
          }
          setProduct(foundProduct);
        } catch (error) {
          setError("Failed to fetch product. Please try again later.");
        } finally {
          setLoading(false);
        }
      };
      fetchProduct();
    }
```

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



```
export default function ProductDetail() {
  useEffect(() => {
  }, [id]);

  if (loading) {
    return (
      <div className="flex justify-center items-center h-screen">
        <p className="text-lg font-semibold">Loading...</p>
      </div>
    );
  }

  if (error) {
    return (
      <div className="flex justify-center items-center h-screen">
        <p className="text-lg font-semibold text-red-600">{error}</p>
      </div>
    );
  }

  if (!product) {
    notFound();
    return null;
  }

  type ColorKey = "orange" | "gray" | "pink" | "rose" | "blue" | "emerald" | "slate" | "purple";

  const colorMap: Record<ColorKey, string> = {
    orange: "bg-orange-300",
    gray: "bg-gray-300",
    pink: "bg-pink-300",
    rose: "bg-rose-300",
    blue: "bg-blue-300",
    emerald: "bg-emerald-300",
    slate: "bg-slate-300",
    purple: "bg-purple-300",
  };

  return (
    <main>
      <Header />
      <div className="flex flex-col md:flex-row justify-center md:gap-x-20 mt-32 px-4 md:px-0">
        <div className="flex justify-center mb-10 md:mb-0">
          <Image
            src={product.imageUrl}
            alt={product.name}
            width={675}
```



```
export default function ProductDetail() {
                alt="Cart Icon"
                width={28}
                height={28}
                className="md:w-7 md:h-7 transition-opacity duration-300 group-hover:opacity-0"
              />
              <Image
                src="/cart.png"
                alt="Cart Icon"
                width={28}
                height={28}
                className="md:w-7 md:h-7 transition-opacity duration-300 opacity-0 group-hover:opacity-100"
              />
            </div>
            <p >ADD TO CART</p>
          </div>
        </div>
      </div>

      <div >
        <h2 >
          FEATURED PRODUCTS
        </h2>
        <h3 >View All</h3>
      </div>
      <div >
        {Array.from({ length: 5 }).map((_, index) => (
          <div key={index} >
            <Image
              src={`/img${index + 1}.jpg`}
              alt={`Featured Product ${index + 1}`}
              width={270}
              height={263}
              />
            <div >
              <p>Perfumes</p>
              <p >$99</p>
            </div>
          </div>
        ))}
      </div>

      <Footer />
    </main>
  );
}
```

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.