"Day 4 - Dynamic Frontend Components - Nike Store"

Introduction:

This document presents the learnings and implementation of dynamic frontend components for an e-commerce clothing marketplace as part of the Day 4 task. It focuses on designing modular and reusable components to display data fetched from Sanity CMS or APIs. The document outlines key components such as product listing, detail pages, cart, and search functionality.

Core Components and functionalities :

1. Product Listing page.
2. Product details dynamic page.
3. Cart functionality.
4. Checkout flow.
5. Header and Footer component.
6. Wishlist.
7. Category functionality.
8. Search bar functionality.

   Each functionality contributes to a responsive and scalable market place.

1. Product Listing Page :

Product listing page is the primary interface where users can find all the available products and can filter according to category price of color. Products are fetched from Sanity CMS, a responsive design for all devices.

2. Product Details Page :

Product details page dynamically shows details about each procust like image, description, color, stock, category and it aso offers wishlist and add to cart options to increase users engagement.

3. Cart Functionality :

Cart functionality manages the user's selected products to provide seamless user experience. Users can add products from the product listing page and also from the product detail page. Carts dynamically handle quantity and summarize total cost to ensure real-time shopping experience. Local storage used to ensure cart items are intact while the page is refreshed.

4. **Checkout Flow :**

When users click to checkout it takes them to checkout page where all the cart details come dynamically and total amount calculated, this page contain a form about details of customer, after submitting customer click to submit and two actions performed all the order details go to sanity and record there and later shipment functionality works for shipment process.

5. Header/Footer Components :

Header and footer are reusable components, in header it contains all the page links like “all products” , “Women” , “Men” , “Kid” etc it also contains other pages links like “Wishlist” and “Cart” etc. In footer it has quick links that help users to find their required things.

6.. **Wishlist :**

Users can add items to their cart and can also save their favorite products to their wishlist page by just clicking to the “heart” icon.

7. **Categories Functionality :**

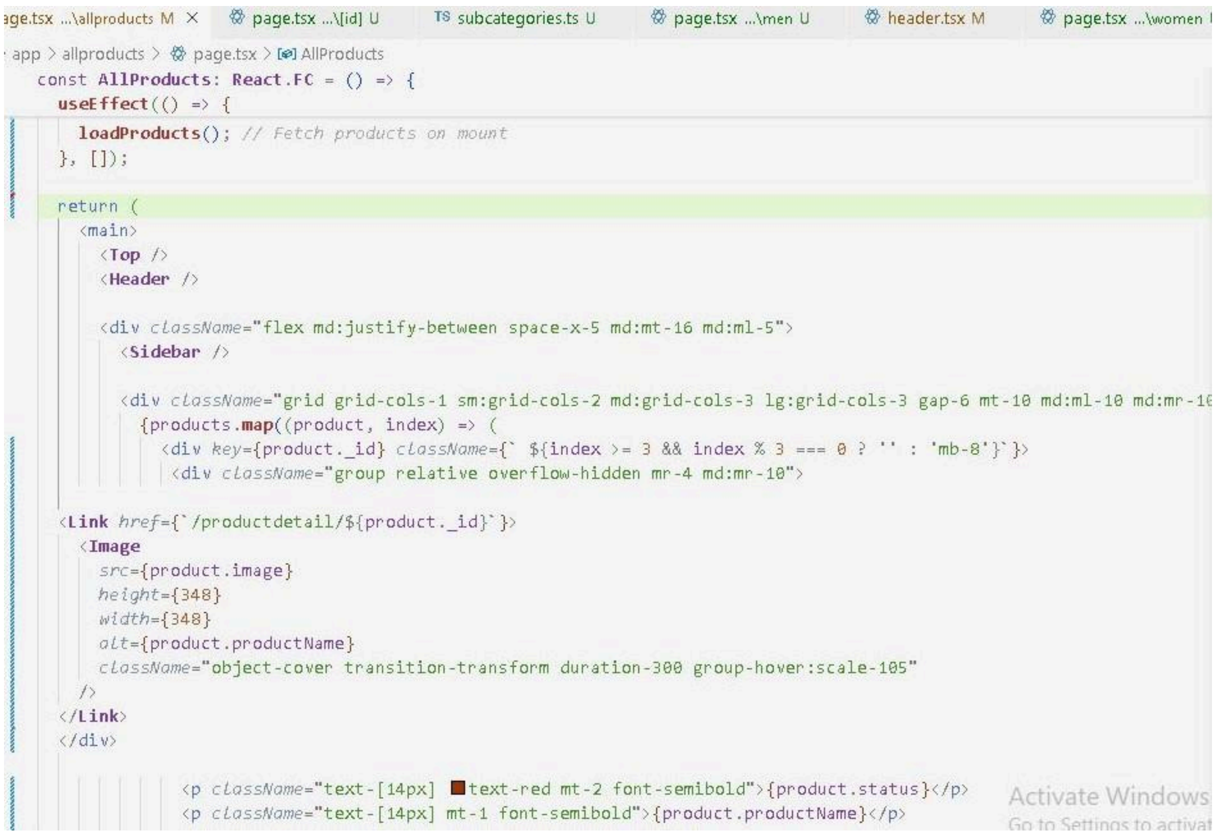
Users can add items to their cart and can also save their favorite products to their wishlist page by just clicking to the “heart” icon.

8. **Search Bar Functionality :**

Users can search any product with its name from the search bar in the header , it helps them to find their desired product quickly.

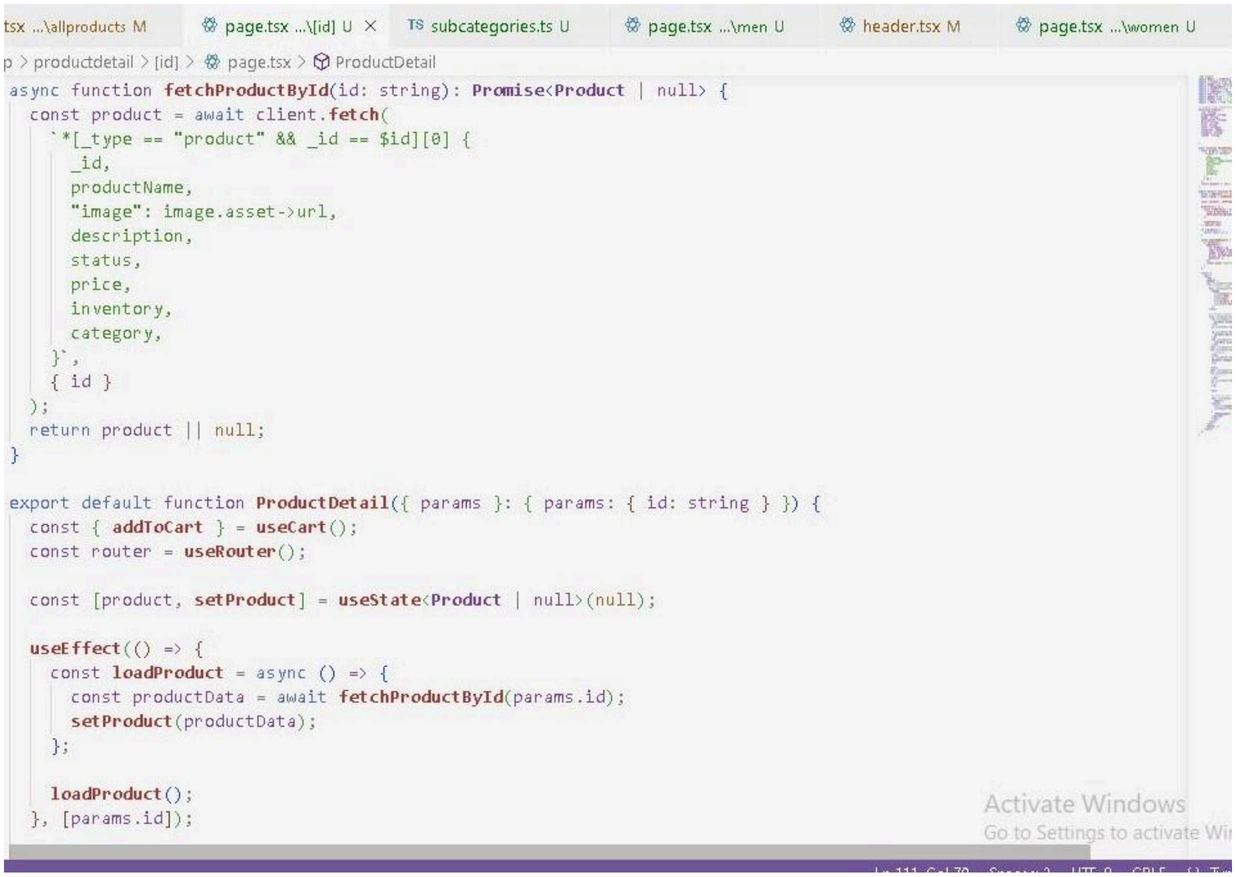
Code Deliverables :

- **Product listing page with Dynamic Data :**



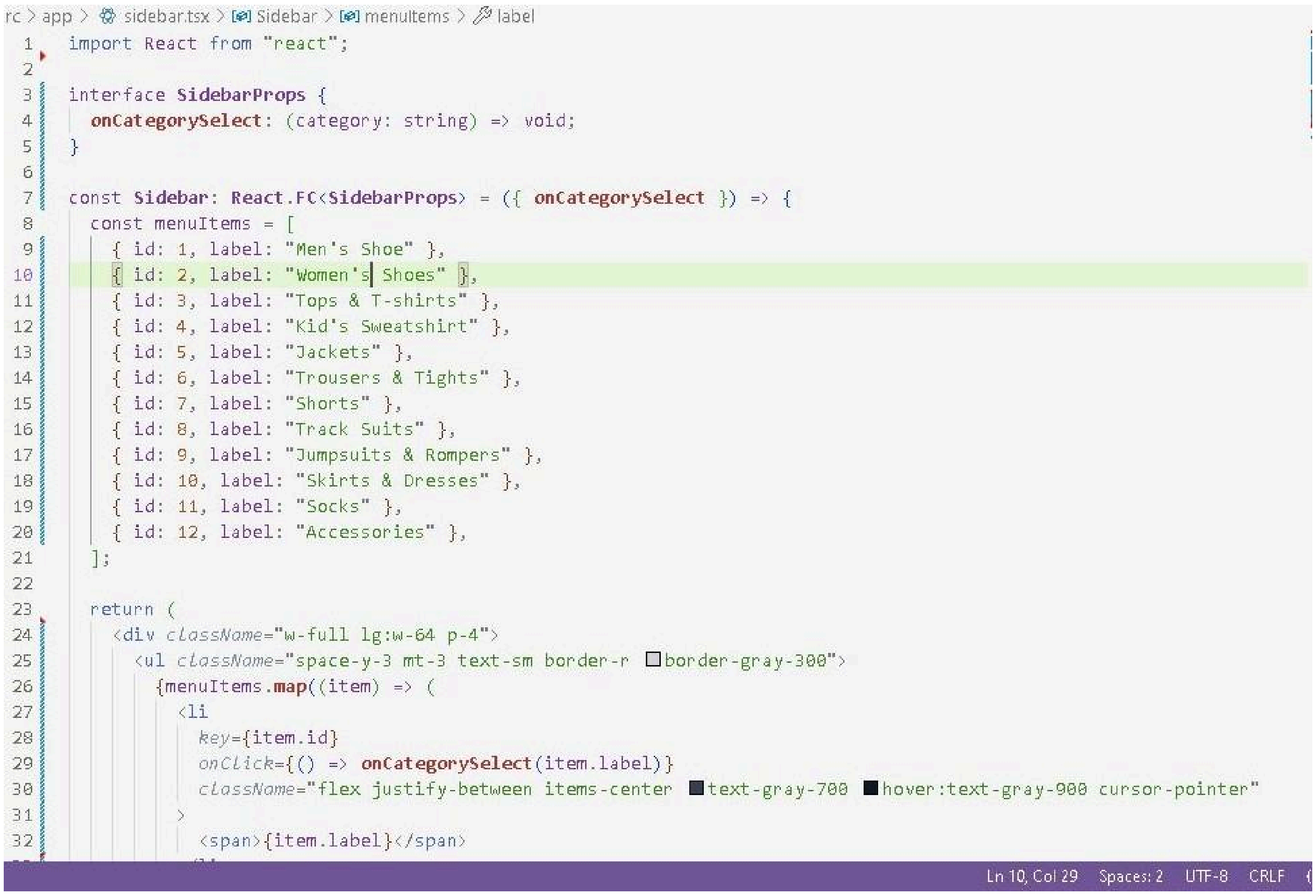
```
const AllProducts: React.FC = () => {
  useEffect(() => {
    loadProducts(); // Fetch products on mount
  }, []);

  return (
    <main>
      <Top />
      <Header />

      <div className="flex md:justify-between space-x-5 md:mt-16 md:ml-5">
        <Sidebar />

        <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-3 gap-6 mt-10 md:ml-10 md:mr-1
          {products.map((product, index) => (
            <div key={product._id} className={` ${index >= 3 && index % 3 === 0 ? '' : 'mb-8'}`}>
             <div className="group relative overflow-hidden mr-4 md:mr-10">

  <Link href={`/productdetail/${product._id}`}>
    <Image
      src={product.image}
      height={348}
      width={348}
      alt={product.productName}
      className="object-cover transition-transform duration-300 group-hover:scale-105"
    />
  </Link>
  </div>

              <p className="text-[14px] text-red mt-2 font-semibold">{product.status}</p>
              <p className="text-[14px] mt-1 font-semibold">{product.productName}</p>
```

- **Product detail page :**



```
async function fetchProductById(id: string): Promise<Product | null> {
  const product = await client.fetch(
    `*[_type == "product" && _id == $id][0] {
      _id,
      productName,
      "image": image.asset->url,
      description,
      status,
      price,
      inventory,
      category,
    }`,
    { id }
  );
  return product || null;
}

export default function ProductDetail({ params }: { params: { id: string } }) {
  const { addToCart } = useCart();
  const router = useRouter();

  const [product, setProduct] = useState<Product | null>(null);

  useEffect(() => {
    const loadProduct = async () => {
      const productData = await fetchProductById(params.id);
      setProduct(productData);
    };

    loadProduct();
  }, [params.id]);
```

- **Categories Functionality :**

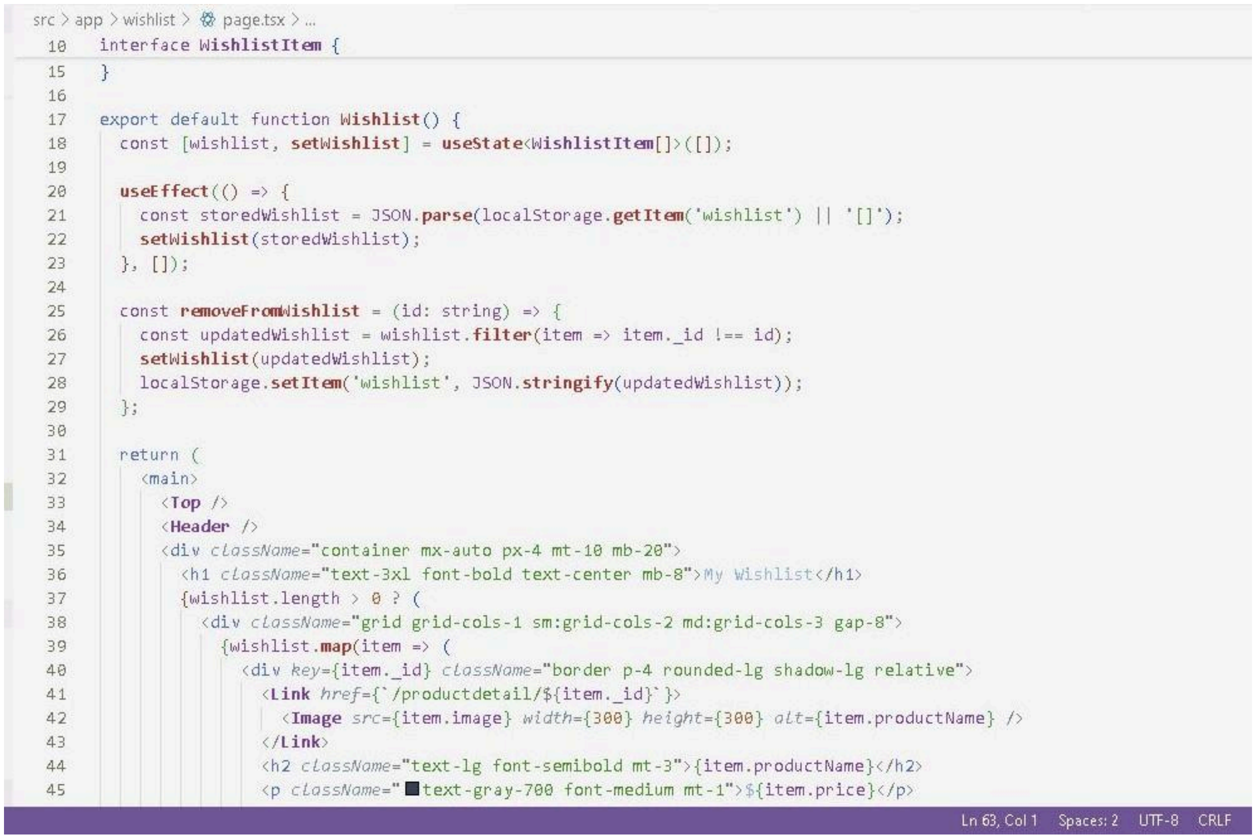


rc > app > sidebar.tsx > Sidebar > menuItems > label

```
import React from "react";

interface SidebarProps {
  onCategorySelect: (category: string) => void;
}

const Sidebar: React.FC<SidebarProps> = ({ onCategorySelect }) => {
  const menuItems = [
    { id: 1, label: "Men's Shoe" },
    { id: 2, label: "Women's Shoes" },
    { id: 3, label: "Tops & T-shirts" },
    { id: 4, label: "Kid's Sweatshirt" },
    { id: 5, label: "Jackets" },
    { id: 6, label: "Trousers & Tights" },
    { id: 7, label: "Shorts" },
    { id: 8, label: "Track Suits" },
    { id: 9, label: "Jumpsuits & Rompers" },
    { id: 10, label: "Skirts & Dresses" },
    { id: 11, label: "Socks" },
    { id: 12, label: "Accessories" },
  ];

  return (
    <div className="w-full lg:w-64 p-4">
      <ul className="space-y-3 mt-3 text-sm border-r border-gray-300">
        {menuItems.map((item) => (
          <li
            key={item.id}
            onClick={() => onCategorySelect(item.label)}
            className="flex justify-between items-center text-gray-700 hover:text-gray-900 cursor-pointer"
          >
            <span>{item.label}</span>
```

Ln 10, Col 29   Spaces: 2   UTF-8   CRLF

- **Wishlist Functionality :**

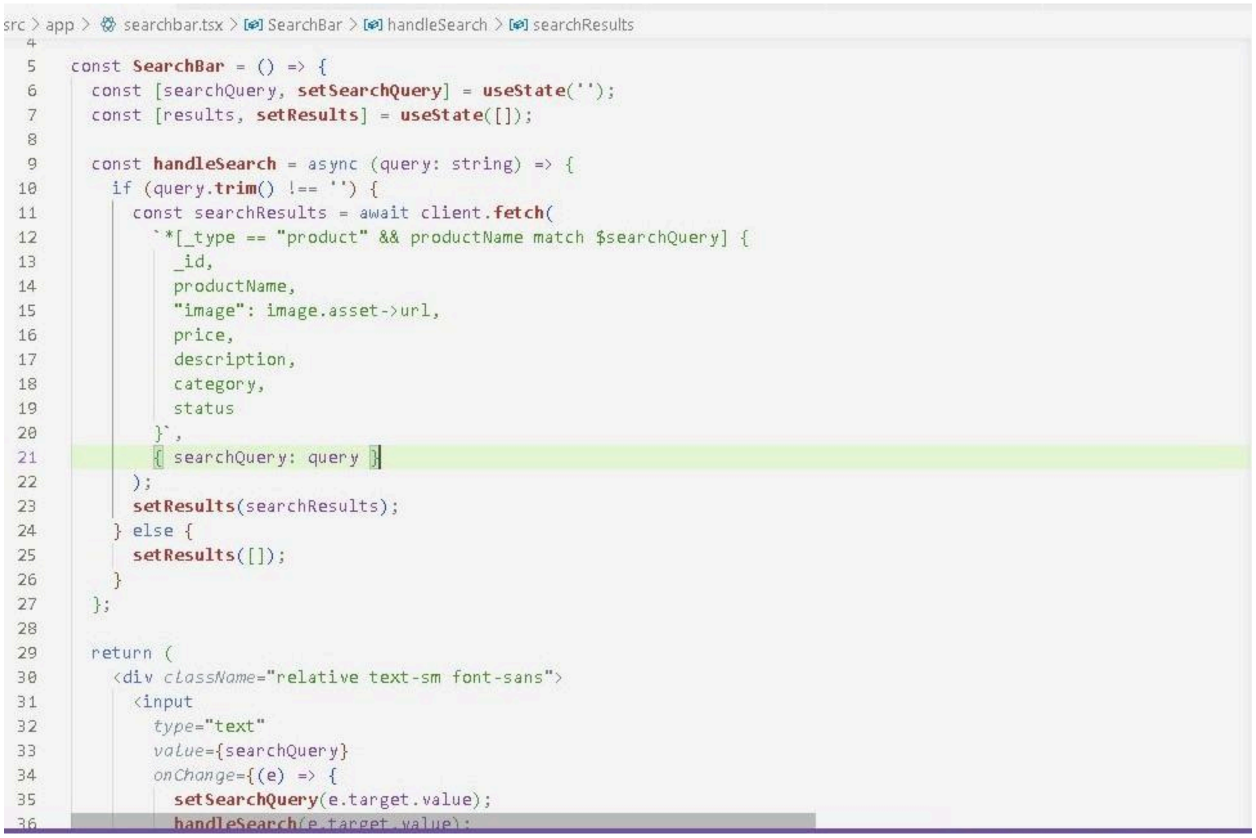


src > app > wishlist > page.tsx > ...

```tsx
interface WishlistItem {
}

export default function Wishlist() {
  const [wishlist, setWishlist] = useState<WishlistItem[]>([]);

  useEffect(() => {
    const storedWishlist = JSON.parse(localStorage.getItem('wishlist') || '[]');
    setWishlist(storedWishlist);
  }, []);

  const removeFromWishlist = (id: string) => {
    const updatedWishlist = wishlist.filter(item => item._id !== id);
    setWishlist(updatedWishlist);
    localStorage.setItem('wishlist', JSON.stringify(updatedWishlist));
  };

  return (
    <main>
      <Top />
      <Header />
      <div className="container mx-auto px-4 mt-10 mb-20">
        <h1 className="text-3xl font-bold text-center mb-8">My Wishlist</h1>
        {wishlist.length > 0 ? (
          <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 gap-8">
            {wishlist.map(item => (
              <div key={item._id} className="border p-4 rounded-lg shadow-lg relative">
                <Link href={`/productdetail/${item._id}`}>
                  <Image src={item.image} width={300} height={300} alt={item.productName} />
                </Link>
                <h2 className="text-lg font-semibold mt-3">{item.productName}</h2>
                <p className="text-gray-700 font-medium mt-1">${item.price}</p>
```

Ln 63, Col 1   Spaces: 2   UTF-8   CRLF

- **SearchBar Functionality :**



```
src > app > searchbar.tsx > SearchBar > handleSearch > searchResults

const SearchBar = () => {
  const [searchQuery, setSearchQuery] = useState('');
  const [results, setResults] = useState([]);

  const handleSearch = async (query: string) => {
    if (query.trim() !== '') {
      const searchResults = await client.fetch(
        `*[_type == "product" && productName match $searchQuery] {
          _id,
          productName,
          "image": image.asset->url,
          price,
          description,
          category,
          status
        }`,
        { searchQuery: query }
      );
      setResults(searchResults);
    } else {
      setResults([]);
    }
  };

  return (
    <div className="relative text-sm font-sans">
      <input
        type="text"
        value={searchQuery}
        onChange={(e) => {
          setSearchQuery(e.target.value);
          handleSearch(e.target.value);
```

Challenges faced and solutions implemented :

1. **Challenge**: Managing dynamic product data fetching in Next.js.
    - **Solution**: Used `useEffect` with `fetch` API to load product data asynchronously on component mount.

2. **Challenge**: Handling dynamic category filtering with the sidebar.
    - **Solution**: Implemented category filtering using state and dynamically updating product lists based on selected categories.

3. **Challenge**: Managing local storage for wishlist functionality.
    - **Solution**: Used `localStorage` API to store and retrieve product data for the wishlist on page load and interactions.

4. **Challenge**: Ensuring the search results appear smoothly without disrupting the page design.
    - **Solution**: Applied Tailwind CSS for creating an overlay search results modal with proper z-index and styling.

Best Practices Followed :

- **Clear Project Structure**: I organized my code with proper components and folders for easy maintenance and readability.

- **Version Control**: I used Git for version control to track changes and collaborate efficiently on my projects.

- **Responsive Design**: I ensured my websites are mobile-friendly by using Tailwind CSS's responsive utilities.

- **Reusable Code**: I created reusable components and functions to avoid repetition and improve code efficiency.