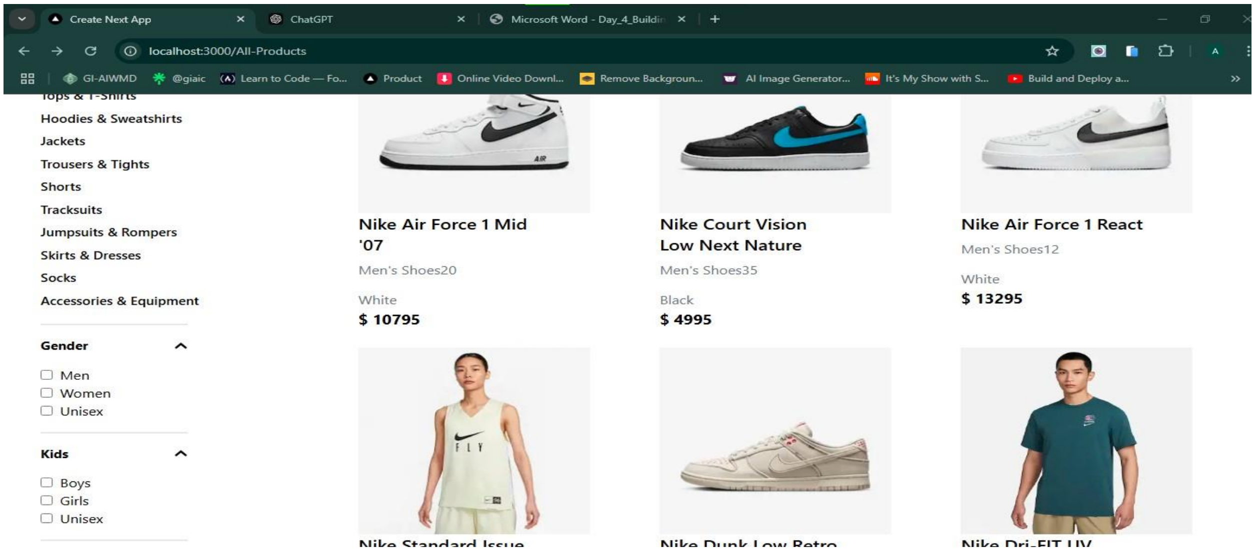
Day 4 - Dynamic Frontend Components – Every Thing

Functional Deliverables:

Dynamic Listing Product:



Tops & T-Shirts
Hoodies & Sweatshirts
Jackets
Trousers & Tights
Shorts
Tracksuits
Jumpsuits & Rompers
Skirts & Dresses
Socks
Accessories & Equipment

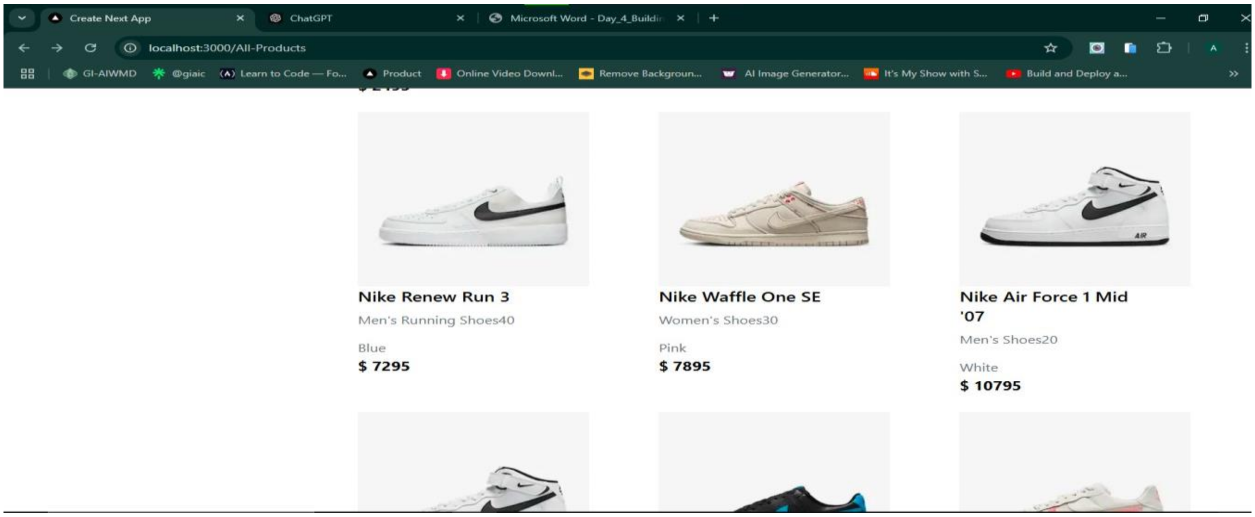
Gender

□ Men
□ Women
□ Unisex

Kids

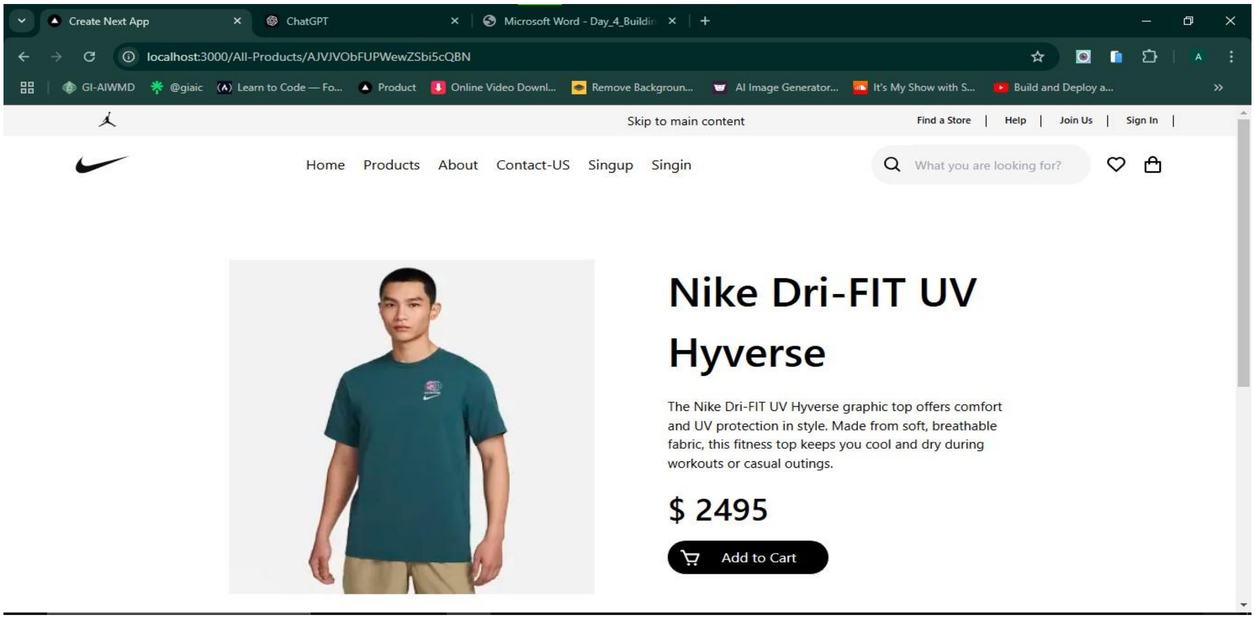
□ Boys
□ Girls
□ Unisex

Nike Air Force 1 Mid '07 Men's Shoes20 White $ 10795	Nike Court Vision Low Next Nature Men's Shoes35 Black $ 4995	Nike Air Force 1 React Men's Shoes12 White $ 13295
Nike Standard Issue	Nike Dunk Low Retro	Nike Dri-FIT UV



Nike Renew Run 3 Men's Running Shoes40 Blue $ 7295	Nike Waffle One SE Women's Shoes30 Pink $ 7895	Nike Air Force 1 Mid '07 Men's Shoes20 White $ 10795

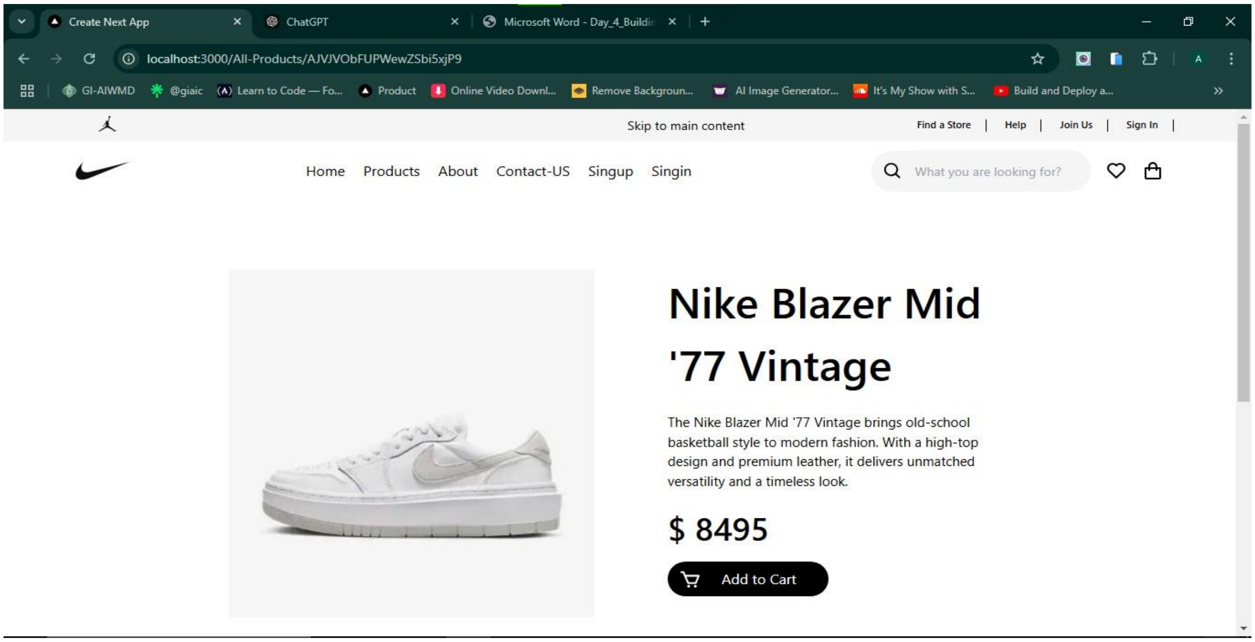
Product detail pages with accurate routing and data rendering:



Nike Dri-FIT UV Hyverse

The Nike Dri-FIT UV Hyverse graphic top offers comfort and UV protection in style. Made from soft, breathable fabric, this fitness top keeps you cool and dry during workouts or casual outings.

$ 2495



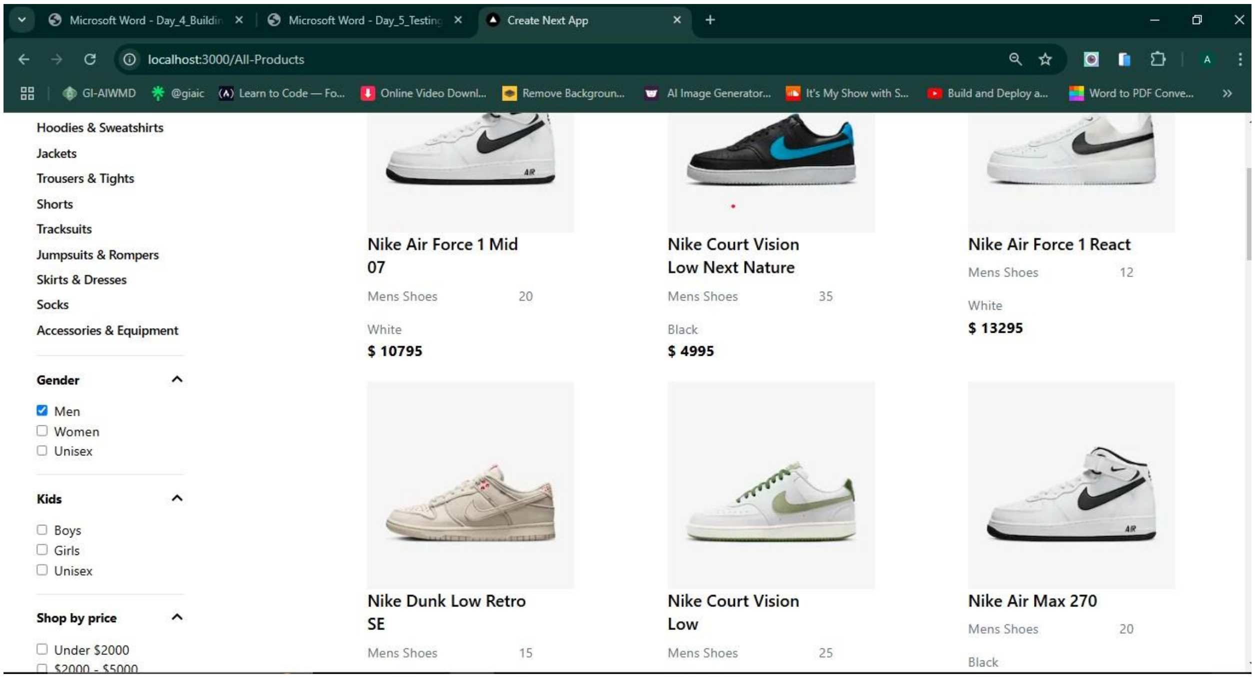
Nike Blazer Mid '77 Vintage

The Nike Blazer Mid '77 Vintage brings old-school basketball style to modern fashion. With a high-top design and premium leather, it delivers unmatched versatility and a timeless look.

$ 8495

Category & Price filters :

Category Filter ➔



Hoodies & Sweatshirts
Jackets
Trousers & Tights
Shorts
Tracksuits
Jumpsuits & Rompers
Skirts & Dresses
Socks
Accessories & Equipment

Gender

☑ Men
☐ Women
☐ Unisex

Kids

☐ Boys
☐ Girls
☐ Unisex

Shop by price

☐ Under $2000
☐ $2000 - $5000

Nike Air Force 1 Mid 07
Mens Shoes 20
White
$ 10795

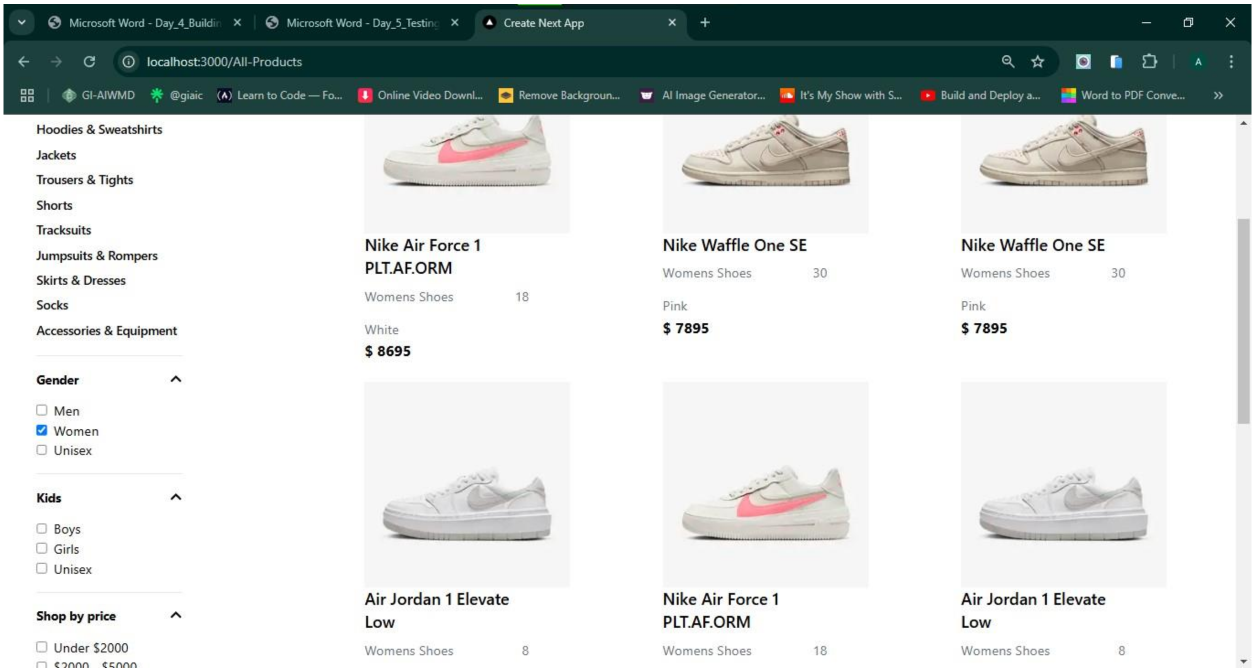
Nike Court Vision Low Next Nature
Mens Shoes 35
Black
$ 4995

Nike Air Force 1 React
Mens Shoes 12
White
$ 13295

Nike Dunk Low Retro SE
Mens Shoes 15

Nike Court Vision Low
Mens Shoes 25

Nike Air Max 270
Mens Shoes 20
Black



Hoodies & Sweatshirts
Jackets
Trousers & Tights
Shorts
Tracksuits
Jumpsuits & Rompers
Skirts & Dresses
Socks
Accessories & Equipment

Gender

☐ Men
☑ Women
☐ Unisex

Kids

☐ Boys
☐ Girls
☐ Unisex

Shop by price

☐ Under $2000
☐ $2000 - $5000

Nike Air Force 1 PLT.AF.ORM
Womens Shoes 18
White
$ 8695

Nike Waffle One SE
Womens Shoes 30
Pink
$ 7895

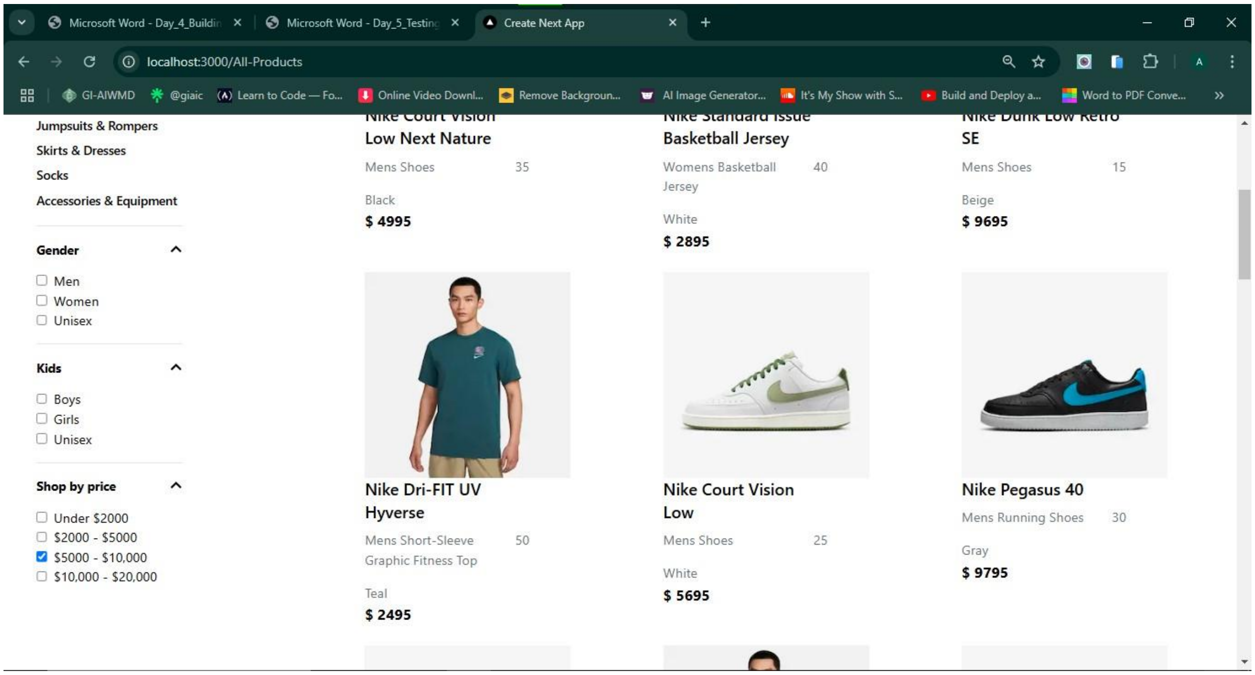
Nike Waffle One SE
Womens Shoes 30
Pink
$ 7895

Air Jordan 1 Elevate Low
Womens Shoes 8

Nike Air Force 1 PLT.AF.ORM
Womens Shoes 18

Air Jordan 1 Elevate Low
Womens Shoes 8

Price Filter ➔



Jumpsuits & Rompers
Skirts & Dresses
Socks
Accessories & Equipment

Gender
□ Men
□ Women
□ Unisex

Kids
□ Boys
□ Girls
□ Unisex

Shop by price
□ Under $2000
□ $2000 - $5000
☑ $5000 - $10,000
□ $10,000 - $20,000

Nike Court Vision Low Next Nature
Mens Shoes 35
Black
$ 4995

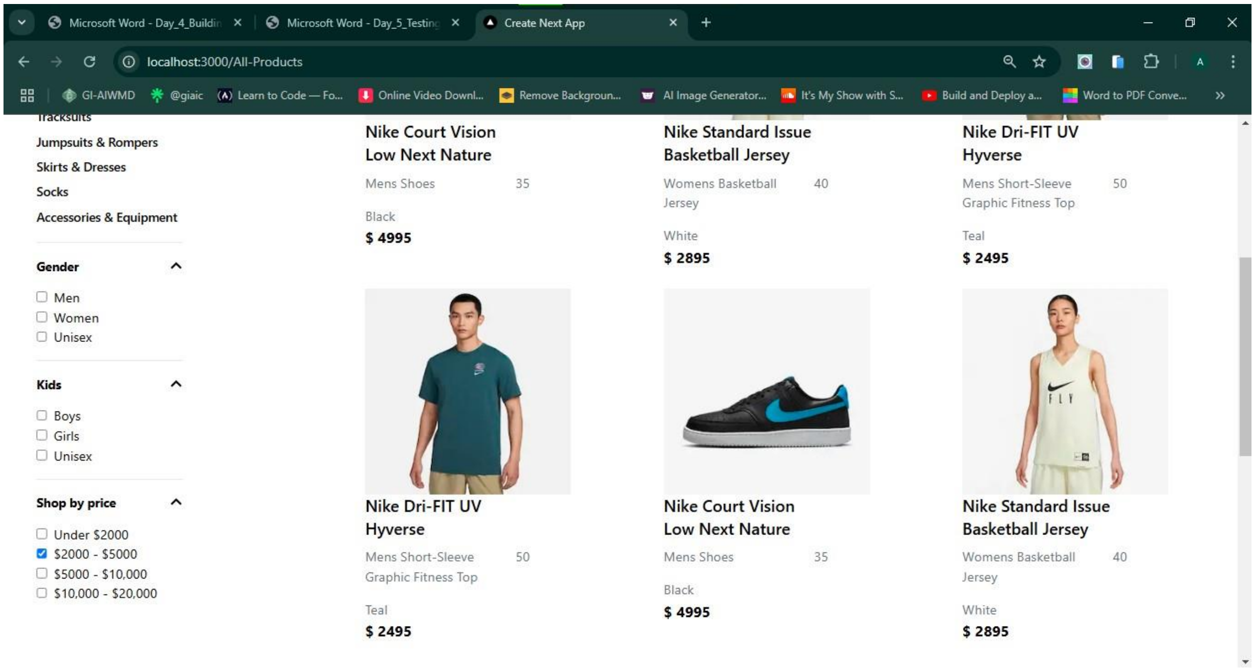
Nike Standard Issue Basketball Jersey
Womens Basketball Jersey 40
White
$ 2895

Nike Dunk Low Retro SE
Mens Shoes 15
Beige
$ 9695

Nike Dri-FIT UV Hyverse
Mens Short-Sleeve Graphic Fitness Top 50
Teal
$ 2495

Nike Court Vision Low
Mens Shoes 25
White
$ 5695

Nike Pegasus 40
Mens Running Shoes 30
Gray
$ 9795



Tracksuits
Jumpsuits & Rompers
Skirts & Dresses
Socks
Accessories & Equipment

Gender
□ Men
□ Women
□ Unisex

Kids
□ Boys
□ Girls
□ Unisex

Shop by price
□ Under $2000
☑ $2000 - $5000
□ $5000 - $10,000
□ $10,000 - $20,000

Nike Court Vision Low Next Nature
Mens Shoes 35
Black
$ 4995

Nike Standard Issue Basketball Jersey
Womens Basketball Jersey 40
White
$ 2895

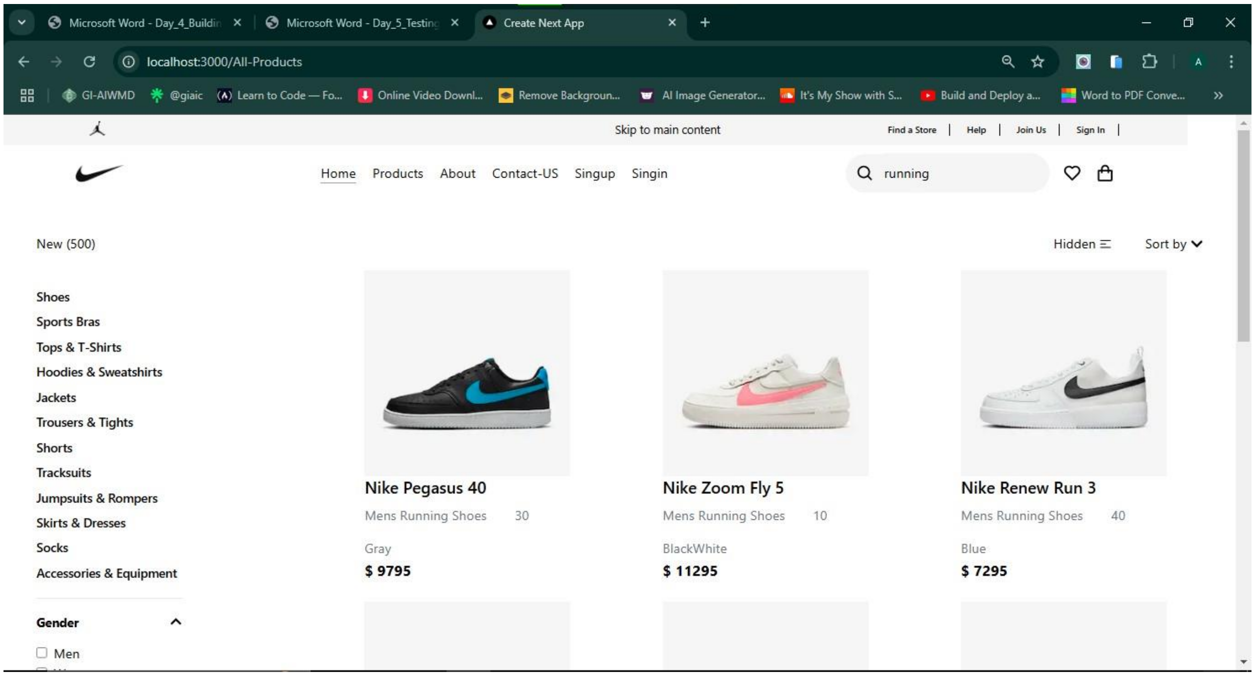
Nike Dri-FIT UV Hyverse
Mens Short-Sleeve Graphic Fitness Top 50
Teal
$ 2495

Nike Dri-FIT UV Hyverse
Mens Short-Sleeve Graphic Fitness Top 50
Teal
$ 2495

Nike Court Vision Low Next Nature
Mens Shoes 35
Black
$ 4995

Nike Standard Issue Basketball Jersey
Womens Basketball Jersey 40
White
$ 2895

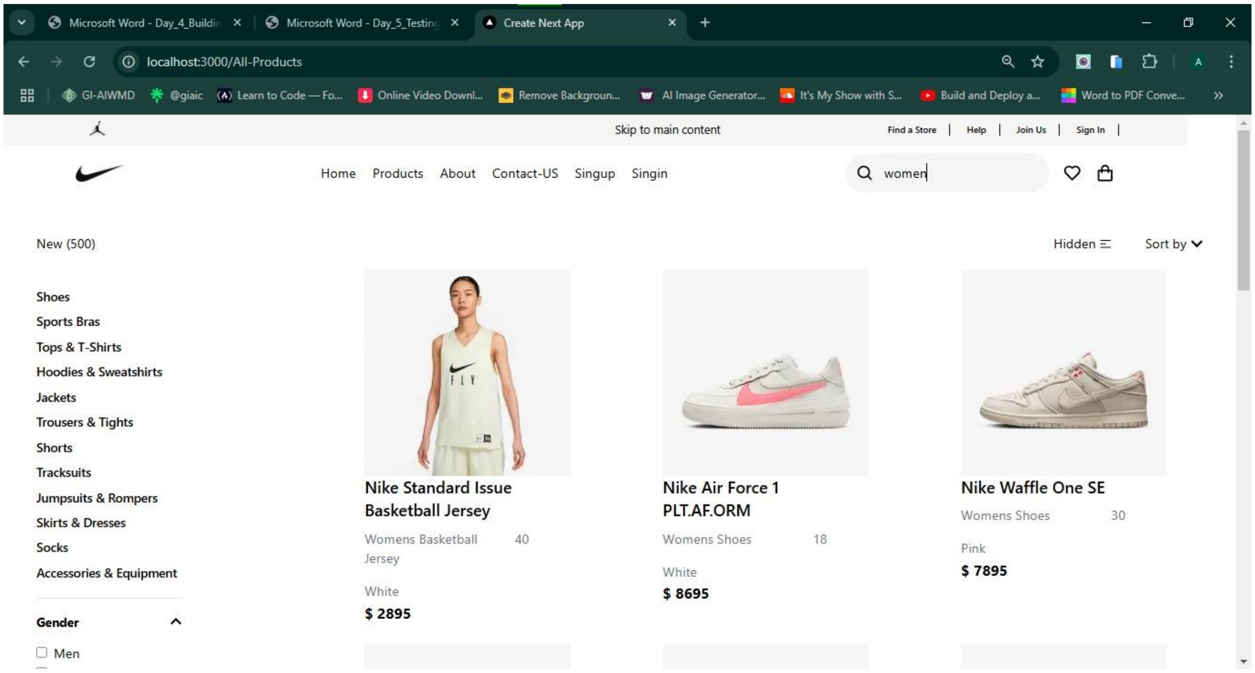
Search bar:



Nike Pegasus 40
Mens Running Shoes 30
Gray
$ 9795

Nike Zoom Fly 5
Mens Running Shoes 10
BlackWhite
$ 11295

Nike Renew Run 3
Mens Running Shoes 40
Blue
$ 7295

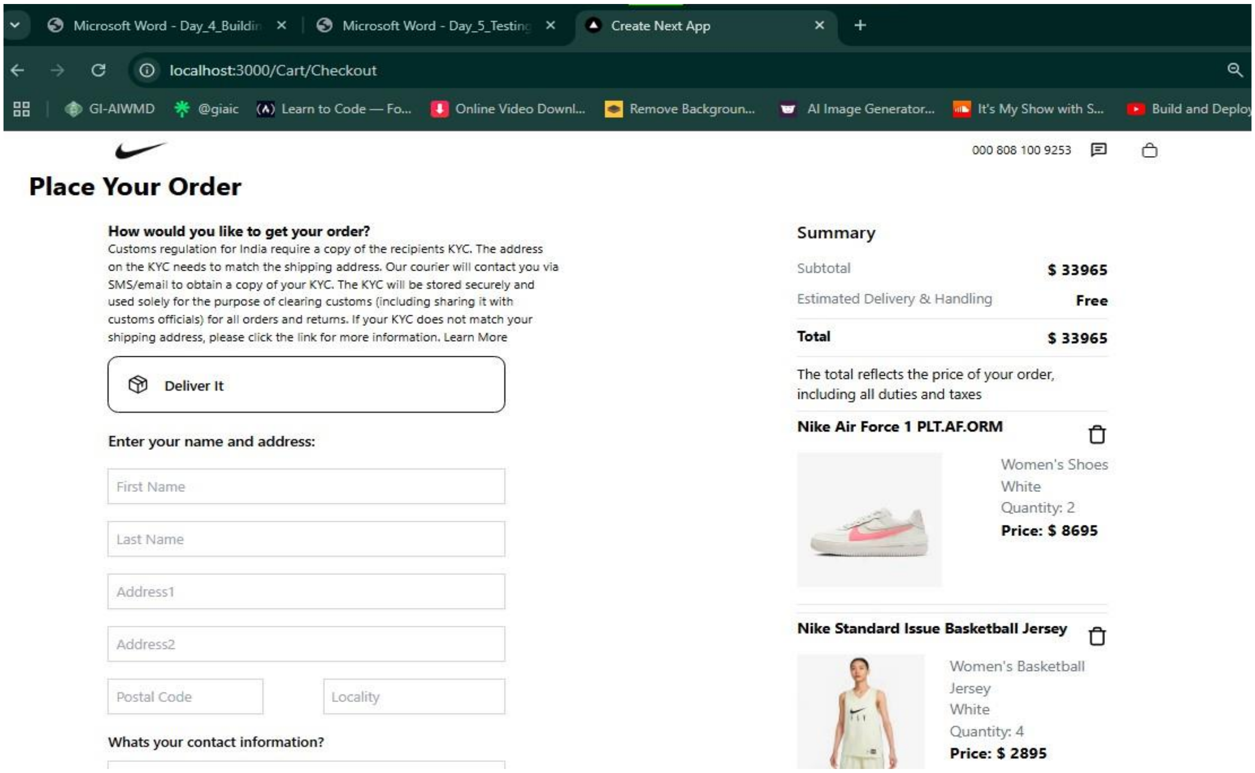


Nike Standard Issue Basketball Jersey
Womens Basketball Jersey 40
White
$ 2895

Nike Air Force 1 PLT.AF.ORM
Womens Shoes 18
White
$ 8695

Nike Waffle One SE
Womens Shoes 30
Pink
$ 7895

Checkout Flow Component:



Place Your Order

How would you like to get your order?

Customs regulation for India require a copy of the recipients KYC. The address on the KYC needs to match the shipping address. Our courier will contact you via SMS/email to obtain a copy of your KYC. The KYC will be stored securely and used solely for the purpose of clearing customs (including sharing it with customs officials) for all orders and returns. If your KYC does not match your shipping address, please click the link for more information. Learn More

Deliver It

Enter your name and address:

First Name

Last Name

Address1

Address2

Postal Code | Locality

Whats your contact information?

Summary

Subtotal	$ 33965
Estimated Delivery & Handling	Free
Total	**$ 33965**

The total reflects the price of your order, including all duties and taxes

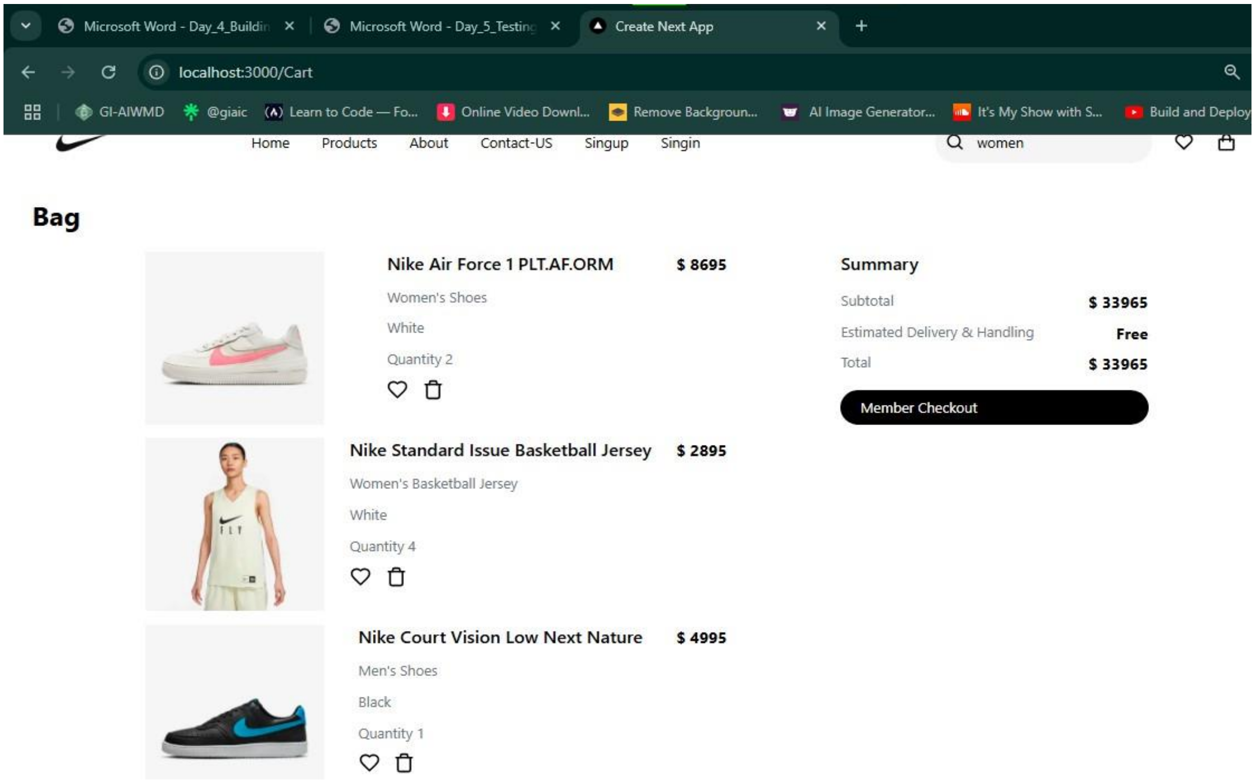
Nike Air Force 1 PLT.AF.ORM

Women's Shoes
White
Quantity: 2
Price: $ 8695

Nike Standard Issue Basketball Jersey

Women's Basketball Jersey
White
Quantity: 4
Price: $ 2895

Cart Component:



Bag

Nike Air Force 1 PLT.AF.ORM	$ 8695
Women's Shoes	
White	
Quantity 2	
Nike Standard Issue Basketball Jersey	$ 2895
Women's Basketball Jersey	
White	
Quantity 4	
Nike Court Vision Low Next Nature	$ 4995
Men's Shoes	
Black	
Quantity 1	

Summary

Subtotal	$ 33965
Estimated Delivery & Handling	Free
Total	$ 33965

Member Checkout

Code Deliverables:

Code snippets for key components:

Product Listing Components & Logic ➔

COMPONENT➔

```
import Image from "next/image";
import React from "react";
import Link from "next/link";
import { urlFor } from "@/sanity/lib/image";
interface CardProps {
  productimageUrl: string;
  productid: string;
  productName: string;
  productCategory: string;
  productinventory: number;
  productprice: number;
  productcolors: string[];

}
const Card = (props:CardProps) => {
  return (
    <>
      <div className="max-w-[1440px]">
        <Link href={`/All-Products/${props.productid}`}>
          <Imag  src={urlFor(props.productimageUrl).url()}  alt="Product Image” width={250}
            height={250}
          />
          <h3 className="text-xl font-semibold mb-2 w-[200px]">{props.productName}
          </h3>
          <p className="flex justify-between  text-gray-500 mb-4 w-[200px]">
            {props.productCategory}
            <p className="text-gray-500">{props.productinventory}</p>
          </p>
          <p className="text-gray-500">{props.productcolors}</p>
          <p className="text-lg font-bold">$ {props.productprice}</p>
        </Link>
      </div>
    </>
  );
};
export default Card;
```

LOGIC➔

```
interface Product {
  _id: string;
  productName: string;
  category: string;
  price: number;
  inventory: number;
  colors: string[];
  status: string;
  imageUrl: string;
  description: string;
}
const ProductDetails = () => {
  let { search } = useSearch();
  const api = `*[_type == "product"]{
  _id,
  productName,
  category,
  price,
  inventory,
  colors,
  status,
  "imageUrl": image.asset->url,
  description
}`;

  const [productData, setProductData] = useState<Product[]>([]);
  const [filteredProduct, setFilteredProduct] =
    useState<Product[]>(productData);
  // for fetching data
  useEffect(() => {
    async function fetchData() {
      const product = await client.fetch(api);
      console.log(product);
      setProductData(product);
      setFilteredProduct(product);
    }
    fetchData();
```

```
 }, []);
```

Search Bar Logic & Copmonent:

COOMPONENT➔

```
<div className="hidden lg:flex  lg:justify-center lg:items-center rounded-r-[300px]">
          <span className="cursor-pointer bg-[#F5F5F5]  px-3 py-3  rounded-l-[50%] ">
            <BiSearch size={23}  />
          </span>
          <input
            value={search}
            onChange={handleSearch}
            placeholder="What you are looking for?"
            className="bg-[#F5F5F5] px-3 py-3 pl-0 outline-none border-none
placeholder:text-[14px] rounded-r-[300px]"
          />
        </div>
```

LOGIC➔

```
"use client"
import React, { useContext, createContext, useState, ReactNode, } from "react";

interface SearchContext {
  search: string;
  handleSearch: (event: React.ChangeEvent<HTMLInputElement>) => void;
  setSearch: React.Dispatch<React.SetStateAction<string>>;
}

const searchbarContext = createContext<SearchContext|undefined>(undefined);

const SearchProvider = ({ children }: { children:ReactNode }) => {
  const [search, setSearch] = useState("");

  const handleSearch = (event: React.ChangeEvent<HTMLInputElement>) => {
    setSearch(event.target.value);
  };

  return (
    <searchbarContext.Provider value={{ search, handleSearch, setSearch }}>
      {children}
    </searchbarContext.Provider>
  );
};

export default SearchProvider;
export const useSearch = ():SearchContext => {
  const context = useContext(searchbarContext);
  if (!context) {
    throw new Error("useSearch must be used within a SearchProvider");
  }
  return context;
};

  // for serching data
  useEffect(() => {
    if (search === "") {
      setFilteredProduct(productData);
      return;
    } else {
      const filteredData = productData.filter((product) => {
        return (
          product.productName.toLocaleLowerCase() &&
          product.category.toLowerCase().includes(search.toLowerCase())
        );
      });
      setFilteredProduct(filteredData);
    }
  }, [search, productData]);
```

Filter Logic & Component:

COMPONETN➔

```
  <div className="flex justify-between items-center font-bold text-[15px] border-t-2 w-[50%] mt-4 py-4 pb-4">
             Gender{" "}
             <span>
               <FaChevronUp />
             </span>
           </div>
           <div>
             <input
               type="checkbox"
               className="mr-2"
               onChange={() => filterHandler({ category: "men's" })}
             />
             Men
           </div>
           <div>
             <input
```

```
                type="checkbox"
                className="mr-2"
                onChange={() => filterHandler({ category: "women's" })}
              />
              Women
            </div>
            <div>
              <input
                type="checkbox"
                className="mr-2"
                onChange={() => filterHandler({ category: "unisex" })}
              />
              Unisex
            </div>
          </div>
          <div>
            <div className="flex justify-between items-center font-bold text-[15px] border-t-2
w-[50%] mt-4 py-4 pb-4">
              Kids{" "}
              <span>
                <FaChevronUp />
              </span>
            </div>
            <div>
              <input type="checkbox" className="mr-2" />
              Boys
            </div>
            <div>
              <input type="checkbox" className="mr-2" />
              Girls
            </div>
            <div>
              <input type="checkbox" className="mr-2" />
              Unisex
            </div>
          </div>
          <div></div>
          <div>
            <div className="font-bold flex justify-between items-center text-[15px] border-t-2
w-[50%]  mt-4 py-4">
              Shop by price{" "}
```

```
 <span>
  <FaChevronUp />
 </span>
</div>
<div>
 <input
  type="checkbox"
  className="mr-2"
  onChange={(e) => filterHandler({ priceRange: 2000 })}
 />
 Under $2000
</div>
<div>
 <input
  type="checkbox"
  className="mr-2"
  onChange={(e) => filterHandler({ priceRange: 5000 })}
 />
 $2000 - $5000
</div>
<div>
 <input
  type="checkbox"
  className="mr-2"
  onChange={(e) => filterHandler({ priceRange: 10000 })}
 />
 $5000 - $10,000
 </div>
```

LOGIC➔

```
function filterHandler(filters: { priceRange?: number; category?: string }) {
  let filteredData = productData;
  if (filters.priceRange) {
   filteredData = filteredData.filter((product) => {
    return (
     filters.priceRange !== undefined && product.price < filters.priceRange
    );
   });
  }
  if (filters.category) {
   filteredData = filteredData.filter((product) => {
      const category = product.category
        .toLocaleLowerCase()
        .replace(/shoes|Running Shoes/gi, "")
        .trim();
      console.log(category);
      return (
        filters.category !== undefined &&
        category === filters.category.toLocaleLowerCase().trim()
      );
    });
  }
  setFilteredProduct(filteredData);
  console.log(filters);
}
```

Add To Cart Logic & Component:

COMPONENT➔

```
<div className="flex justify-evenly flex-col lg:flex-row">
          {cart.map((item) => {
            return (
              <div
                key={item._id}
                className="flex flex-col justify-between gap-7 pb-4 md:flex-row"
              >
                <div>
                  <Image
                    src={urlFor(item.imageUrl).url()}
                    alt="MEn"
                    width={200}
                    height={200}
                  />
                </div>
                <div className="flex flex-col gap-3">
                  <h3 className="text-xl font-semibold">
                    {item.productName}
                  </h3>
                  <p className="text-gray-500">{item.category}</p>
                  <p className="text-gray-500">{item.colors}</p>

                  <div className="flex gap-4">
```

```
           <p className="text-gray-500">Quantity {item.quantity}</p>
          </div>

          <div className="flex gap-4 cursor-pointer">
           <Heart />
           <Trash onClick={() => removeFromCart(item._id)} />
          </div>
         </div>
         <p className="text-lg font-bold">$ {item.price}</p>
        </div>
       );
      })}
```

LOGIC➔

```
const { cart, removeFromCart } = useCart();
 const [sum, setSum] = useState(0);

 useEffect(() => {
  const total = cart.reduce(
   (acc: number, item: { price: number; quantity: number }) =>
    Math.round(acc + item.price * item.quantity),
   0
  );
  setSum(total);
 }, [cart]);

//CONTEXT
"use client";
import React from "react";
import { createContext, useContext, useState, useEffect } from "react";

interface CartItem {
 _id: number;
 productName: string;
 category: string;
 price: number;
 inventory: number;
 colors: string[];
 status: string;
 imageUrl: string;
 description: string;
```

```
  quantity: number;
}

type CartContextType = {
  cart: CartItem[];
  addToCart: (product: CartItem, quantity: number) => void;
  removeFromCart: (productId: number) => void;
};

const cartContext = createContext<CartContextType | undefined>(undefined);

const CartProvider = ({ children }: { children: React.ReactNode }) => {
  const [cart, setcart] = useState<CartItem[]>(() => {
    if (typeof window !== "undefined") {
      const savedData = localStorage.getItem("cart");
      if (savedData) {
        try {
          return JSON.parse(savedData);
        } catch (e) {
          console.error("Failed to parse cart data:", e);
          return [];
        }
      }
    }
    return [];
  });
  useEffect(() => {
    localStorage.setItem("cart", JSON.stringify(cart));
  }, [cart]);

  const addToCart = (product: CartItem, quantity: number) => {
    setcart((cartitem) => {
      const itemExist = cartitem.find((item) => item._id === product._id);
      console.log(itemExist);
      if (itemExist) {
        return cartitem.map((item) => {
          if (item._id === product._id) {
            console.log("_id", item._id, product._id);
            return { ...item, quantity: quantity };
          } else {
            return item;
```

```
     }
    });
   }

   return [...cartitem, { ...product, quantity }];
  });
 };

 const removeFromCart = (productId: number) => {
  setcart((cartItem) => {
   return cartItem.filter((item) => item._id !== productId);
  });
 };
 return (
  <>
   <cartContext.Provider value={{ cart, addToCart, removeFromCart }}>
    {children}
   </cartContext.Provider>
  </>
 );
};

export const useCart = (): CartContextType => {
 const context = useContext(cartContext);
 if (!context) {
  throw new Error("useCart must be used within a CartProvider");
 }
 return context;
};
export default CartProvider;
```

Checkout Logic & Component:

Component➔

```
{cart.map((item) => {
 return (
  <>
   <div
    key={item._id}
    className="border-t-2 border-b-2 py-1 my-2 text-[17px] "
   >
```

```
        <div className="flex justify-between ">
          <h1 className="font-bold mb-5 tetx-[25px]">
            {item.productName}
          </h1>
          <button onClick={() => removeFromCart(item._id)}>
            <Trash />
          </button>
        </div>
        <div className="flex justify-between flex-col gap-7 pb-4 md:flex-row">
          <div>
            <Image
              src={urlFor(item.imageUrl).url()}
              width={160}
              height={160}
              alt="Me n"
            />
          </div>
          <div className=" gap-3">
            <p className="text-gray-500">{item.category}</p>
            <p className="text-gray-500">{item.colors}</p>
            <div className=" ">
              <p className="flex text-gray-500">
                Quantity: {item.quantity}
              </p>
              <p className="text-lg font-bold">
                Price: $ {item.price}
              </p>
            </div>
          </div>
        </div>
      </div>
    </>
  );
})}
```

Logic➔

```
const { cart, removeFromCart } = useCart();
  const [sum, setSum] = useState(0);

  useEffect(() => {
    const total = cart.reduce(
      (acc: number, item: { price: number; quantity: number }) =>
        Math.round(acc + item.price * item.quantity),
      0
    );
    setSum(total);
  }, [cart]);
```

Documentation:

Technical Report:

1. I first identifying that according to my marketplace what features I have to to implement, then I identify the reusable component like header, footer, card. After identifying implemented logic and fetch data dynamic by provider API.

2. Due to beginner and no experience and expertise but so too much problem I face that are not tell in class like it's a challenge for me to send data from ProductDetails to cart so after searching I find the solution of cartcontex,I also face api integarion issue, Component reusability for send a data and Building Logic for filter search and add to cart e.t.c.

3. Try to use Best Practice but, the lack of experience and knowledge there is bad quality code. I Try best Practice like Descriptive Naming for variable , Code Formatting, Use component for reuse , State Management for cart and searchbar.