

HACKATHON 3 (DAY 4)

In day 4 of hackathon we have to create functional components to make website dynamic.

FILTER PRODUCTS :

I have created a functionality of filter the products by its category and name and from high price to low price

```
const filterAndSortProducts = () => {  
  let filteredProducts = products.filter(product =>  
    product.title.toLowerCase().includes(searchBar.toLowerCase())  
  );  
  
  switch(sortOption) {  
    case "high-low":  
      return filteredProducts.sort((a, b) => b.price - a.price);  
    case "low-high":  
      return filteredProducts.sort((a, b) => a.price - b.price);  
    default:  
      return filteredProducts;  
  }  
};  
  
const sortedAndFilteredProducts = filterAndSortProducts();
```

PRODUCT LISTING :

All the data from the sanity integrate into product listing component to select the products for customers

```

{sortedAndFilteredProducts.length > 0 ? (
  sortedAndFilteredProducts.map((item, index) => (
    <motion.div
      key={item._id}
      className="bg-white rounded-lg shadow-lg overflow-hidden flex flex-col h-full"
      initial={{ opacity: 0, y: 50 }}
      animate={{ opacity: 1, y: 0 }}
      transition={{ duration: 0.5, delay: index * 0.1 }}
    >
      {item.slug ? (
        <Link href={` /shop/${item.slug.current || item.slug}`}>
        <div className="relative w-full">
          {item.imageUrl ? (
            <Image
              src={urlFor(item.imageUrl).url()}
              alt={item.title}
              width={300}
              height={350}
              // layout="fill"
              // objectFit="cover"
              style={{ objectFit: "cover" }}
              className="transition-transform duration-300 hover:scale-110"
            />
          ) : (
            <div className="w-full h-full flex items-center justify-center bg-gray-200">
              <p className="text-gray-500">No image available</p>
            </div>
          )}
        </div>
      ) : (
        <div className="flex flex-col flex-grow p-6">
          <h2 className="font-bold text-xl text-gray-800 mb-2 line-clamp-1">
            {item.title}
          </h2>
        </div>
      )}
    </motion.div>
  )
)}

```

SEARCH BAR :

Search bar component to search the required product for customer.

```

</div> />
<div className='mb-8 animate-fade-in w-full max-w-2xl mx-auto'>
  <div className="relative flex items-center">
    <input
      value={searchBar}
      onChange={(e) => setSearchBar(e.target.value)}
      placeholder='Search product categories...'
      className="w-full py-3 pl-4 pr-32 □ text-gray-800 ■ bg-white border-2
        □ border-gray-600 rounded-full outline-none ■ focus:border-blue-500
        focus:ring-2 ■ focus:ring-blue-200 transition-all duration-300 shadow-sm"
    />
    <div className="absolute right-2 top-1/2 transform -translate-y-1/2">
      <select
        className="appearance-none ■ bg-gray-100 border ■ border-gray-300
          □ text-gray-700 py-2 px-4 pr-8 rounded-full leading-tight
          focus:outline-none ■ focus:bg-white □ focus:border-gray-500"
        value={sortOption}
        onChange={(e) => setSortOption(e.target.value)}
      >
        <option value="relevant">Sort by: Relevant</option>
        <option value="high-low">Sort by: High to Low</option>
        <option value="low-high">Sort by: Low to High</option>
      </select>
      <div className="pointer-events-none absolute inset-y-0 right-0 flex
        items-center px-2 □ text-gray-700">
        <ChevronDown className="h-4 w-4" />
      </div>
    </div>
  </div>
</div>
<div className="mt-4 flex flex-wrap justify-center gap-2">
  {[ 'Chair', 'Sofa', "Table", "Vase", "Lamp", "Bed" ].map((category) => (
    <button
      key={category}

```

CART COMPONENT :

For cart functionality I use context api to add cart functionality in which add to cart, remove cart, increasing and decreasing quantity with total price logics present

```

components > use client > cartContext.tsx > ...
1  use client
2  import { createContext, useContext, useEffect, useState } from 'react';
3
4  interface CartItem {
5    id: number;
6    title: string;
7    price: number;
8    quantity: number;
9    image: string;
10   description:string
11  }
12
13  interface CartContextType {
14    cartItems: CartItem[];
15    addToCart: (item: CartItem) => void;
16    removeFromCart: (id: number) => void;
17    totalPrice: () => number
18    increaseQuantity: (id: number)=> void
19    decreaseQuantity: (id: number)=> void
20  }
21
22  const CartContext = createContext<CartContextType | undefined>(undefined);
23
24  export const useCart = ():CartContextType => {
25    const context = useContext(CartContext);
26    if (!context) {
27      throw new Error('useCart must be used within a CartProvider');
28    }
29    return context;
30  };
31
32  export const CartProvider = ({ children }: { children: React.ReactNode }) => {
33    const getInitialCart = (): CartItem[] => {
34      if (typeof window !== "undefined") {
35        const storedCart = localStorage.getItem("cart");
36        return storedCart ? JSON.parse(storedCart) : [];
37      }
38    };
39
40    const [cartItems, setCartItems] = useState<CartItem[]>(getInitialCart());
41    const [totalPrice, setTotalPrice] = useState<number>(0);
42
43    const addToCart = (item: CartItem) => {
44      const existingItem = cartItems.find((i) => i.id === item.id);
45      if (existingItem) {
46        setCartItems(
47          cartItems.map((i) =>
48            i.id === item.id ? { ...i, quantity: i.quantity + 1 } : i
49          )
50        );
51      } else {
52        setCartItems([...cartItems, item]);
53      }
54    };
55
56    const removeFromCart = (id: number) => {
57      setCartItems(cartItems.filter((i) => i.id !== id));
58    };
59
60    const increaseQuantity = (id: number) => {
61      const existingItem = cartItems.find((i) => i.id === id);
62      if (existingItem) {
63        setCartItems(
64          cartItems.map((i) =>
65            i.id === id ? { ...i, quantity: i.quantity + 1 } : i
66          )
67        );
68      }
69    };
70
71    const decreaseQuantity = (id: number) => {
72      const existingItem = cartItems.find((i) => i.id === id);
73      if (existingItem & & existingItem.quantity > 0) {
74        setCartItems(
75          cartItems.map((i) =>
76            i.id === id ? { ...i, quantity: i.quantity - 1 } : i
77          )
78        );
79      }
80    };
81
82    const calculateTotalPrice = () => {
83      const total = cartItems.reduce(
84        (total, item) => total + item.price * item.quantity, 0
85      );
86      setTotalPrice(total);
87    };
88
89    useEffect(() => {
90      calculateTotalPrice();
91    }, [cartItems]);
92
93    return (
94      <CartContext.Provider value={{
95        cartItems,
96        addToCart,
97        removeFromCart,
98        increaseQuantity,
99        decreaseQuantity,
100        totalPrice,
101      }}>
102        {children}
103      </CartContext.Provider>
104    );
105  };

```

CHECKOUT FLOW :

Checkout page include the order details , shipping details and payment method to buy products for customers

checkout > ✖ page.tsx > ...

```
"use client"

import { useState, useRef } from "react";
import { useCart } from "@components/cartContext";
import { motion, AnimatePresence } from "framer-motion";
import { X, Loader2 } from 'lucide-react';

interface ShippingDetails {
  name: string;
  phone: string;
  addressLine1: string;
  addressLine2: string;
  city: string;
  state: string;
  postalCode: string;
  country: string;
  addressResidentialIndicator: "yes" | "no";
  paymentMethod: "creditCard" | "paypal" | "bankTransfer";
}

interface OrderConfirmation {
  shipmentId: string;
  // rateId: string;
  shipDate: string;
  userDetails: ShippingDetails;
}

const OrderConfirmationDrawer = ({ isOpen, onClose, orderDetails }: { isOpen:
boolean; onClose: () => void; orderDetails: OrderConfirmation }) => {
  return (
    <AnimatePresence>
      {isOpen && (
        <motion.div
          initial={{ x: "100%" }}
          animate={{ x: 0 }}
        >
```

ORDER PLACED:

By submitting the order and shipping details, a drawer of shipping order will show on the screen

Order Confirmation



Shipment Details

Shipment ID: se-44743301

Ship Date: 1/20/2025

Shipping Address

Bashar

Fedral B Area Block 10

Karachi, Pakistan, US 12345

Pakistan

Contact Information

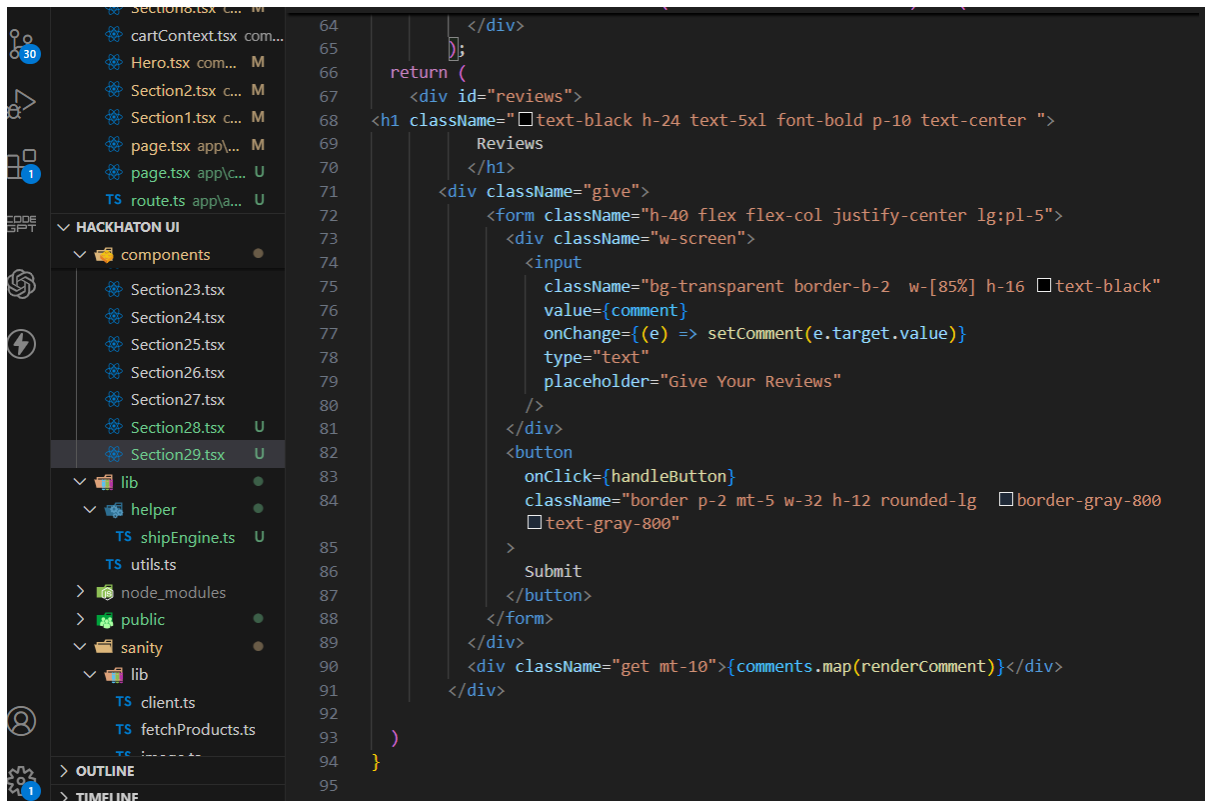
Phone: 03042985456

Payment Method

creditCard

REVIEWS:

Customer give reviews on products



RELATED PRODUCTS:

Filter the products for featured and latest items showing on the home page

FOOTER AND HEADER:

Footer and Header components are almost same for whole website for links of the pages and sections

components > Footer.tsx > ...

```
1 import React from 'react'
2 import { Button } from "@/components/ui/button"
3 import Link from "next/link";
4 export default function Footer() {
5   return (
6     <div>
7       <link
8         href="https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css"
9         rel="stylesheet"
10      />
11
12     <div className='md:w-screen md:h-[488px] md:flex justify-center items-center'>
13       <div className='md:h-full md:w-[1050px] flex flex-col justify-center '>
14         <div className='flex md:justify-between justify-around'>
15           <div className="logo font-bold text-[#252B42] text-[24px]">
16             Bandage
17
18           </div>
19           <div className="links text-2xl flex gap-3 text-[#23A6F0]">
20             <i className='bx bxl-facebook-circle' ></i>
21             <i className='bx bxl-instagram'></i>
22             <i className='bx bxl-twitter' ></i>
23
24           </div>
25         </div>
26         <div className="footer mt-10 md:flex gap-14 grid grid-cols-2 p-5 md:p-0 pl
27           <div className='flex flex-col gap-3'>
28             <h1 className='font-bold'>Company Info</h1>
29             <Link href={"/about"}>
30               <p>About Us</p>
31
32             </Link>
33             <Link href={"/pricing"}>
34               <p>Carrier</p>
35
36             </Link>
```



```

components > Header2.tsx > Header2
1  "use client";
2  import React from "react";
3  import { useState } from "react";
4  import Link from "next/link";
5  import { useCart } from "@components/cartContext";
6
7  export default function Header2() {
8      const [isOpen, setIsOpen] = useState(false);
9
10     const handleMenu = () => {
11         setIsOpen(!isOpen);
12     };
13     const handleClose = () => {
14         setIsOpen(false);
15     };
16
17     const { cartItems } = useCart()
18     const cartCount = cartItems.length
19
20     return (
21         <div>
22             <div
23                 className=" w-screen h-[65px] flex justify-center
24                 items-center"
25             >
26                 <div className=" w-[1049px] flex items-center justify-around md:justify-between"
27                     <div className="logo">
28                         <h1 className="font-bold text-[24px]">Bandage</h1>
29                     </div>
30                     <div className="nav hidden md:flex ">
31                         <ul className=" flex gap-5">
32                             <Link href={"/home"}>
33                                 <li>Home</li>
34                             </Link>
35                             <Link href={"/shop"}>
36                                 <li>Shop</li>

```

NOTIFICATION :

Notification is used to mentioned the messages like product is added in the cart etc

HELP CENTER COMPONENT:

Customers contact us from the contact page by filling the form and ask questions about the products

```

components > Section28.tsx > Section28
1  "use client"
2  import { useState } from "react";
3
4  export default function Section28() {
5      const [formData, setFormData] = useState({
6          name: "",
7          email: "",
8          phone: "",
9          issueType: "",
10         message: "",
11     });
12
13     const handleChange = (e:any) => {
14         setFormData({ ...formData, [e.target.name]: e.target.value });
15     };
16
17     const handleSubmit = async (e:any) => {
18         e.preventDefault();
19
20         const formData = {
21             name: e.target.name.value,
22             email: e.target.email.value,
23             message: e.target.message.value,
24         };
25
26         try {
27             const response = await fetch('/api/sendEmail', {
28                 method: 'POST',
29                 headers: { 'Content-Type': 'application/json' },
30                 body: JSON.stringify(formData),
31             });
32
33             const result = await response.json();
34
35             if (!response.ok) {
36                 throw new Error(result.message || 'Something went wrong!');

```

BEST PRACTICES :

- Create each section in a component which help to handle the section easily and we also reuse it on other pages.

- Use context api for cart functionality like add to cart functionality, cart items, total price to use it into multiple pages easily
- Search Bar is used to filter the products by using its name and category