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}
```

SEARCH BAR :

Search bar component to search the required product for customer.

```
<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 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 > cartContext.tsx > ...

```tsx
import { createContext, useContext, useEffect, useState } from 'react';

interface CartItem {
  id: number;
  title: string;
  price: number;
  quantity: number;
  image: string;
  description:string
}

interface CartContextType {
  cartItems: CartItem[];
  addToCart: (item: CartItem) => void;
  removeFromCart: (id: number) => void;
  totalPrice: () => number
  increaseQuantity: (id: number)=> void
  decreaseQuantity: (id: number)=> void
}

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

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

export const CartProvider = ({ children }: { children: React.ReactNode }) => {
    const getInitialCart = (): CartItem[] => {
        if (typeof window !== "undefined") {
          const storedCart = localStorage.getItem("cart");
```

CHECKOUT FLOW :

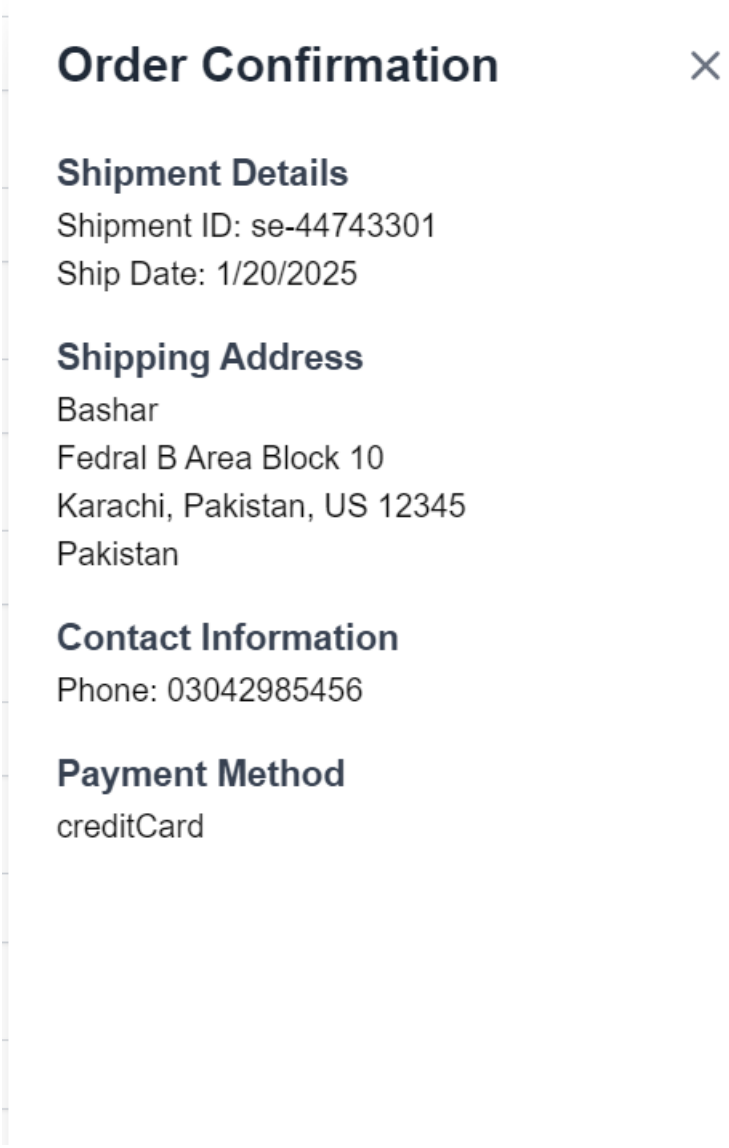
Checkout page include the order details , shipping details and payment method to buy products for customers

checkout > page.tsx > ...

```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%" }}
```

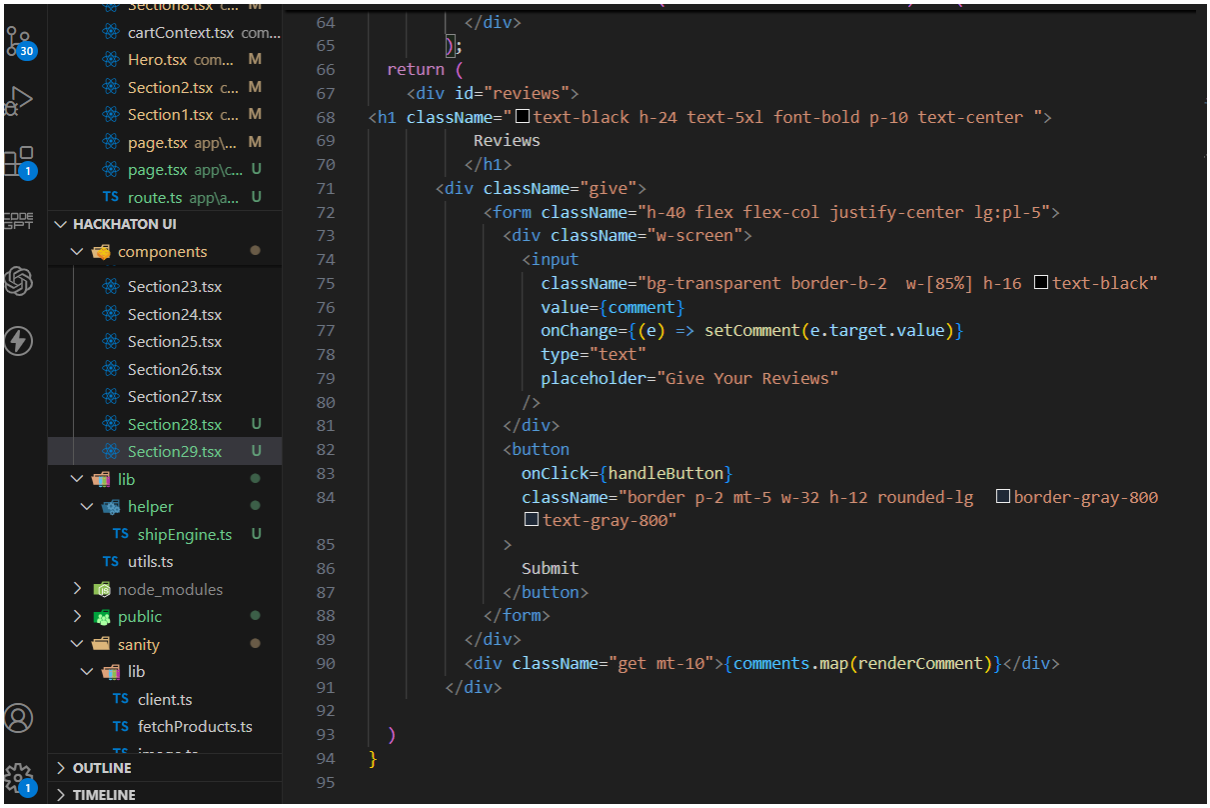
ORDER PLACED:

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



REVIEWS:

Customer give reviews on products



RELATED PRODUCTS:

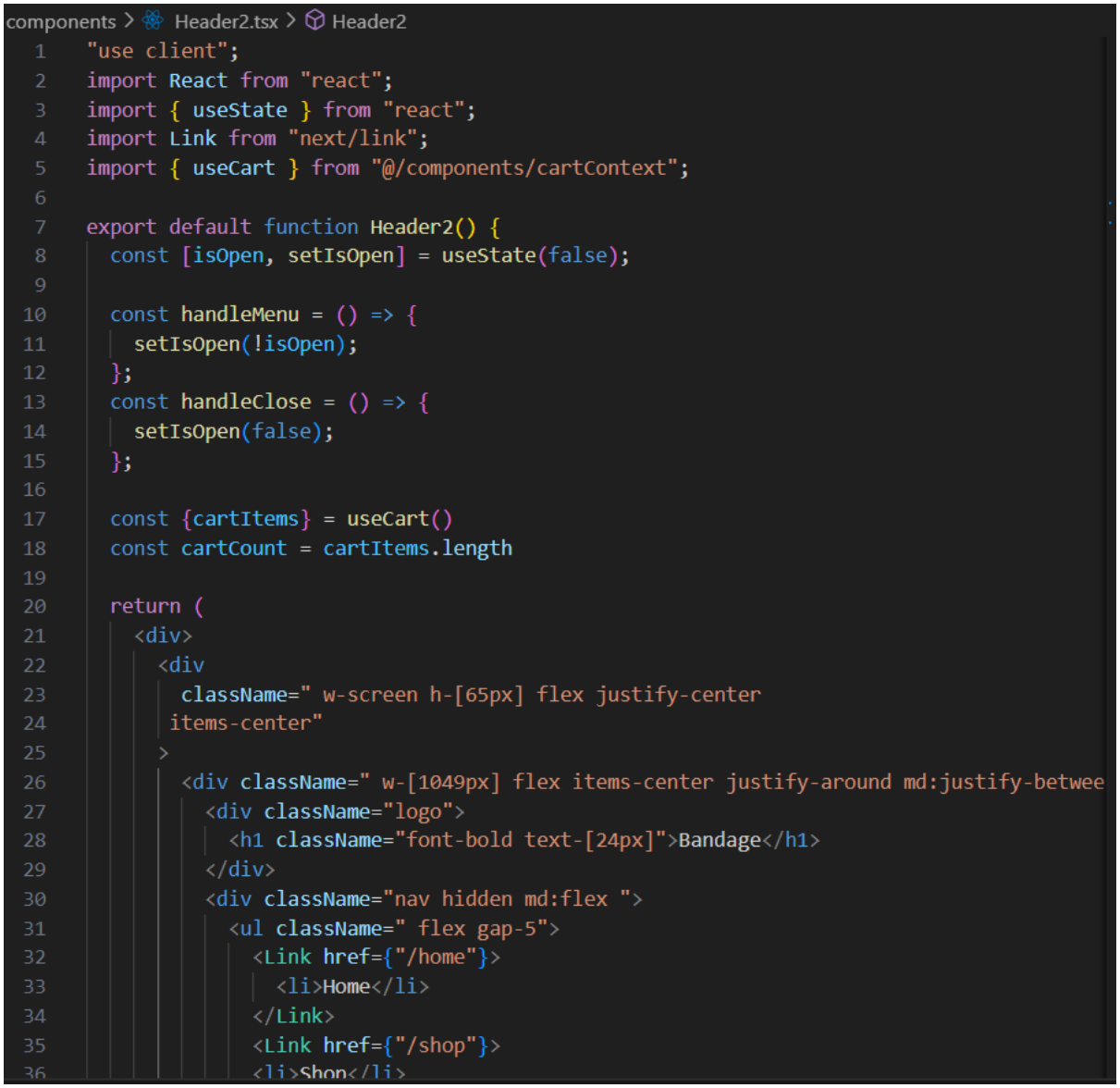
Filter the products for featured and latest items showing on the home page

FOOTER AND HEADER:

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

components > Footer.tsx > ...

```tsx
import React from 'react'
import { Button } from "@/components/ui/button"
import Link from "next/link";
export default function Footer() {
  return (
    <div>
      <link
        href="https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css"
        rel="stylesheet"
       />

      <div className='md:w-screen md:h-[488px] md:flex justify-center items-center'>
       <div className='md:h-full md:w-[1050px] flex flex-col justify-center '>
          <div className='flex  md:justify-between justify-around'>
            <div className="logo font-bold text-[#252B42] text-[24px]">
            Bandage

            </div>
            <div className="links text-2xl flex gap-3 text-[#23A6F0]">
            <i className='bx bxl-facebook-circle'  ></i>
            <i className='bx bxl-instagram'></i>
            <i className='bx bxl-twitter'  ></i>

            </div>
          </div>
          <div className="footer mt-10 md:flex gap-14 grid grid-cols-2 p-5 md:p-0 pl
            <div className='flex flex-col gap-3'>
            <h1 className='font-bold'>Company Info</h1>
            <Link href={"/about"}>
            <p>About Us</p>

          </Link>
          <Link href={"/pricing"}>
            <p>Carrier</p>
```



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

```tsx
"use client"
import { useState } from "react";

export default function Section28() {
  const [formData, setFormData] = useState({
    name: "",
    email: "",
    phone: "",
    issueType: "",
    message: "",
  });

  const handleChange = (e:any) => {
    setFormData({ ...formData, [e.target.name]: e.target.value });
  };

const handleSubmit = async (e:any) => {
    e.preventDefault();

    const formData = {
      name: e.target.name.value,
      email: e.target.email.value,
      message: e.target.message.value,
    };

    try {
      const response = await fetch('/api/sendEmail', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify(formData),
      });

      const result = await response.json();

      if (!response.ok) {
        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