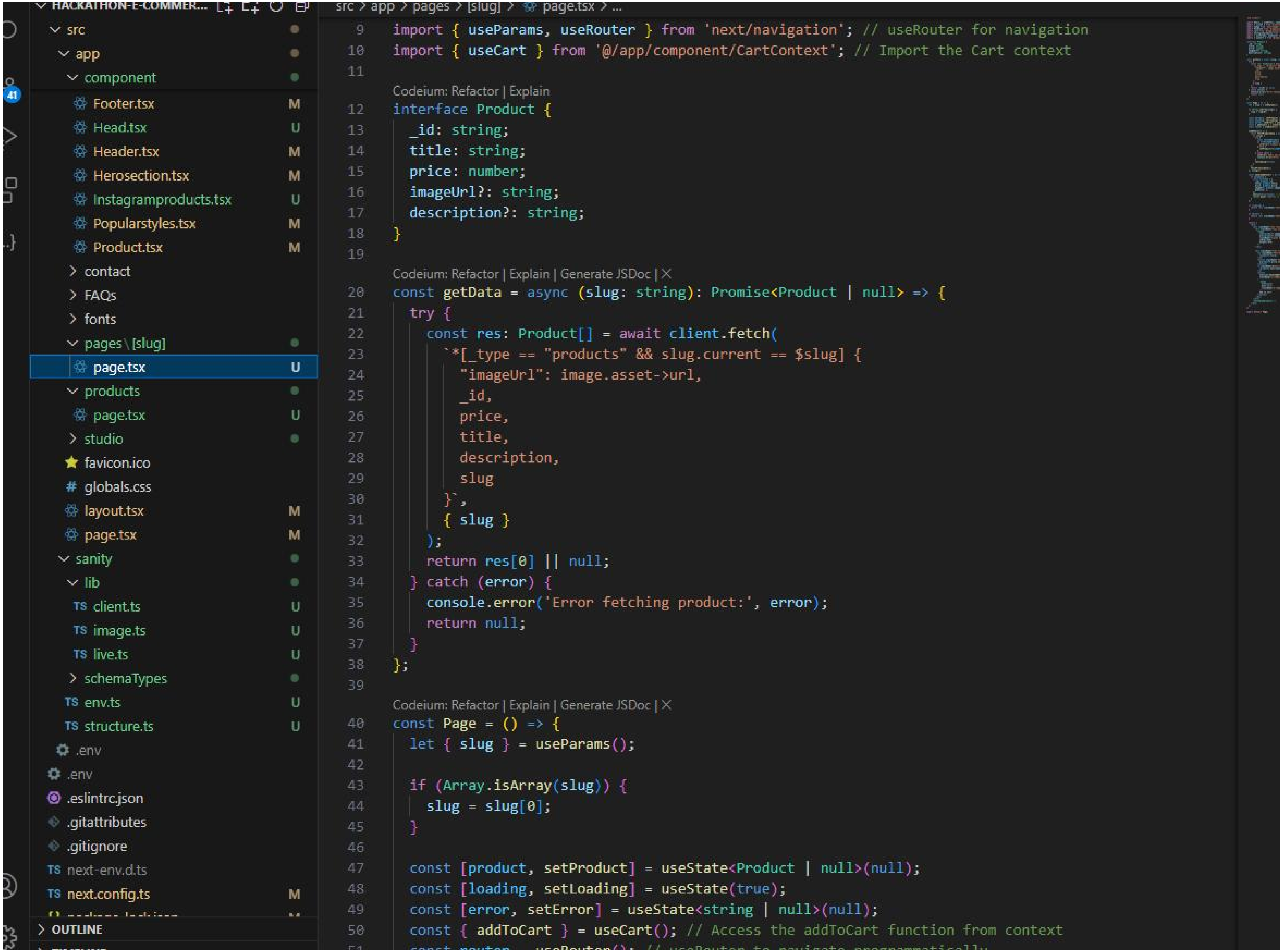
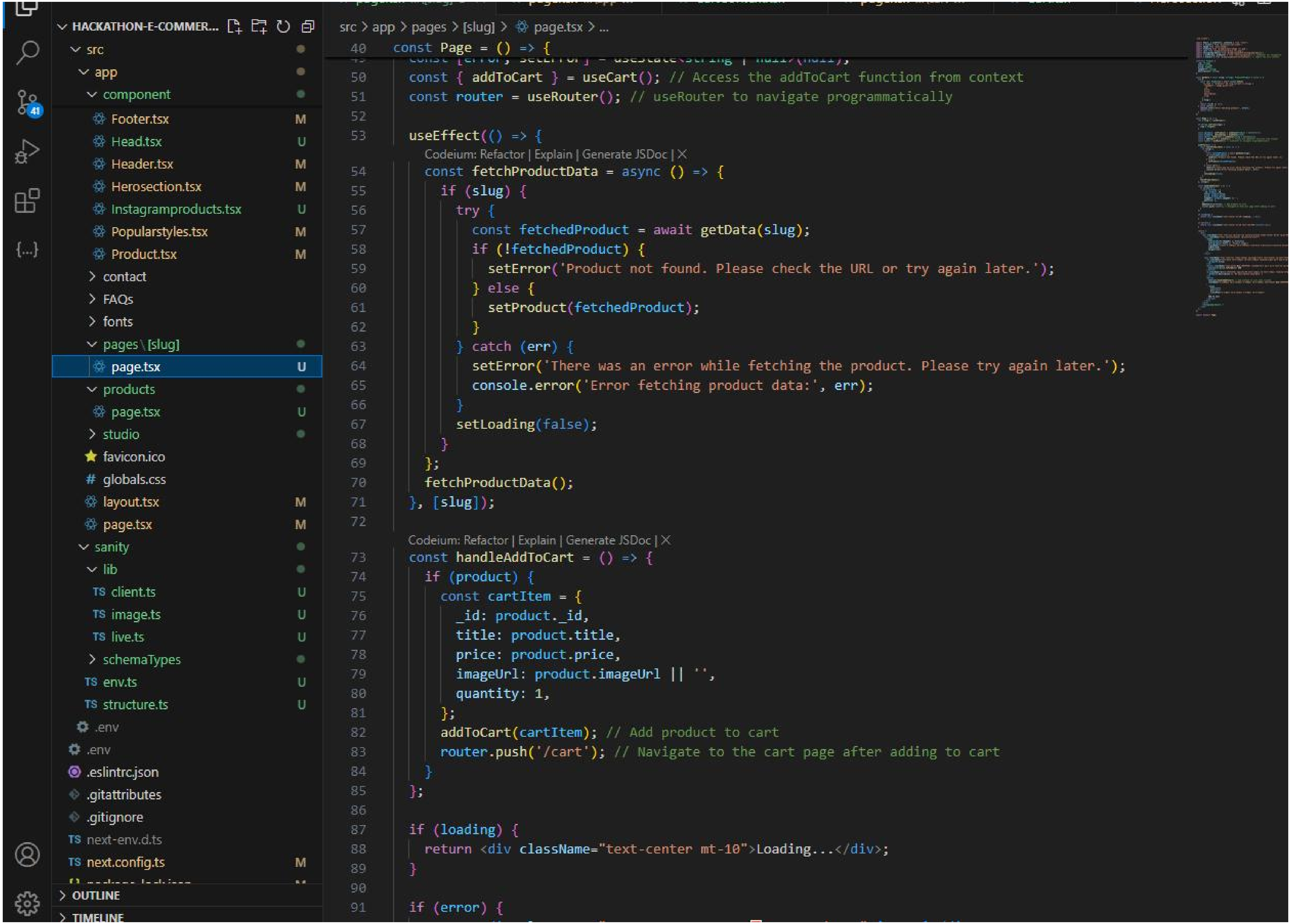
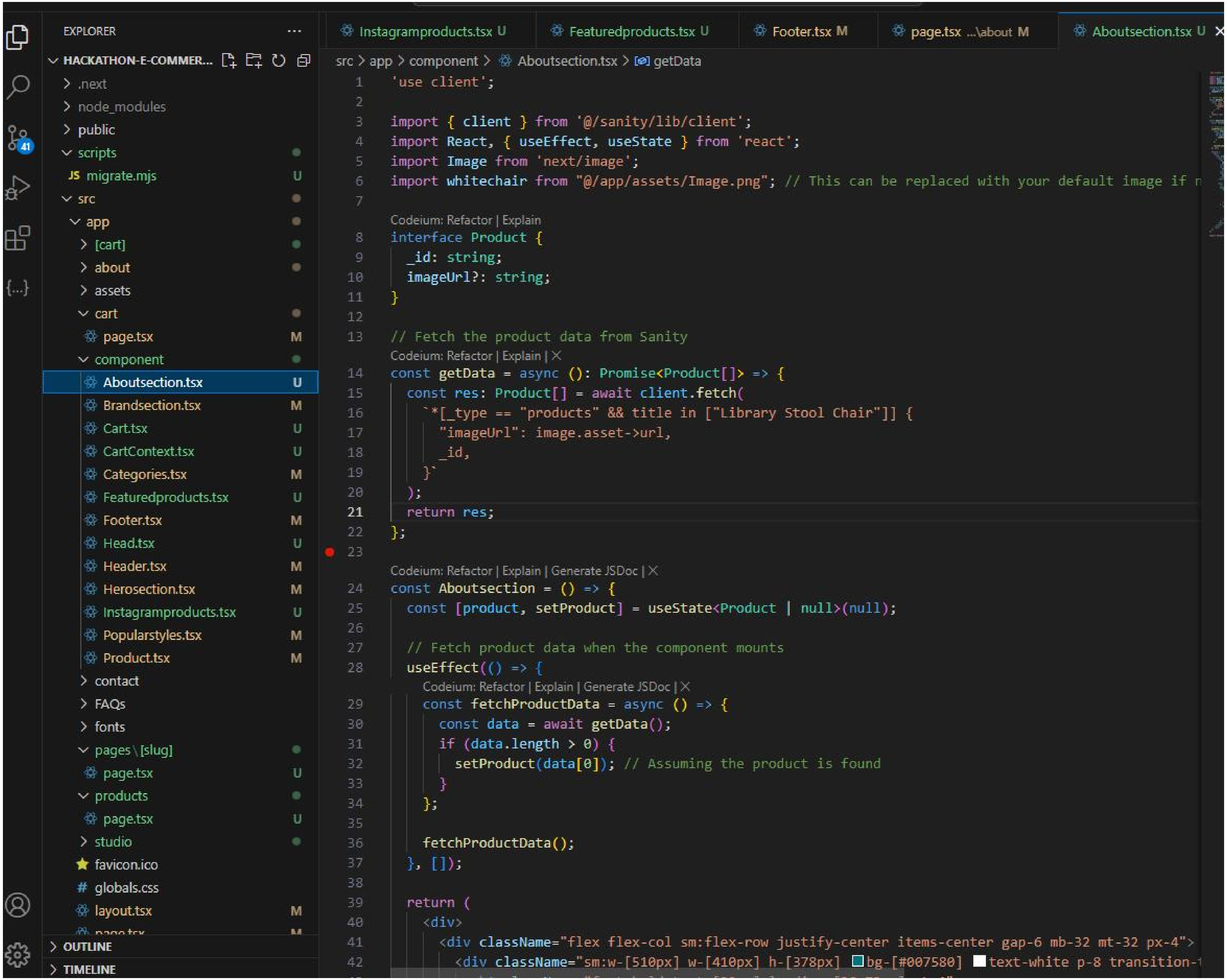


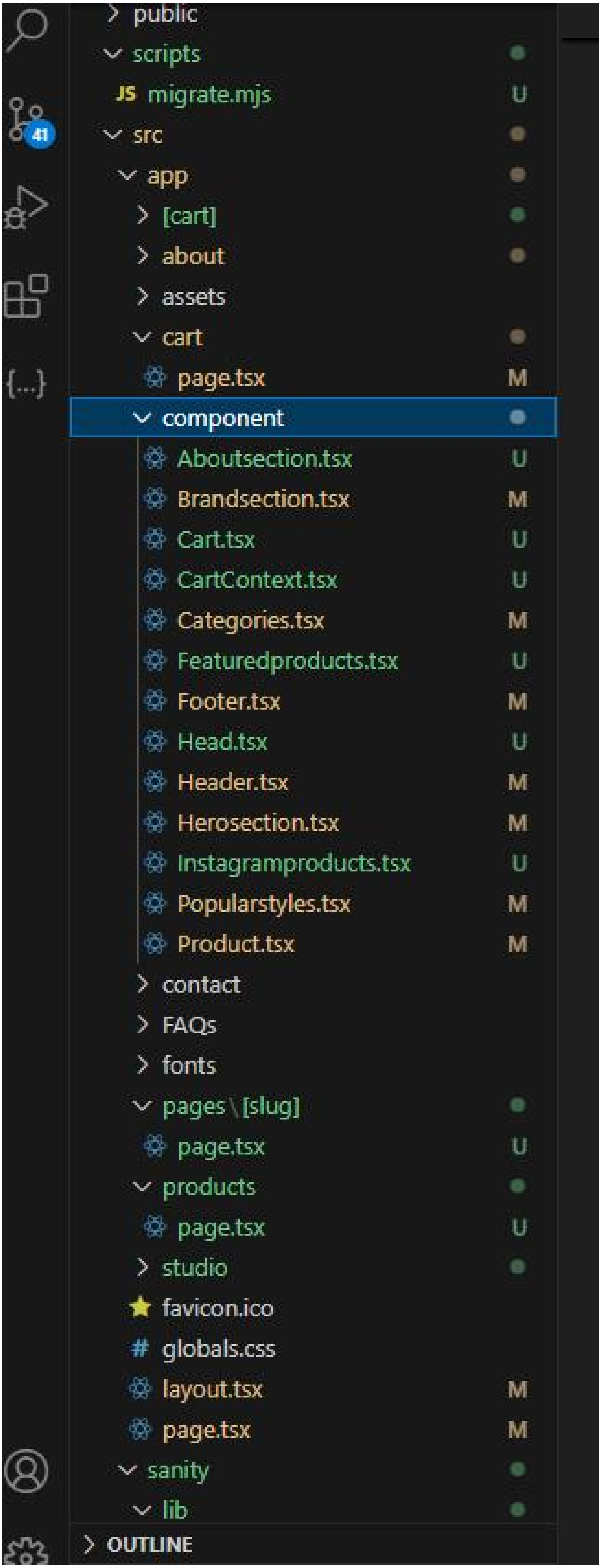
```tsx
import { useParams, useRouter } from 'next/navigation'; // useRouter for navigation
import { useCart } from '@/app/component/CartContext'; // Import the Cart context

interface Product {
  _id: string;
  title: string;
  price: number;
  imageUrl?: string;
  description?: string;
}

const getData = async (slug: string): Promise<Product | null> => {
  try {
    const res: Product[] = await client.fetch(
      `*[_type == "products" && slug.current == $slug] {
        "imageUrl": image.asset->url,
        _id,
        price,
        title,
        description,
        slug
      }`,
      { slug }
    );
    return res[0] || null;
  } catch (error) {
    console.error('Error fetching product:', error);
    return null;
  }
};

const Page = () => {
  let { slug } = useParams();

  if (Array.isArray(slug)) {
    slug = slug[0];
  }

  const [product, setProduct] = useState<Product | null>(null);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState<string | null>(null);
  const { addToCart } = useCart(); // Access the addToCart function from context
  const router = useRouter(); // useRouter to navigate programmatically
```



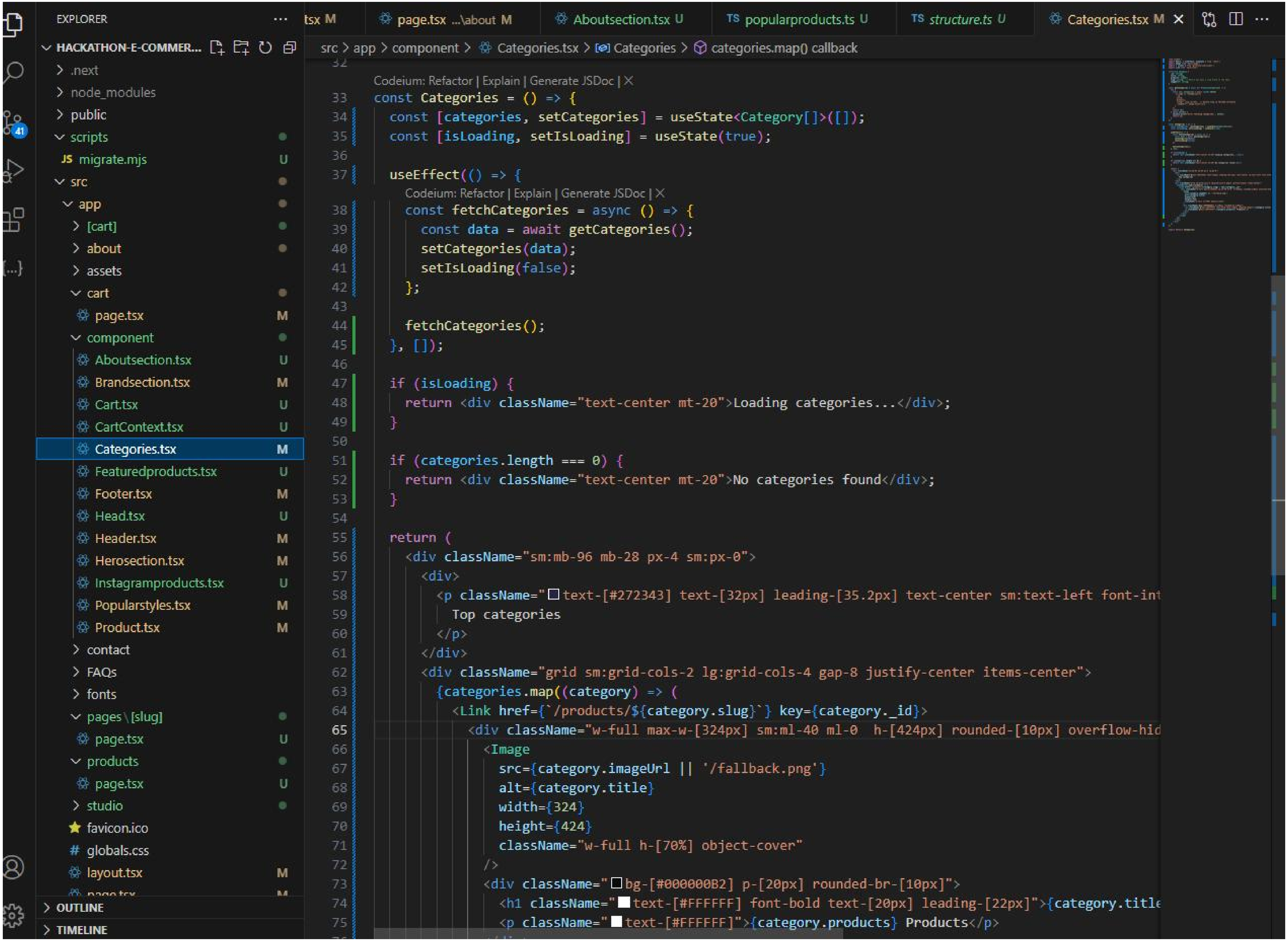
```tsx
const Page = () => {
  const [error, setError] = useState<string | null>(null);
  const { addToCart } = useCart(); // Access the addToCart function from context
  const router = useRouter(); // useRouter to navigate programmatically

  useEffect(() => {
    const fetchProductData = async () => {
      if (slug) {
        try {
          const fetchedProduct = await getData(slug);
          if (!fetchedProduct) {
            setError('Product not found. Please check the URL or try again later.');
          } else {
            setProduct(fetchedProduct);
          }
        } catch (err) {
          setError('There was an error while fetching the product. Please try again later.');
          console.error('Error fetching product data:', err);
        }
        setLoading(false);
      }
    };
    fetchProductData();
  }, [slug]);

  const handleAddToCart = () => {
    if (product) {
      const cartItem = {
        _id: product._id,
        title: product.title,
        price: product.price,
        imageUrl: product.imageUrl || '',
        quantity: 1,
      };
      addToCart(cartItem); // Add product to cart
      router.push('/cart'); // Navigate to the cart page after adding to cart
    }
  };

  if (loading) {
    return <div className="text-center mt-10">Loading...</div>;
  }

  if (error) {
```



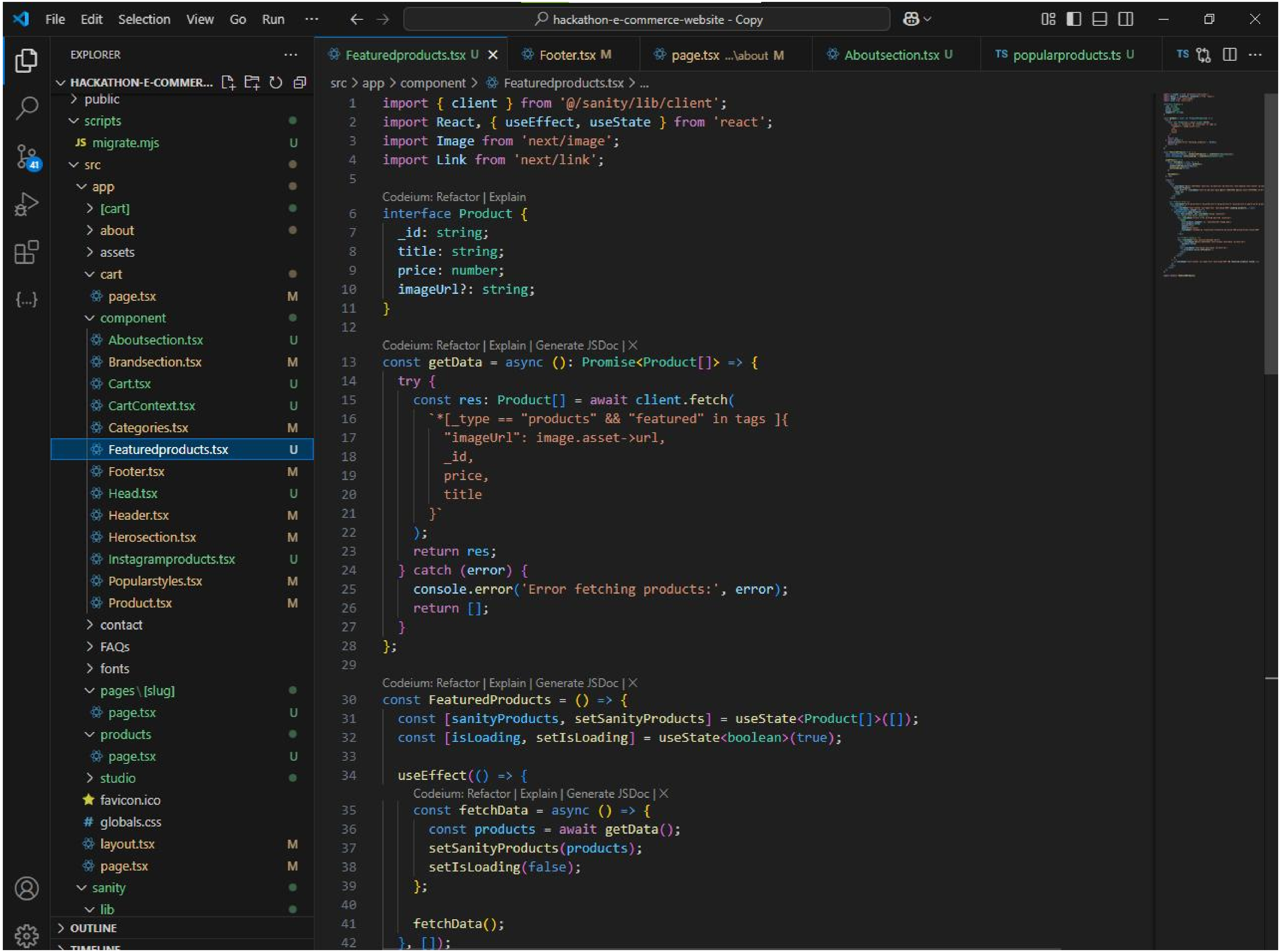
```tsx
'use client';

import { client } from '@/sanity/lib/client';
import React, { useEffect, useState } from 'react';
import Image from 'next/image';
import whitechair from "@/app/assets/Image.png"; // This can be replaced with your default image if n

interface Product {
  _id: string;
  imageUrl?: string;
}

// Fetch the product data from Sanity
const getData = async (): Promise<Product[]> => {
  const res: Product[] = await client.fetch(
    `*[_type == "products" && title in ["Library Stool Chair"]] {
      "imageUrl": image.asset->url,
      _id,
    }`
  );
  return res;
};

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

  // Fetch product data when the component mounts
  useEffect(() => {
    const fetchProductData = async () => {
      const data = await getData();
      if (data.length > 0) {
        setProduct(data[0]); // Assuming the product is found
      }
    };

    fetchProductData();
  }, []);

  return (
    <div>
      <div className="flex flex-col sm:flex-row justify-center items-center gap-6 mb-32 mt-32 px-4">
        <div className="sm:w-[510px] w-[410px] h-[378px] bg-[#007580] text-white p-8 transition-
```



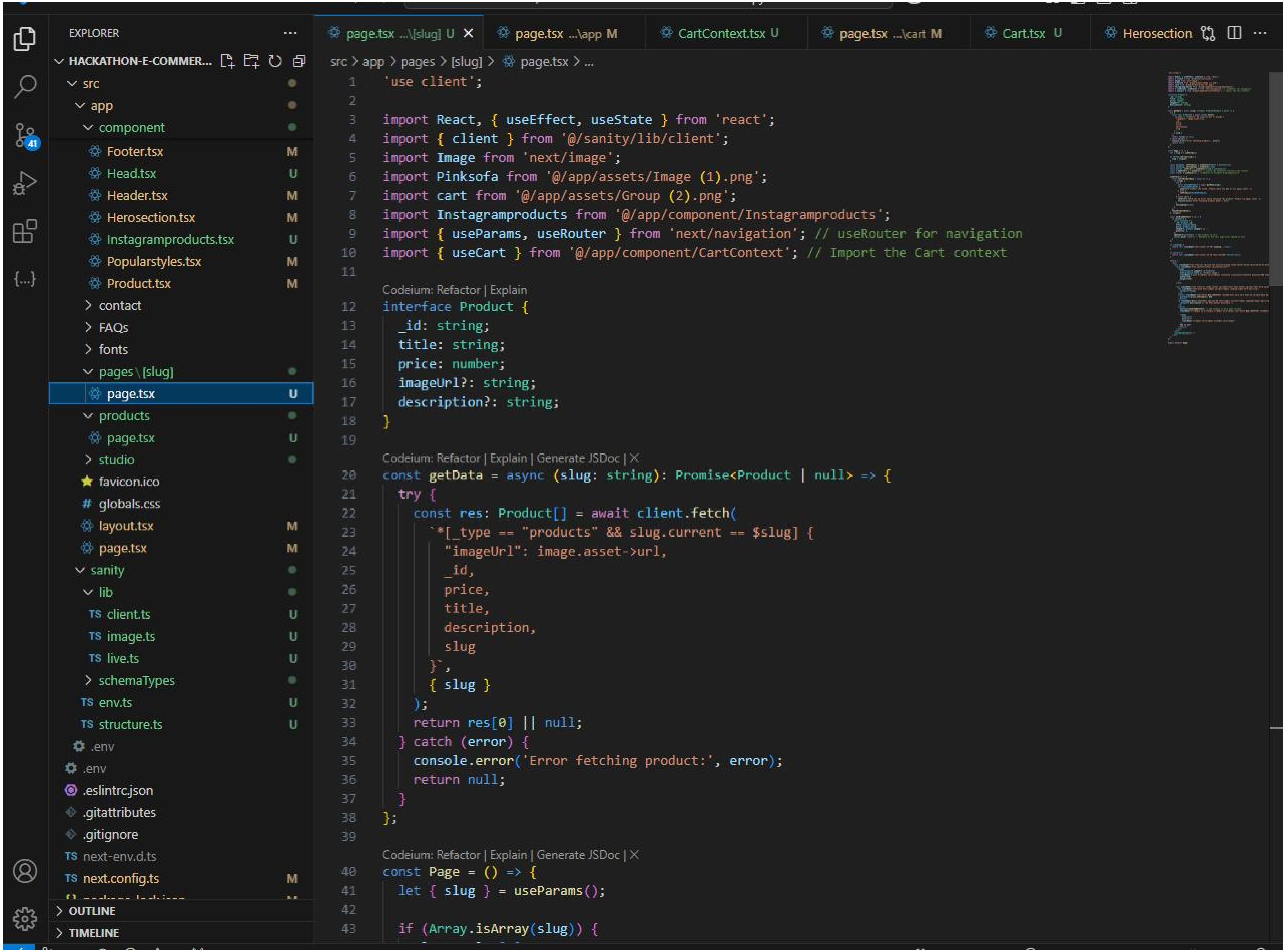
```
  > public
  v scripts                          ●
    JS migrate.mjs                   U
  v src                              ●
    v app                            ●
      > [cart]                       ●
      > about                        ●
      > assets
      v cart                         ●
        page.tsx                     M
      v component                    ●
        Aboutsection.tsx             U
        Brandsection.tsx             M
        Cart.tsx                     U
        CartContext.tsx              U
        Categories.tsx               M
        Featuredproducts.tsx         U
        Footer.tsx                   M
        Head.tsx                     U
        Header.tsx                   M
        Herosection.tsx              M
        Instagramproducts.tsx        U
        Popularstyles.tsx            M
        Product.tsx                  M
      > contact
      > FAQs
      > fonts
      v pages \ [slug]               ●
        page.tsx                     U
      v products                     ●
        page.tsx                     U
      > studio                       ●
      favicon.ico
      # globals.css
      layout.tsx                     M
      page.tsx                       M
    v sanity                         ●
      v lib                          ●
> OUTLINE
```



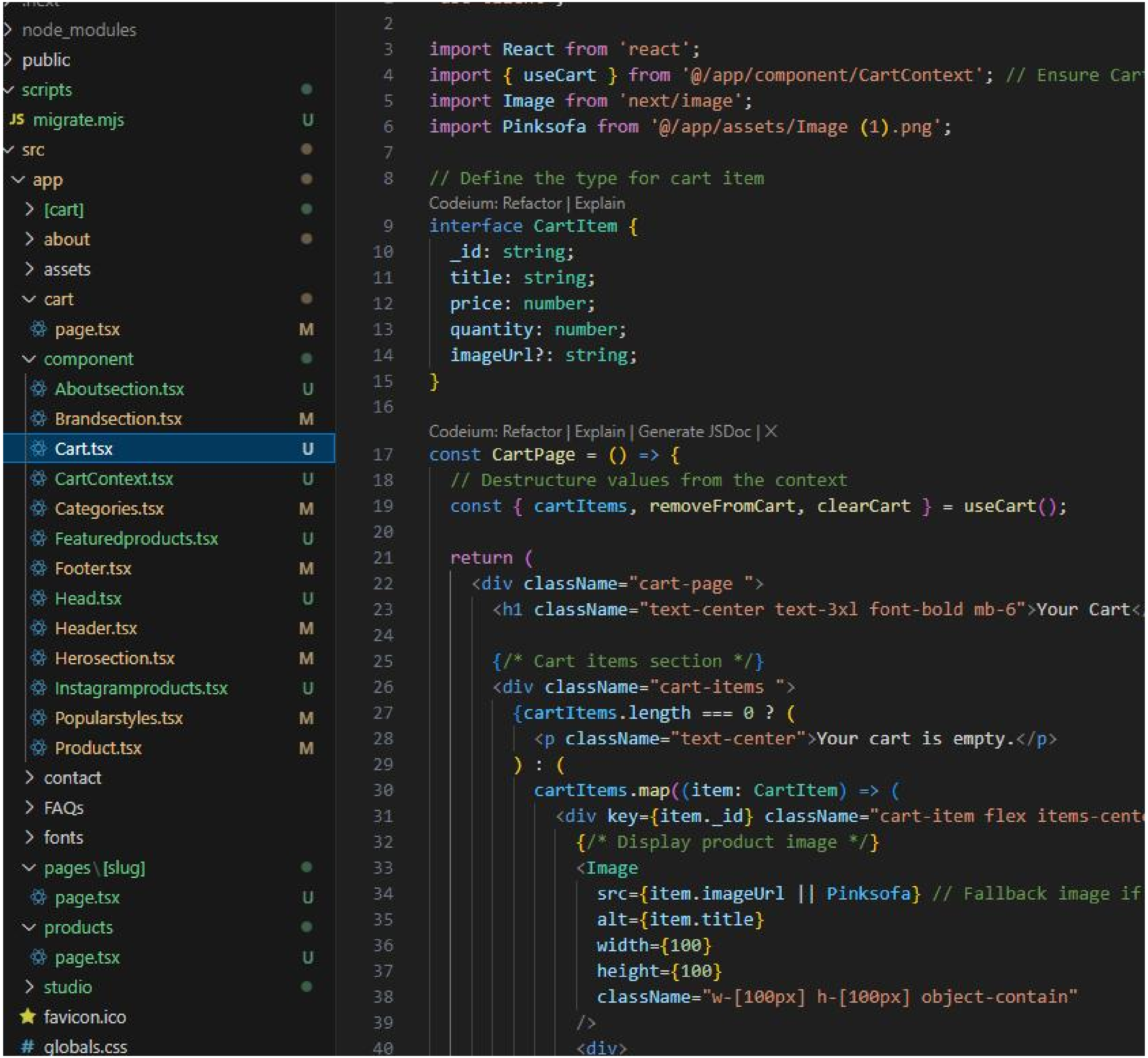
```
Codeium: Refactor | Explain | Generate JSDoc | ×
const Categories = () => {
  const [categories, setCategories] = useState<Category[]>([]);
  const [isLoading, setIsLoading] = useState(true);

  useEffect(() => {
    Codeium: Refactor | Explain | Generate JSDoc | ×
    const fetchCategories = async () => {
      const data = await getCategories();
      setCategories(data);
      setIsLoading(false);
    };

    fetchCategories();
  }, []);

  if (isLoading) {
    return <div className="text-center mt-20">Loading categories...</div>;
  }

  if (categories.length === 0) {
    return <div className="text-center mt-20">No categories found</div>;
  }

  return (
    <div className="sm:mb-96 mb-28 px-4 sm:px-0">
      <div>
        <p className="□text-[#272343] text-[32px] leading-[35.2px] text-center sm:text-left font-int
          Top categories
        </p>
      </div>
      <div className="grid sm:grid-cols-2 lg:grid-cols-4 gap-8 justify-center items-center">
        {categories.map((category) => (
          <Link href={`/products/${category.slug}`} key={category._id}>
            <div className="w-full max-w-[324px] sm:ml-40 ml-0  h-[424px] rounded-[10px] overflow-hid
              <Image
                src={category.imageUrl || '/fallback.png'}
                alt={category.title}
                width={324}
                height={424}
                className="w-full h-[70%] object-cover"
              />
              <div className="□bg-[#000000B2] p-[20px] rounded-br-[10px]">
                <h1 className="■text-[#FFFFFF] font-bold text-[20px] leading-[22px]">{category.title
                <p className="■text-[#FFFFFF]">{category.products} Products</p>
```



```tsx
import { client } from '@/sanity/lib/client';
import React, { useEffect, useState } from 'react';
import Image from 'next/image';
import Link from 'next/link';

interface Product {
  _id: string;
  title: string;
  price: number;
  imageUrl?: string;
}

const getData = async (): Promise<Product[]> => {
  try {
    const res: Product[] = await client.fetch(
      `*[_type == "products" && "featured" in tags ]{
        "imageUrl": image.asset->url,
        _id,
        price,
        title
      }`
    );
    return res;
  } catch (error) {
    console.error('Error fetching products:', error);
    return [];
  }
};

const FeaturedProducts = () => {
  const [sanityProducts, setSanityProducts] = useState<Product[]>([]);
  const [isLoading, setIsLoading] = useState<boolean>(true);

  useEffect(() => {
    const fetchData = async () => {
      const products = await getData();
      setSanityProducts(products);
      setIsLoading(false);
    };

    fetchData();
  }, []);
```

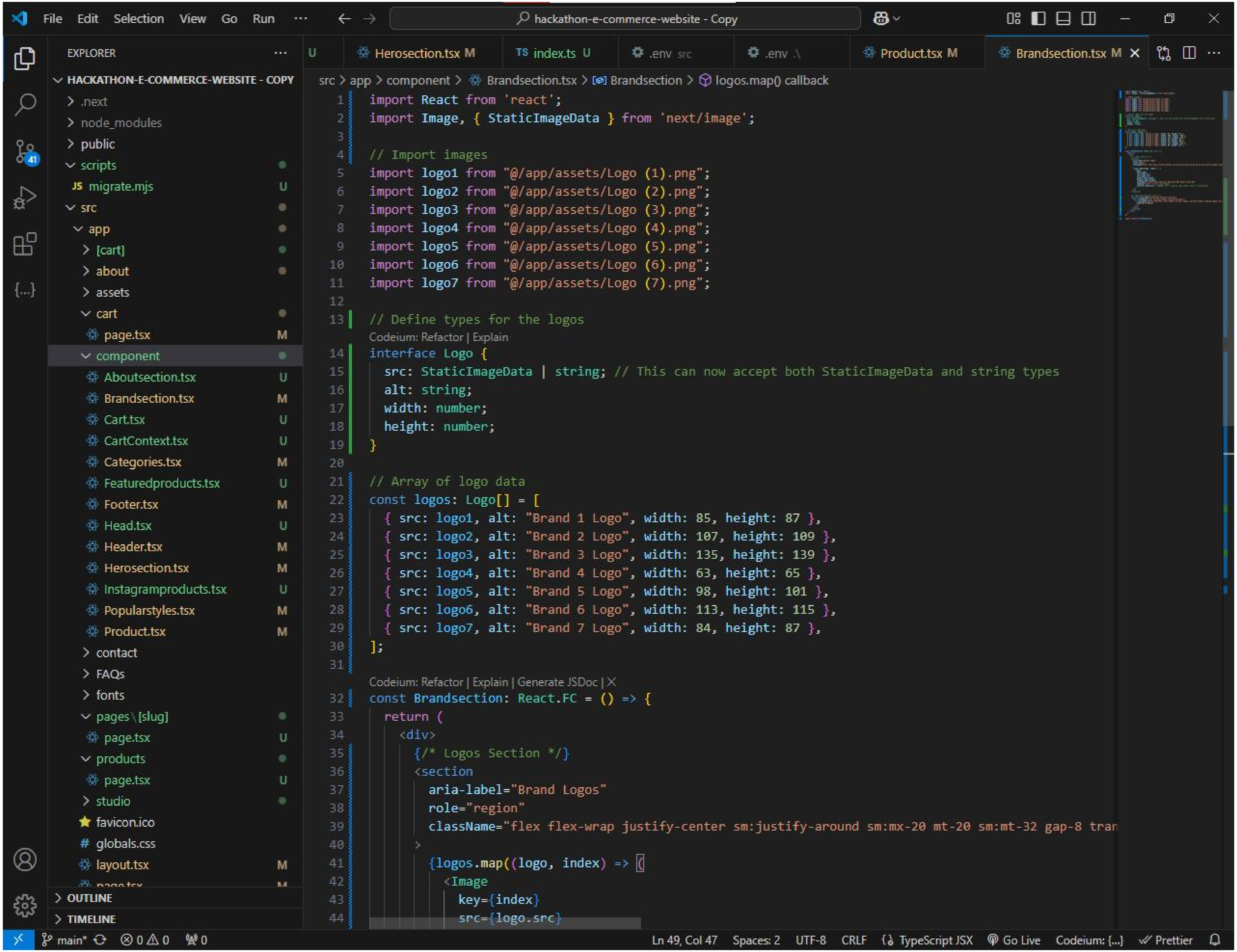


```tsx
'use client';

import React, { useEffect, useState } from 'react';
import { client } from '@/sanity/lib/client';
import Image from 'next/image';
import Pinksofa from '@/app/assets/Image (1).png';
import cart from '@/app/assets/Group (2).png';
import Instagramproducts from '@/app/component/Instagramproducts';
import { useParams, useRouter } from 'next/navigation'; // useRouter for navigation
import { useCart } from '@/app/component/CartContext'; // Import the Cart context

interface Product {
  _id: string;
  title: string;
  price: number;
  imageUrl?: string;
  description?: string;
}

const getData = async (slug: string): Promise<Product | null> => {
  try {
    const res: Product[] = await client.fetch(
      `*[_type == "products" && slug.current == $slug] {
        "imageUrl": image.asset->url,
        _id,
        price,
        title,
        description,
        slug
      }`,
      { slug }
    );
    return res[0] || null;
  } catch (error) {
    console.error('Error fetching product:', error);
    return null;
  }
};

const Page = () => {
  let { slug } = useParams();

  if (Array.isArray(slug)) {
```

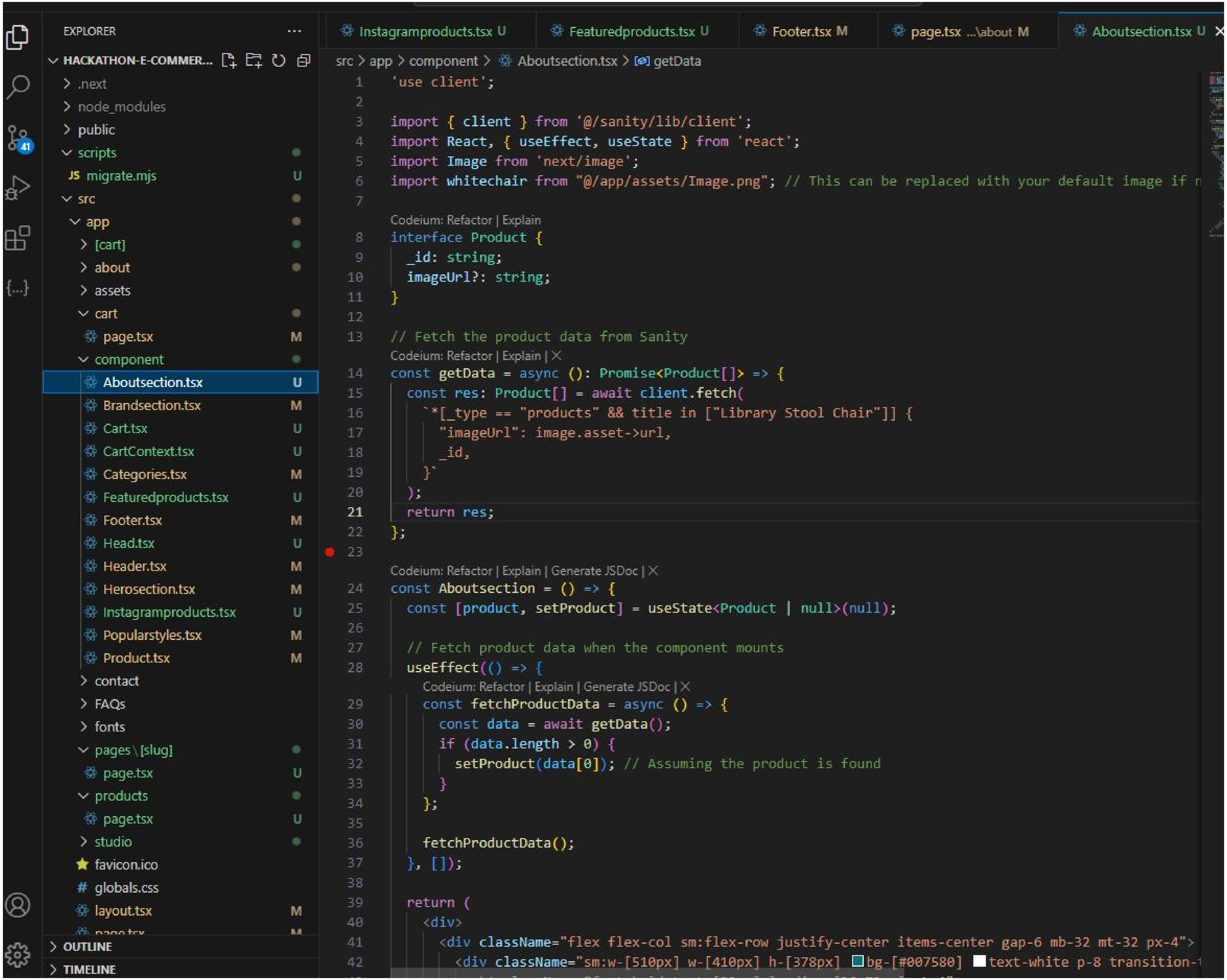


```
> node_modules
> public
v scripts
  JS migrate.mjs                U
v src
  v app
    > [cart]
    > about
    > assets
    v cart
      page.tsx                  M
    v component
      Aboutsection.tsx          U
      Brandsection.tsx          M
      Cart.tsx                  U
      CartContext.tsx           U
      Categories.tsx            M
      Featuredproducts.tsx      U
      Footer.tsx                M
      Head.tsx                  U
      Header.tsx                M
      Herosection.tsx           M
      Instagramproducts.tsx     U
      Popularstyles.tsx         M
      Product.tsx               M
    > contact
    > FAQs
    > fonts
    v pages \ [slug]
      page.tsx                  U
    v products
      page.tsx                  U
    > studio
    favicon.ico
    # globals.css
```

```

import React from 'react';
import { useCart } from '@/app/component/CartContext'; // Ensure Car
import Image from 'next/image';
import Pinksofa from '@/app/assets/Image (1).png';

// Define the type for cart item
Codeium: Refactor | Explain
interface CartItem {
  _id: string;
  title: string;
  price: number;
  quantity: number;
  imageUrl?: string;
}

Codeium: Refactor | Explain | Generate JSDoc | X
const CartPage = () => {
  // Destructure values from the context
  const { cartItems, removeFromCart, clearCart } = useCart();

  return (
    <div className="cart-page ">
      <h1 className="text-center text-3xl font-bold mb-6">Your Cart</

      {/* Cart items section */}
      <div className="cart-items ">
        {cartItems.length === 0 ? (
          <p className="text-center">Your cart is empty.</p>
        ) : (
          cartItems.map((item: CartItem) => (
            <div key={item._id} className="cart-item flex items-cent
              {/* Display product image */}
              <Image
                src={item.imageUrl || Pinksofa} // Fallback image if
                alt={item.title}
                width={100}
                height={100}
                className="w-[100px] h-[100px] object-contain"
              />
              <div>
```



```tsx
import React from 'react';
import Image, { StaticImageData } from 'next/image';

// Import images
import logo1 from "@/app/assets/Logo (1).png";
import logo2 from "@/app/assets/Logo (2).png";
import logo3 from "@/app/assets/Logo (3).png";
import logo4 from "@/app/assets/Logo (4).png";
import logo5 from "@/app/assets/Logo (5).png";
import logo6 from "@/app/assets/Logo (6).png";
import logo7 from "@/app/assets/Logo (7).png";

// Define types for the logos
interface Logo {
  src: StaticImageData | string; // This can now accept both StaticImageData and string types
  alt: string;
  width: number;
  height: number;
}

// Array of logo data
const logos: Logo[] = [
  { src: logo1, alt: "Brand 1 Logo", width: 85, height: 87 },
  { src: logo2, alt: "Brand 2 Logo", width: 107, height: 109 },
  { src: logo3, alt: "Brand 3 Logo", width: 135, height: 139 },
  { src: logo4, alt: "Brand 4 Logo", width: 63, height: 65 },
  { src: logo5, alt: "Brand 5 Logo", width: 98, height: 101 },
  { src: logo6, alt: "Brand 6 Logo", width: 113, height: 115 },
  { src: logo7, alt: "Brand 7 Logo", width: 84, height: 87 },
];

const Brandsection: React.FC = () => {
  return (
    <div>
      {/* Logos Section */}
      <section
        aria-label="Brand Logos"
        role="region"
        className="flex flex-wrap justify-center sm:justify-around sm:mx-20 mt-20 sm:mt-32 gap-8 tran
      >
        {logos.map((logo, index) => (
          <Image
            key={index}
            src={logo.src}
```



```tsx
'use client';

import { client } from '@/sanity/lib/client';
import React, { useEffect, useState } from 'react';
import Image from 'next/image';
import whitechair from "@/app/assets/Image.png"; // This can be replaced with your default image if

interface Product {
  _id: string;
  imageUrl?: string;
}

// Fetch the product data from Sanity
const getData = async (): Promise<Product[]> => {
  const res: Product[] = await client.fetch(
    `*[_type == "products" && title in ["Library Stool Chair"]] {
      "imageUrl": image.asset->url,
      _id,
    }`
  );
  return res;
};

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

  // Fetch product data when the component mounts
  useEffect(() => {
    const fetchProductData = async () => {
      const data = await getData();
      if (data.length > 0) {
        setProduct(data[0]); // Assuming the product is found
      }
    };

    fetchProductData();
  }, []);

  return (
    <div>
      <div className="flex flex-col sm:flex-row justify-center items-center gap-6 mb-32 mt-32 px-4">
        <div className="sm:w-[510px] w-[410px] h-[378px] bg-[#007580] text-white p-8 transition-
```