"Day 3 - API Integration Report - [COMFY]"

ROLL NO: 118068

1)API integration process:

- After migrating data into Sanity, I fetched the data using GROQ queries using helper functions like “fetchFeaturedProducts( )” to fetch products for the featured section, “fetchProductCategories( )” to fetch products for a particular category, and other helper functions to render product data in multiple components.

- For single product pages and single category pages, I use dynamic GROQ queries to match the ID and slug using params and then display the products.

- With the help of QROQ queries, I get the desired data from Sanity using helper functions and render data in components.

2) Adjustments made to schemas:

I don’t really need to change the schema for products and categories as there Is enough data provided in the given API? I just added the slug field in the category for dynamic fetching.

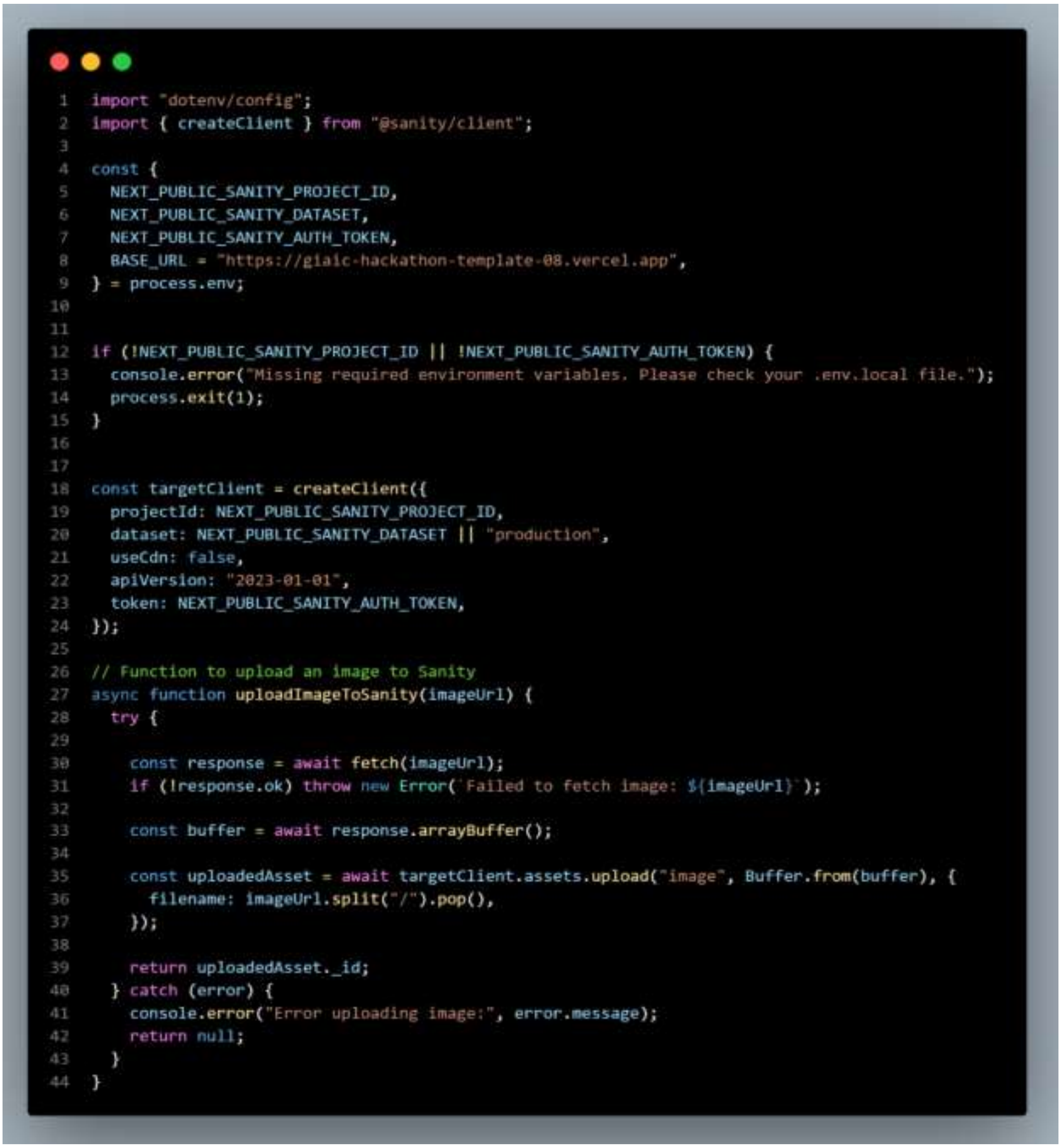
3) Migration steps and tools used:

- Create a .env.local file in the root of project.

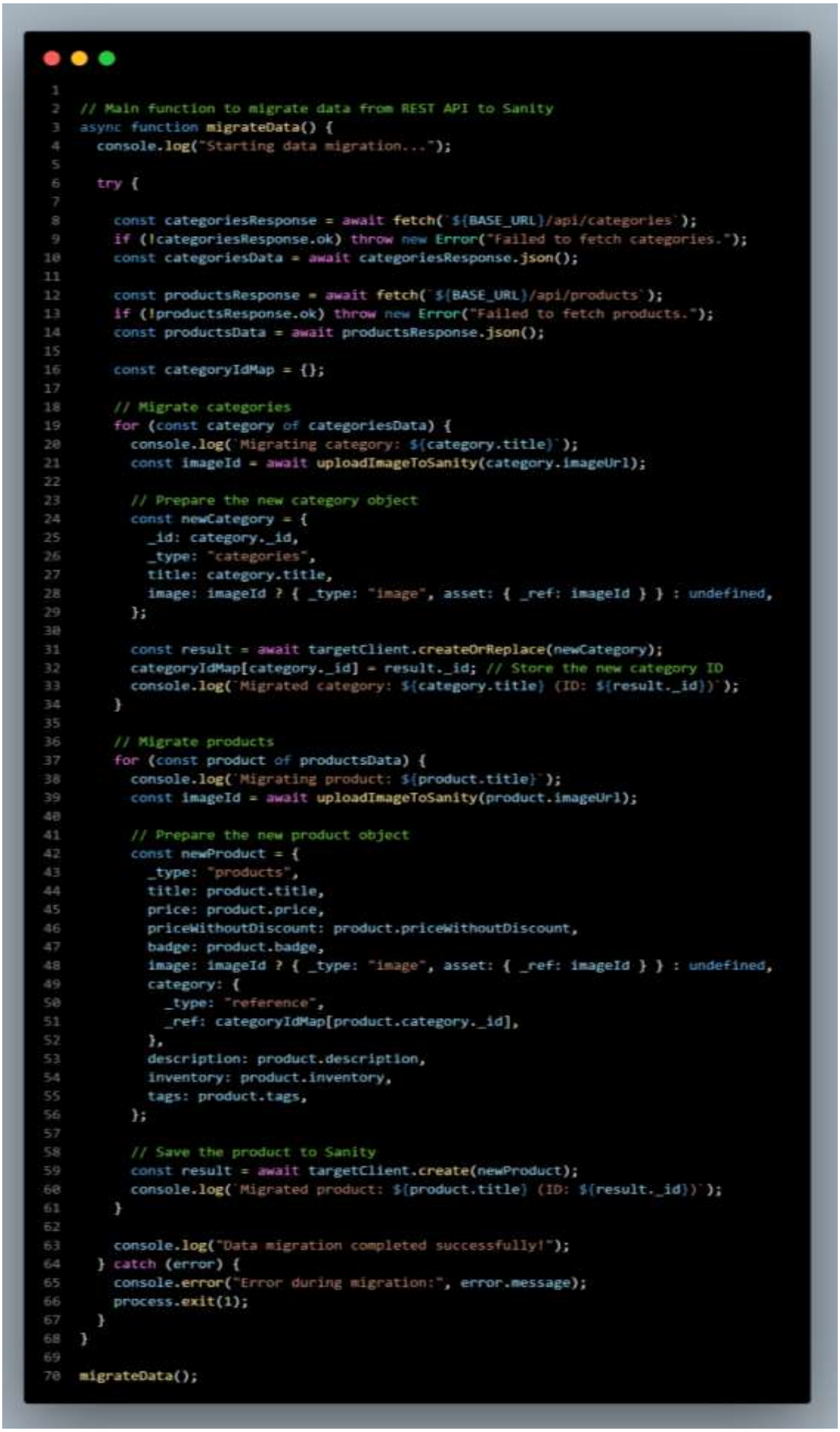
- Get token from sanity and add in .env file with project id and project dataset.

- Create scripts folder in the root of project and inside it create migrate.mjs file.
- In migrate.mjs file write script to transfer data from the provided REST API into Sanity.
- Add “migrate”: ”node scripts/migrate.mjs” inside package..json inside scripts to run the script.
- Install npm install dotenv to load .env into migrate.mjs file.
- Run the command npm run migrate, this insert the data from the rest api to sanity studio.

Migration Script:

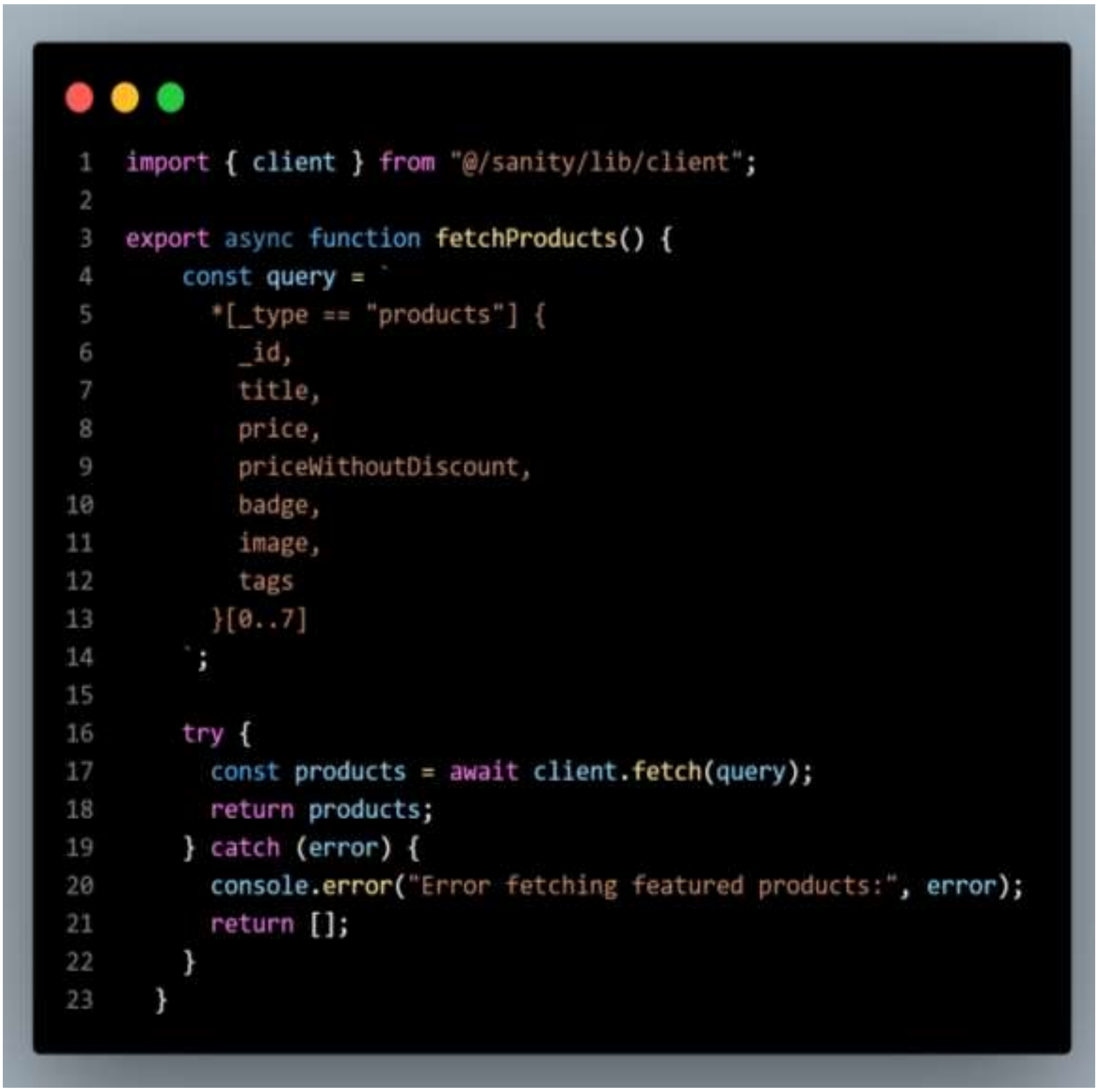


```
import "dotenv/config";
import { createClient } from "@sanity/client";

const {
  NEXT_PUBLIC_SANITY_PROJECT_ID,
  NEXT_PUBLIC_SANITY_DATASET,
  NEXT_PUBLIC_SANITY_AUTH_TOKEN,
  BASE_URL = "https://giaic-hackathon-template-08.vercel.app",
} = process.env;


if (!NEXT_PUBLIC_SANITY_PROJECT_ID || !NEXT_PUBLIC_SANITY_AUTH_TOKEN) {
  console.error("Missing required environment variables. Please check your .env.local file.");
  process.exit(1);
}


const targetClient = createClient({
  projectId: NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset: NEXT_PUBLIC_SANITY_DATASET || "production",
  useCdn: false,
  apiVersion: "2023-01-01",
  token: NEXT_PUBLIC_SANITY_AUTH_TOKEN,
});

// Function to upload an image to Sanity
async function uploadImageToSanity(imageUrl) {
  try {

    const response = await fetch(imageUrl);
    if (!response.ok) throw new Error(`Failed to fetch image: ${imageUrl}`);

    const buffer = await response.arrayBuffer();

    const uploadedAsset = await targetClient.assets.upload("image", Buffer.from(buffer), {
      filename: imageUrl.split("/").pop(),
    });

    return uploadedAsset._id;
  } catch (error) {
    console.error("Error uploading image:", error.message);
    return null;
  }
}
```

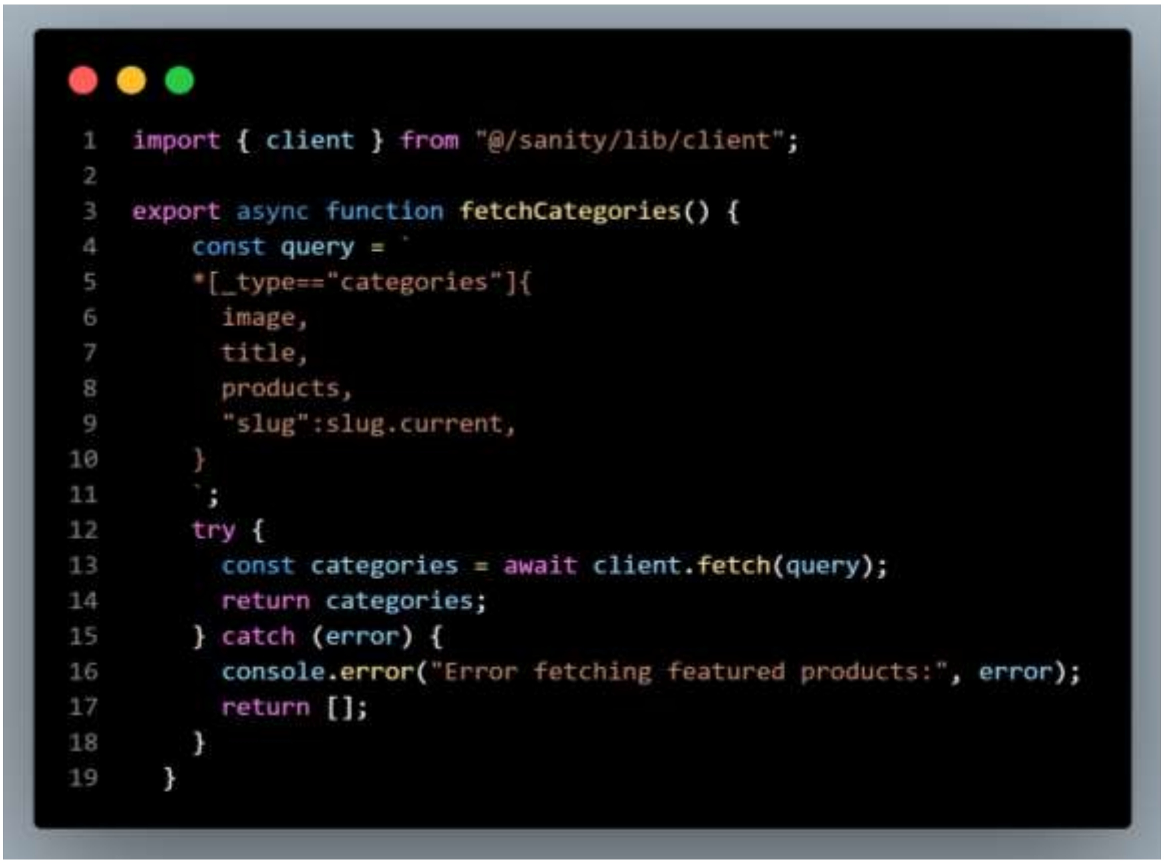


```javascript
// Main function to migrate data from REST API to Sanity
async function migrateData() {
  console.log("Starting data migration...");

  try {

    const categoriesResponse = await fetch(`${BASE_URL}/api/categories`);
    if (!categoriesResponse.ok) throw new Error("Failed to fetch categories.");
    const categoriesData = await categoriesResponse.json();

    const productsResponse = await fetch(`${BASE_URL}/api/products`);
    if (!productsResponse.ok) throw new Error("Failed to fetch products.");
    const productsData = await productsResponse.json();

    const categoryIdMap = {};

    // Migrate categories
    for (const category of categoriesData) {
      console.log(`Migrating category: ${category.title}`);
      const imageId = await uploadImageToSanity(category.imageUrl);

      // Prepare the new category object
      const newCategory = {
        _id: category._id,
        _type: "categories",
        title: category.title,
        image: imageId ? { _type: "image", asset: { _ref: imageId } } : undefined,
      };

      const result = await targetClient.createOrReplace(newCategory);
      categoryIdMap[category._id] = result._id; // Store the new category ID
      console.log(`Migrated category: ${category.title} (ID: ${result._id})`);
    }

    // Migrate products
    for (const product of productsData) {
      console.log(`Migrating product: ${product.title}`);
      const imageId = await uploadImageToSanity(product.imageUrl);

      // Prepare the new product object
      const newProduct = {
        _type: "products",
        title: product.title,
        price: product.price,
        priceWithoutDiscount: product.priceWithoutDiscount,
        badge: product.badge,
        image: imageId ? { _type: "image", asset: { _ref: imageId } } : undefined,
        category: {
          _type: "reference",
          _ref: categoryIdMap[product.category._id],
        },
        description: product.description,
        inventory: product.inventory,
        tags: product.tags,
      };

      // Save the product to Sanity
      const result = await targetClient.create(newProduct);
      console.log(`Migrated product: ${product.title} (ID: ${result._id})`);
    }

    console.log("Data migration completed successfully!");
  } catch (error) {
    console.error("Error during migration:", error.message);
    process.exit(1);
  }
}

migrateData();
```

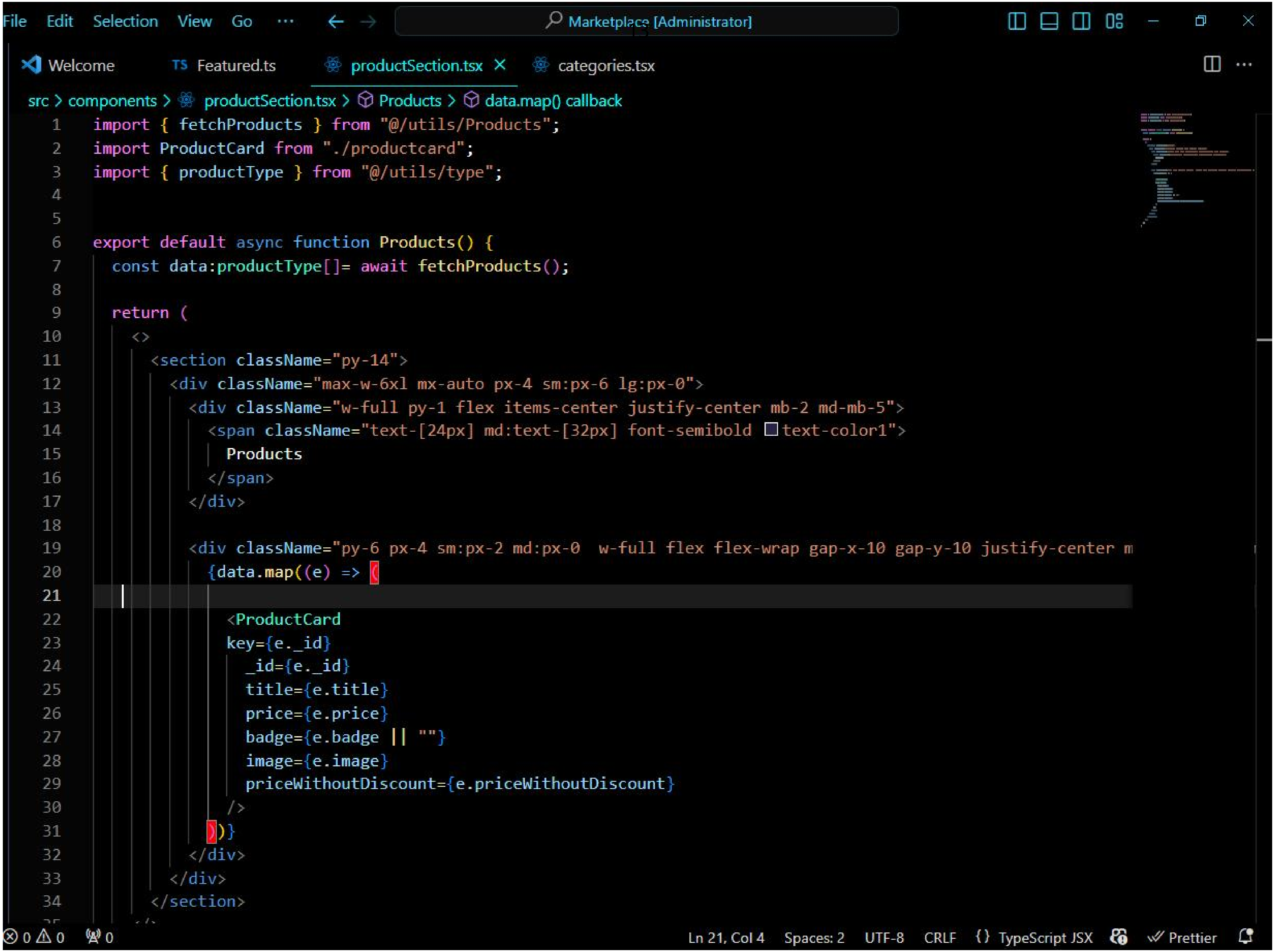
API Integration:



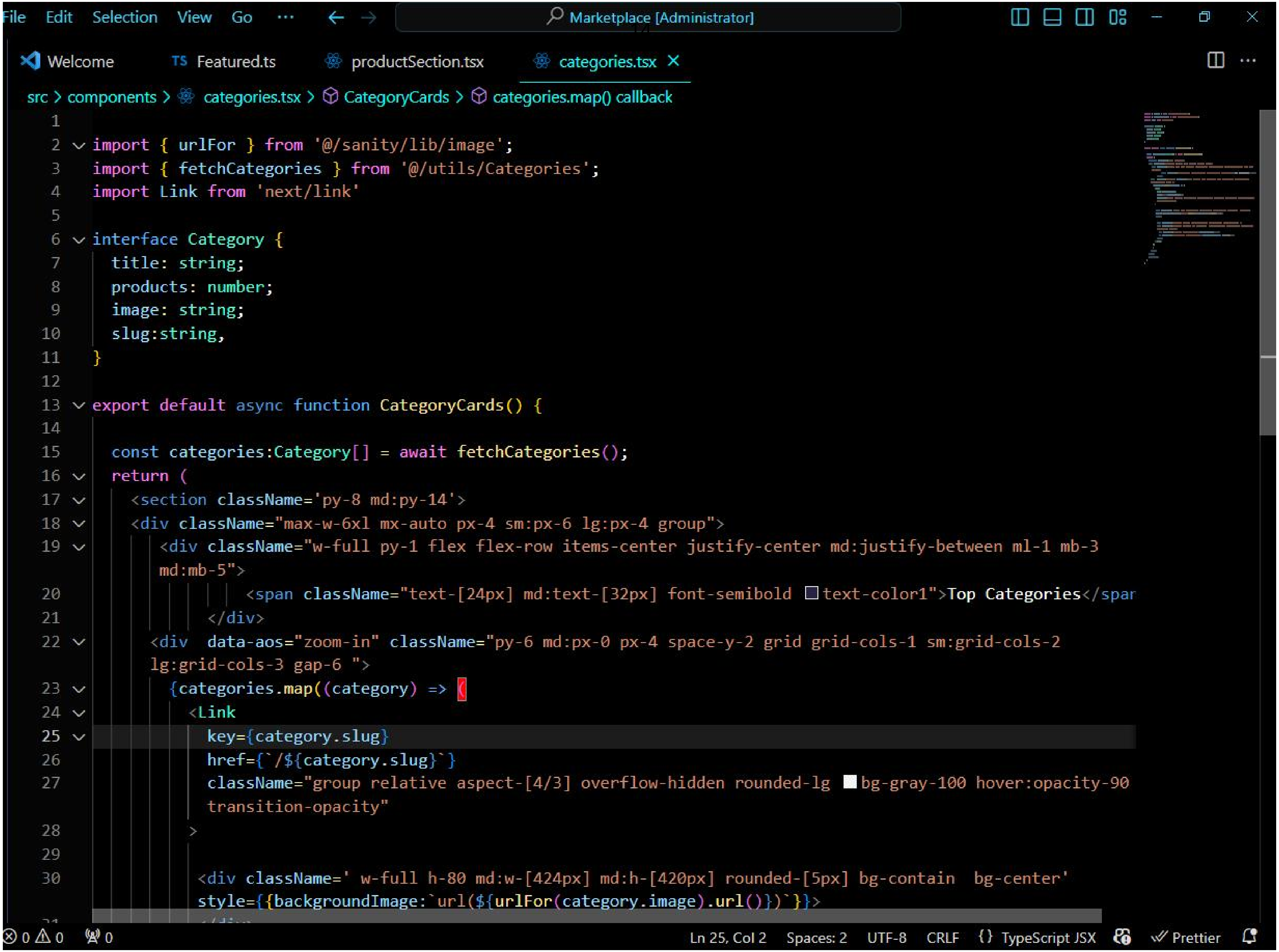
```
import { client } from "@/sanity/lib/client";

export async function fetchProducts() {
    const query = `
      *[_type == "products"] {
        _id,
        title,
        price,
        priceWithoutDiscount,
        badge,
        image,
        tags
      }[0..7]
    `;

    try {
      const products = await client.fetch(query);
      return products;
    } catch (error) {
      console.error("Error fetching featured products:", error);
      return [];
    }
  }
```



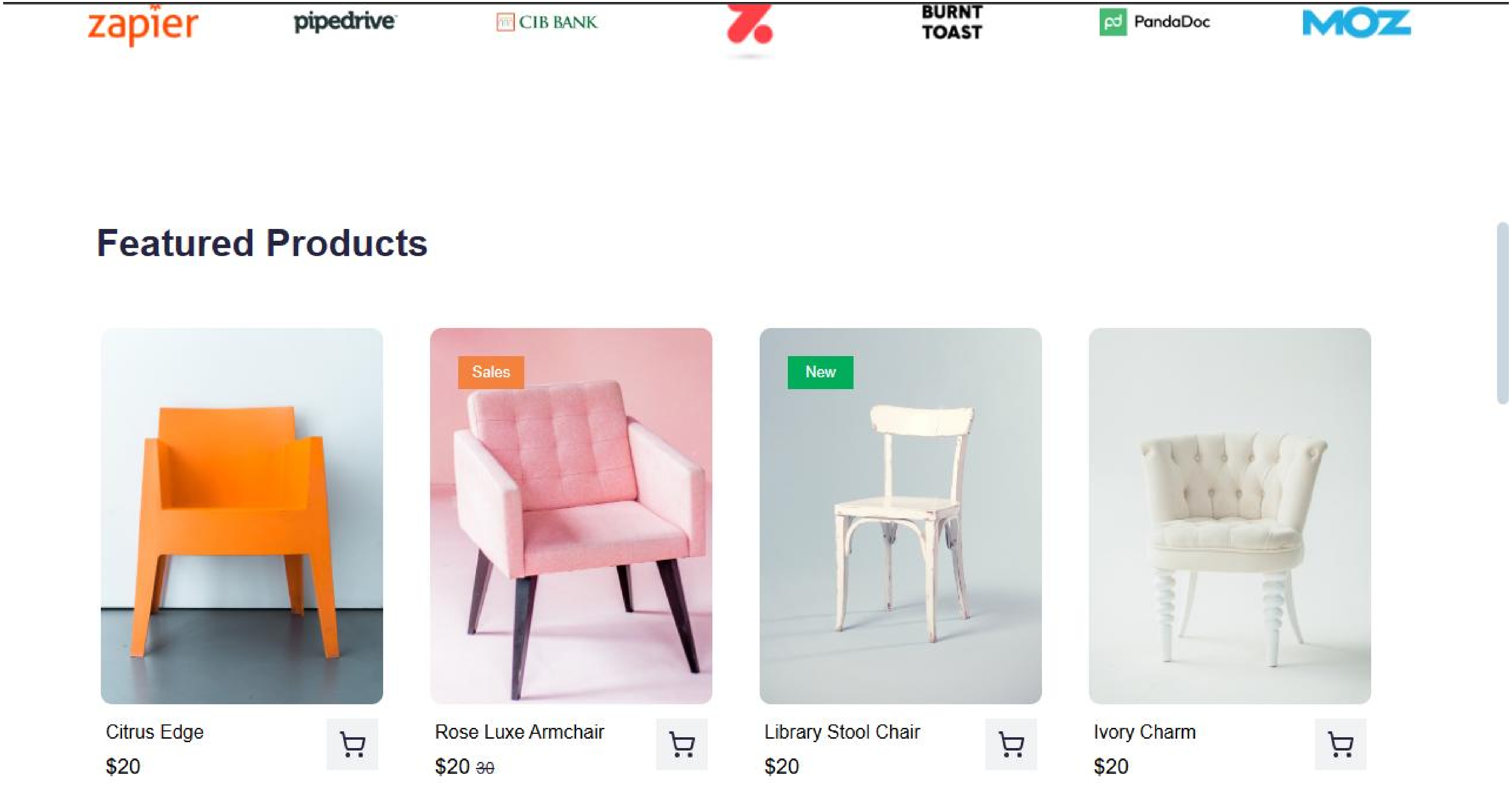
```
import { client } from "@/sanity/lib/client";

export async function fetchCategories() {
    const query = `
    *[_type=="categories"]{
      image,
      title,
      products,
      "slug":slug.current,
    }
    `;
    try {
      const categories = await client.fetch(query);
      return categories;
    } catch (error) {
      console.error("Error fetching featured products:", error);
      return [];
    }
  }
```



```tsx
import { fetchProducts } from "@/utils/Products";
import ProductCard from "./productcard";
import { productType } from "@/utils/type";


export default async function Products() {
  const data:productType[]= await fetchProducts();

  return (
    <>
      <section className="py-14">
        <div className="max-w-6xl mx-auto px-4 sm:px-6 lg:px-0">
          <div className="w-full py-1 flex items-center justify-center mb-2 md-mb-5">
            <span className="text-[24px] md:text-[32px] font-semibold text-color1">
              Products
            </span>
          </div>

          <div className="py-6 px-4 sm:px-2 md:px-0  w-full flex flex-wrap gap-x-10 gap-y-10 justify-center m
            {data.map((e) => (

              <ProductCard
              key={e._id}
                _id={e._id}
                title={e.title}
                price={e.price}
                badge={e.badge || ""}
                image={e.image}
                priceWithoutDiscount={e.priceWithoutDiscount}
              />
            ))}
          </div>
        </div>
      </section>
```



```

import { urlFor } from '@/sanity/lib/image';
import { fetchCategories } from '@/utils/Categories';
import Link from 'next/link'

interface Category {
  title: string;
  products: number;
  image: string;
  slug:string,
}

export default async function CategoryCards() {

  const categories:Category[] = await fetchCategories();
  return (
    <section className='py-8 md:py-14'>
    <div className="max-w-6xl mx-auto px-4 sm:px-6 lg:px-4 group">
      <div className="w-full py-1 flex flex-row items-center justify-center md:justify-between ml-1 mb-3 md:mb-5">
            <span className="text-[24px] md:text-[32px] font-semibold text-color1">Top Categories</span
          </div>
      <div  data-aos="zoom-in" className="py-6 md:px-0 px-4 space-y-2 grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-6 ">
        {categories.map((category) => (
          <Link
            key={category.slug}
            href={`/${category.slug}`}
            className="group relative aspect-[4/3] overflow-hidden rounded-lg bg-gray-100 hover:opacity-90 transition-opacity"
          >

           <div className=' w-full h-80 md:w-[424px] md:h-[420px] rounded-[5px] bg-contain  bg-center'
           style={{backgroundImage:`url(${urlFor(category.image).url()})`}}>
           </div>
```



Featured Products

Citrus Edge

$20

Sales

Rose Luxe Armchair

$20 ~~30~~

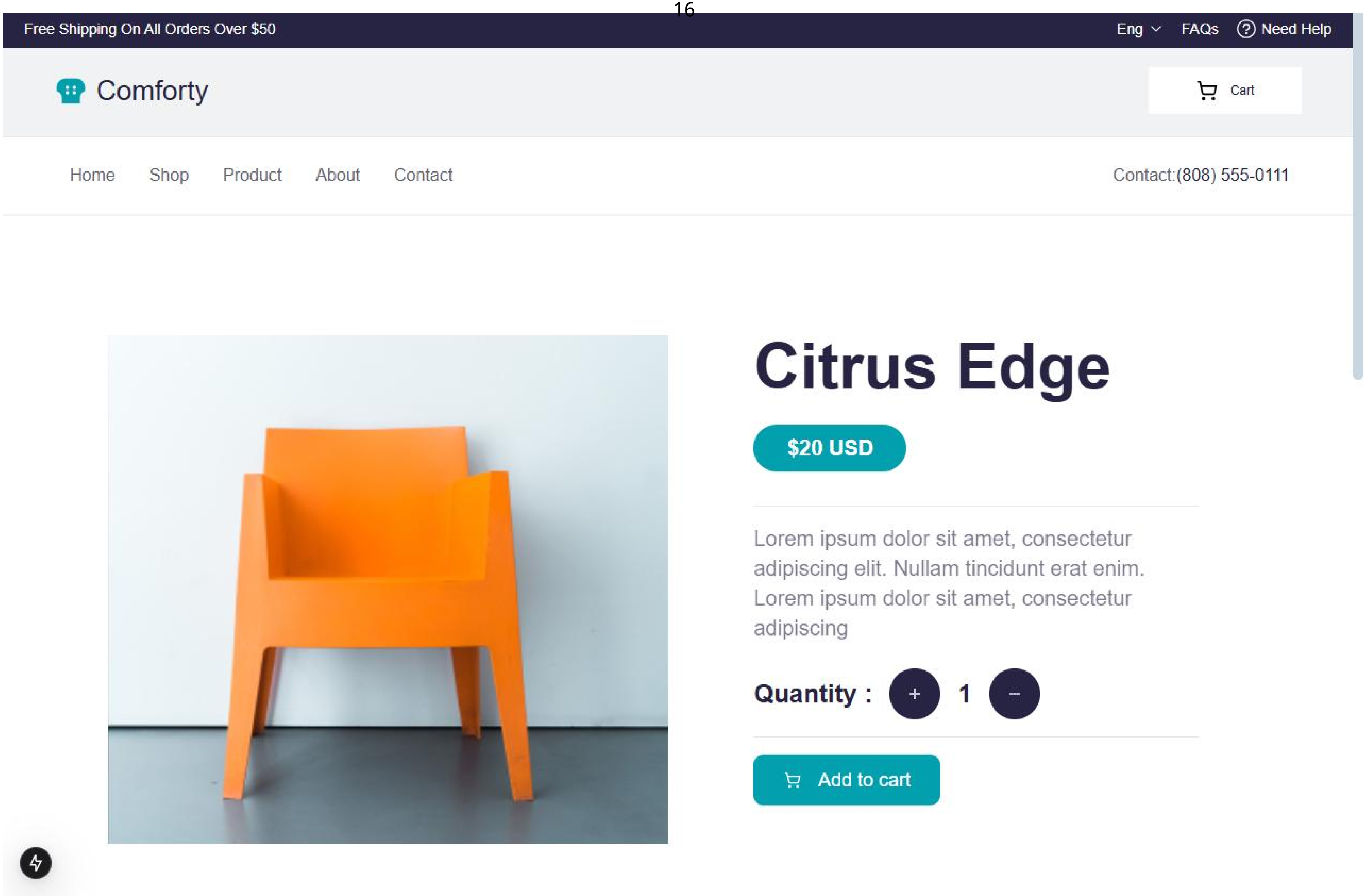
New

Library Stool Chair

$20

Ivory Charm

$20

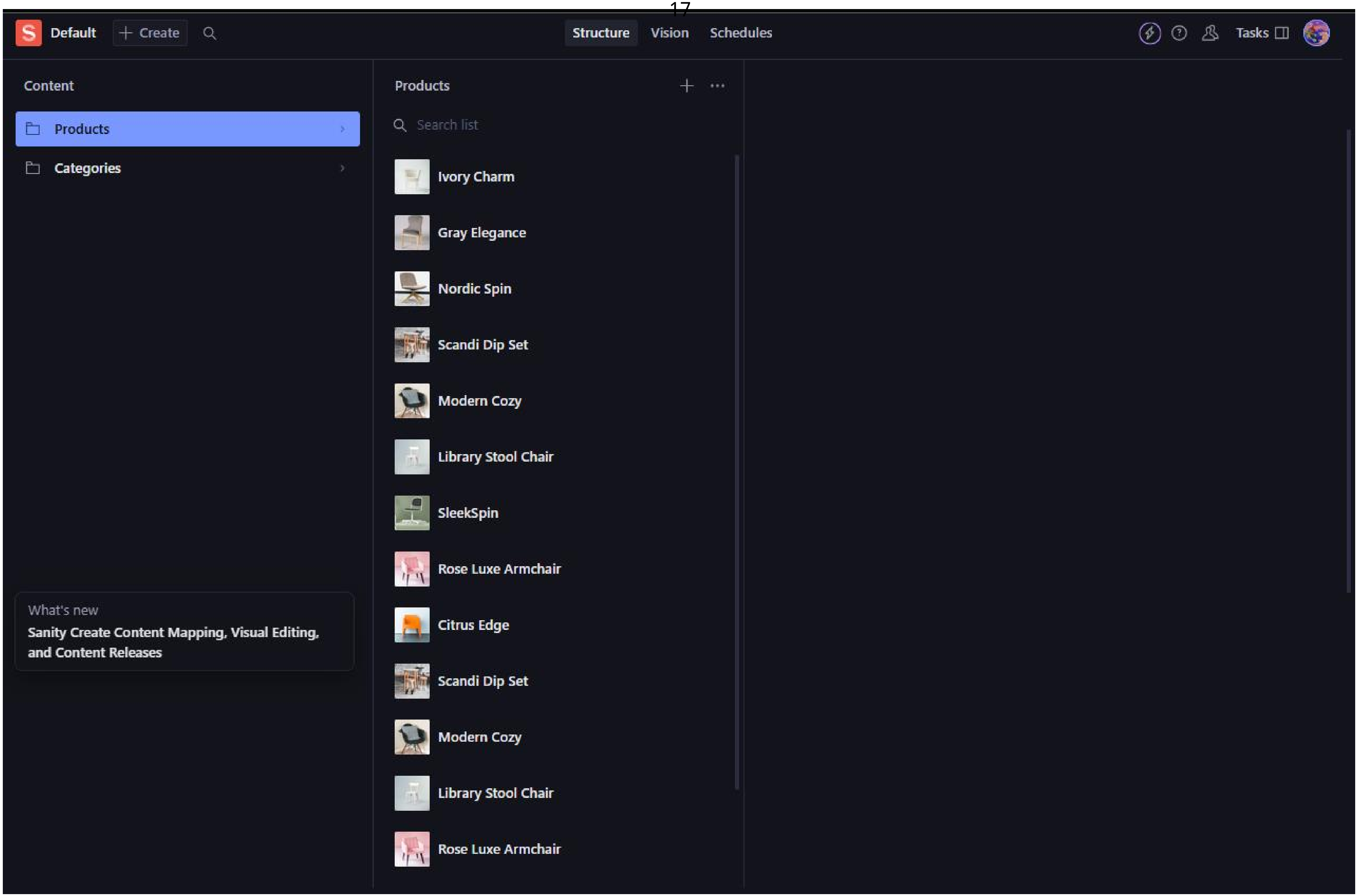


Citrus Edge

$20 USD

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam tincidunt erat enim. Lorem ipsum dolor sit amet, consectetur adipiscing

Quantity : + 1 –



Ivory Charm

Gray Elegance

Nordic Spin

Scandi Dip Set

Modern Cozy

Library Stool Chair

SleekSpin

Rose Luxe Armchair

Citrus Edge

Scandi Dip Set

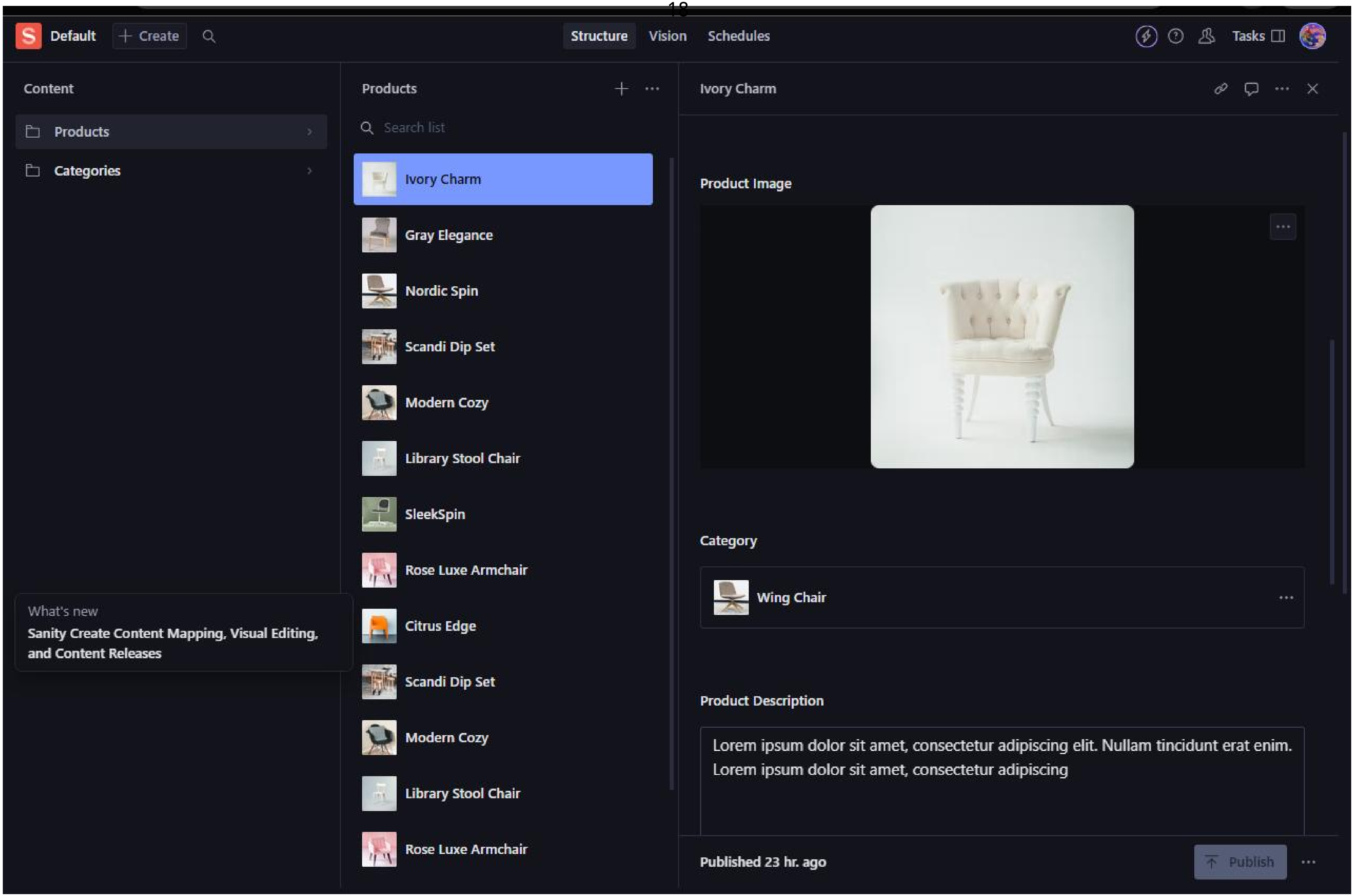
Modern Cozy

Library Stool Chair

Rose Luxe Armchair

What's new

Sanity Create Content Mapping, Visual Editing, and Content Releases



What's new

Sanity Create Content Mapping, Visual Editing, and Content Releases

Ivory Charm

Product Image

Category: Wing Chair

Product Description

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam tincidunt erat enim. Lorem ipsum dolor sit amet, consectetur adipiscing

Published 23 hr. ago