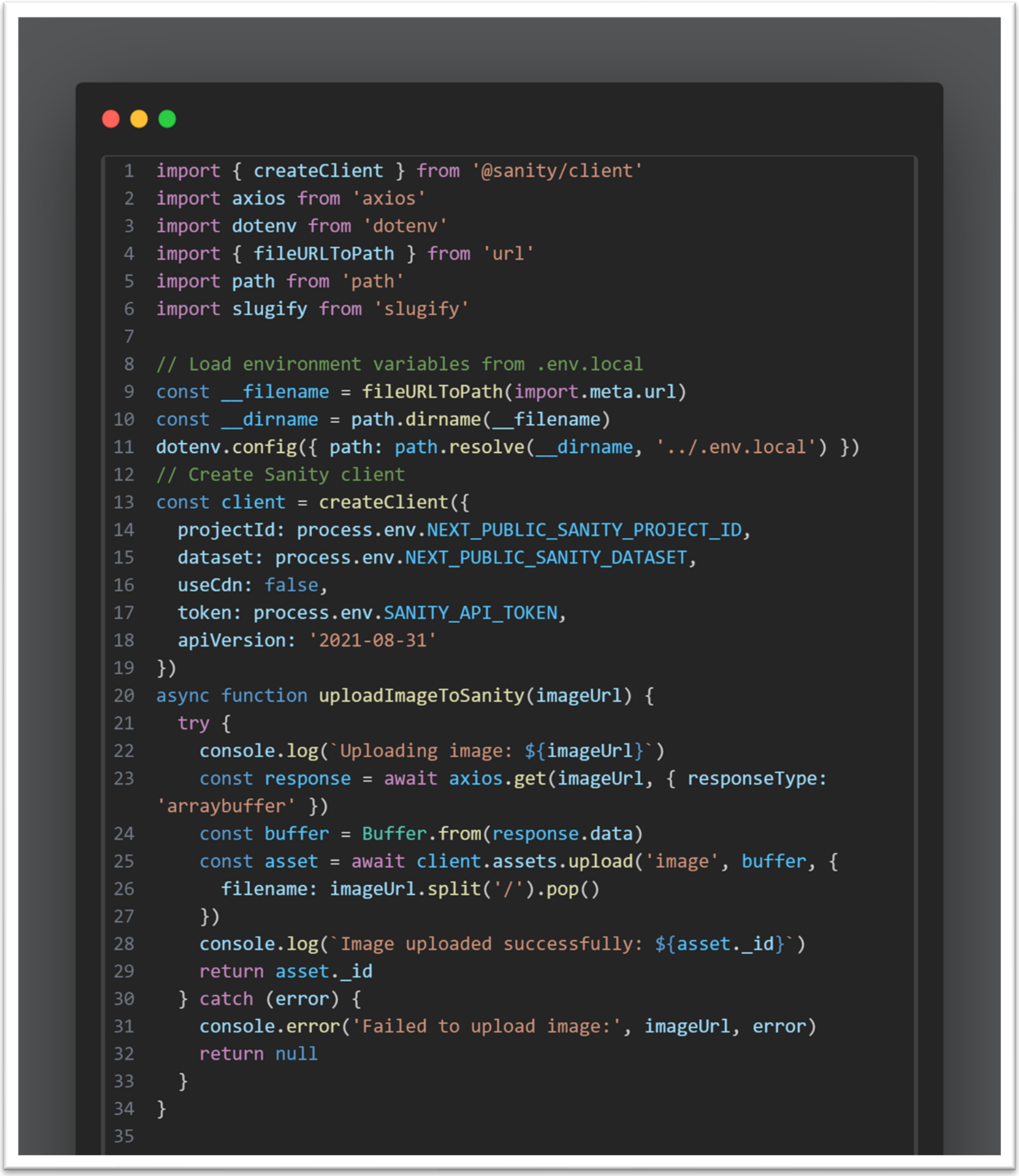
API Integration Report

1.Fetching Data from an API and Storing It in Sanity

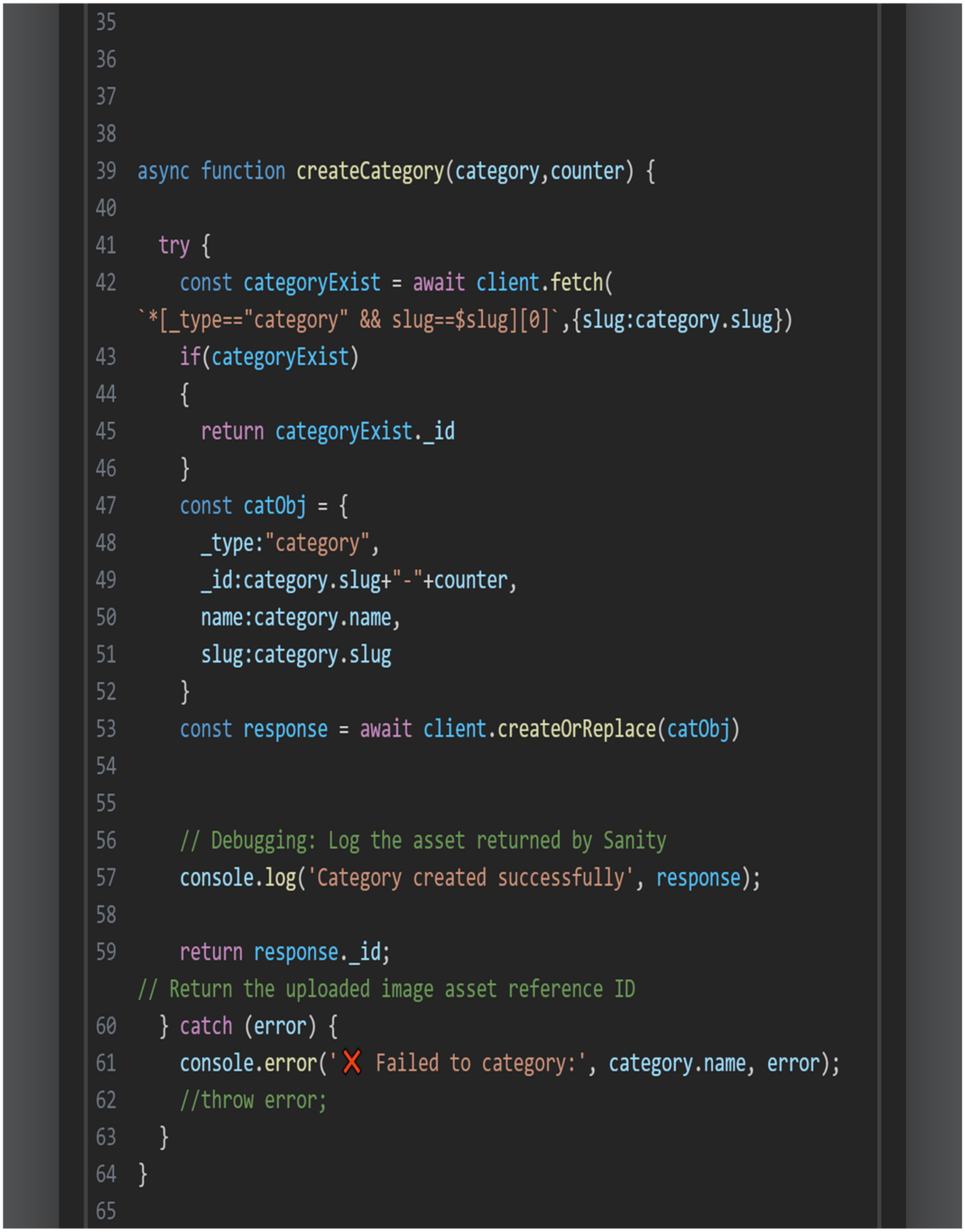
Start with importing necessary libraries such as @sanity/client, axios, and dotenv.

The importData() function:

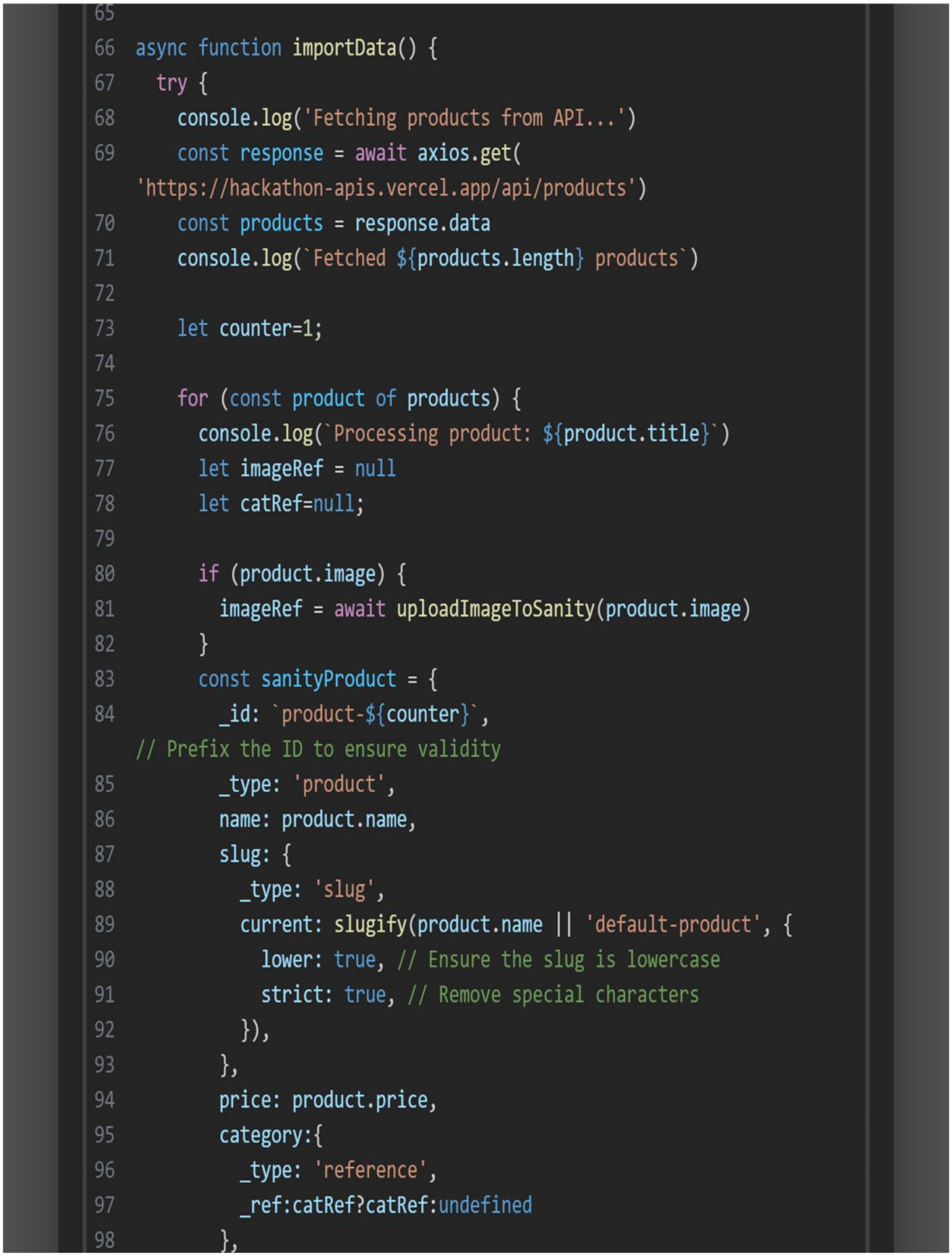
- Fetches the product data from an external API: https://hackathon-apis.vercel.app/api/products.
- Loops through the products and processes each one.
- If a product has an image, the uploadImageToSanity() function uploads the image to Sanity and gets a reference for it.
- A new product object is created with details such as name, price, description, features, dimensions, etc.
- The product is uploaded to your Sanity database using the Sanity client.



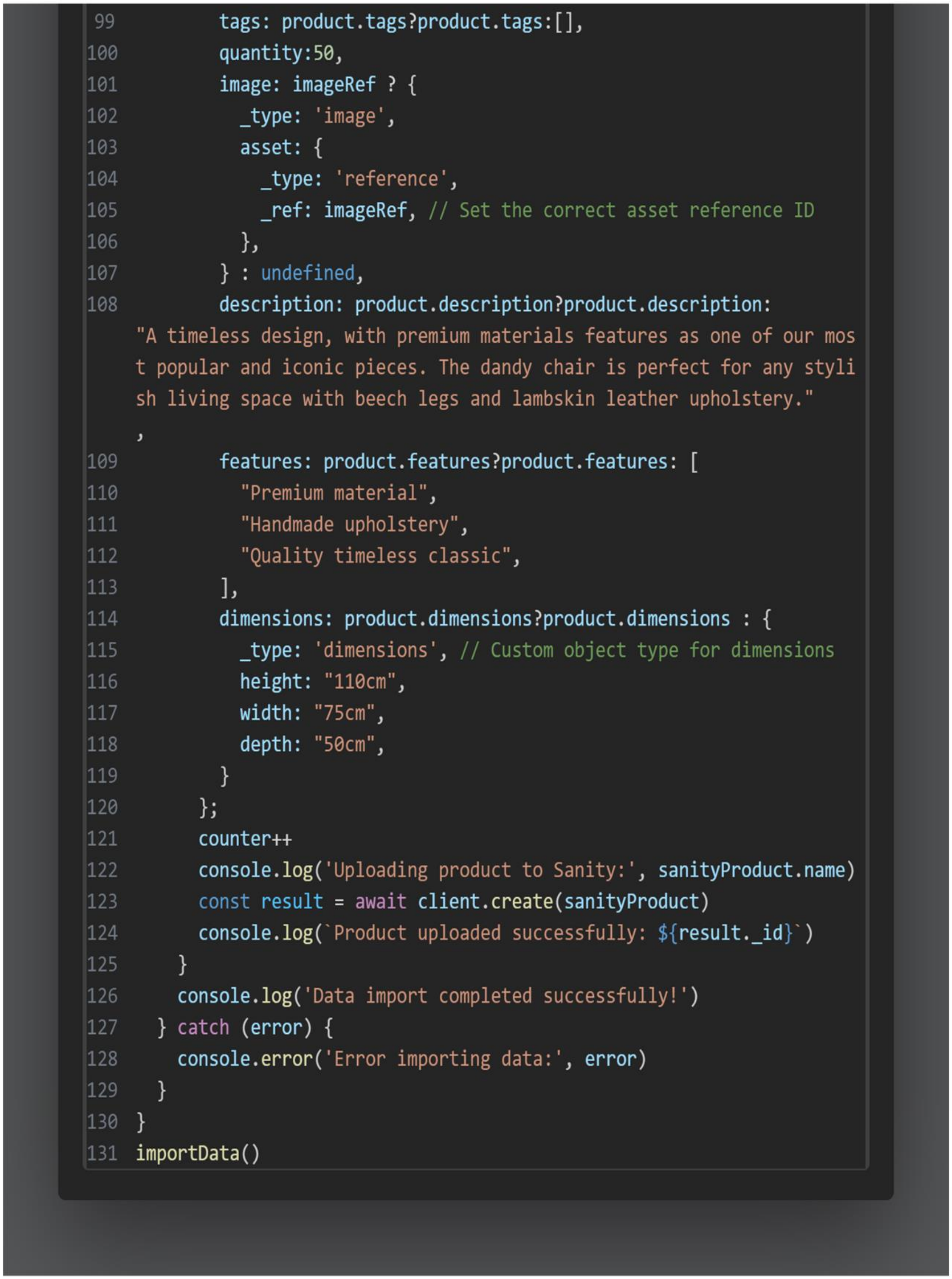
```
import { createClient } from '@sanity/client'
import axios from 'axios'
import dotenv from 'dotenv'
import { fileURLToPath } from 'url'
import path from 'path'
import slugify from 'slugify'

// Load environment variables from .env.local
const __filename = fileURLToPath(import.meta.url)
const __dirname = path.dirname(__filename)
dotenv.config({ path: path.resolve(__dirname, '../.env.local') })
// Create Sanity client
const client = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
  useCdn: false,
  token: process.env.SANITY_API_TOKEN,
  apiVersion: '2021-08-31'
})
async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading image: ${imageUrl}`)
    const response = await axios.get(imageUrl, { responseType: 'arraybuffer' })
    const buffer = Buffer.from(response.data)
    const asset = await client.assets.upload('image', buffer, {
      filename: imageUrl.split('/').pop()
    })
    console.log(`Image uploaded successfully: ${asset._id}`)
    return asset._id
  } catch (error) {
    console.error('Failed to upload image:', imageUrl, error)
    return null
  }
}

```



```
async function createCategory(category,counter) {

  try {
    const categoryExist = await client.fetch(
`*[_type=="category" && slug==$slug][0]`,{slug:category.slug})
    if(categoryExist)
    {
      return categoryExist._id
    }
    const catObj = {
      _type:"category",
      _id:category.slug+"-"+counter,
      name:category.name,
      slug:category.slug
    }
    const response = await client.createOrReplace(catObj)


    // Debugging: Log the asset returned by Sanity
    console.log('Category created successfully', response);

    return response._id;
// Return the uploaded image asset reference ID
  } catch (error) {
    console.error('❌ Failed to category:', category.name, error);
    //throw error;
  }
}

```



```
async function importData() {
  try {
    console.log('Fetching products from API...')
    const response = await axios.get(
'https://hackathon-apis.vercel.app/api/products')
    const products = response.data
    console.log(`Fetched ${products.length} products`)

    let counter=1;

    for (const product of products) {
      console.log(`Processing product: ${product.title}`)
      let imageRef = null
      let catRef=null;

      if (product.image) {
        imageRef = await uploadImageToSanity(product.image)
      }
      const sanityProduct = {
        _id: `product-${counter}`,
// Prefix the ID to ensure validity
        _type: 'product',
        name: product.name,
        slug: {
          _type: 'slug',
          current: slugify(product.name || 'default-product', {
            lower: true, // Ensure the slug is lowercase
            strict: true, // Remove special characters
          }),
        },
        price: product.price,
        category:{
          _type: 'reference',
          _ref:catRef?catRef:undefined
        },
```

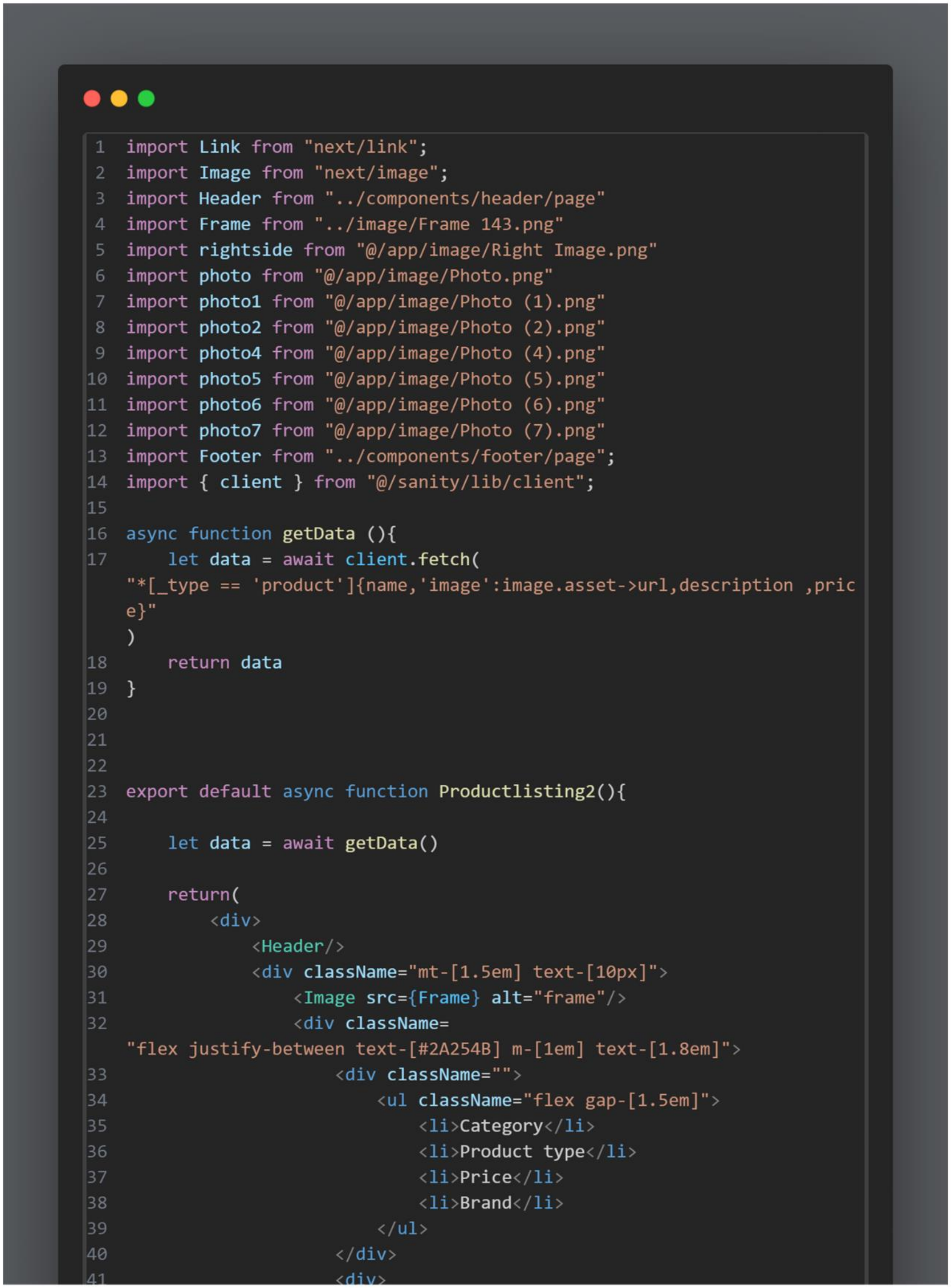


```
        tags: product.tags?product.tags:[],
        quantity:50,
        image: imageRef ? {
          _type: 'image',
          asset: {
            _type: 'reference',
            _ref: imageRef, // Set the correct asset reference ID
          },
        } : undefined,
        description: product.description?product.description: "A timeless design, with premium materials features as one of our most popular and iconic pieces. The dandy chair is perfect for any stylish living space with beech legs and lambskin leather upholstery.",
        features: product.features?product.features: [
          "Premium material",
          "Handmade upholstery",
          "Quality timeless classic",
        ],
        dimensions: product.dimensions?product.dimensions : {
          _type: 'dimensions', // Custom object type for dimensions
          height: "110cm",
          width: "75cm",
          depth: "50cm",
        }
      };
      counter++
      console.log('Uploading product to Sanity:', sanityProduct.name)
      const result = await client.create(sanityProduct)
      console.log(`Product uploaded successfully: ${result._id}`)
    }
    console.log('Data import completed successfully!')
  } catch (error) {
    console.error('Error importing data:', error)
  }
}
importData()
```

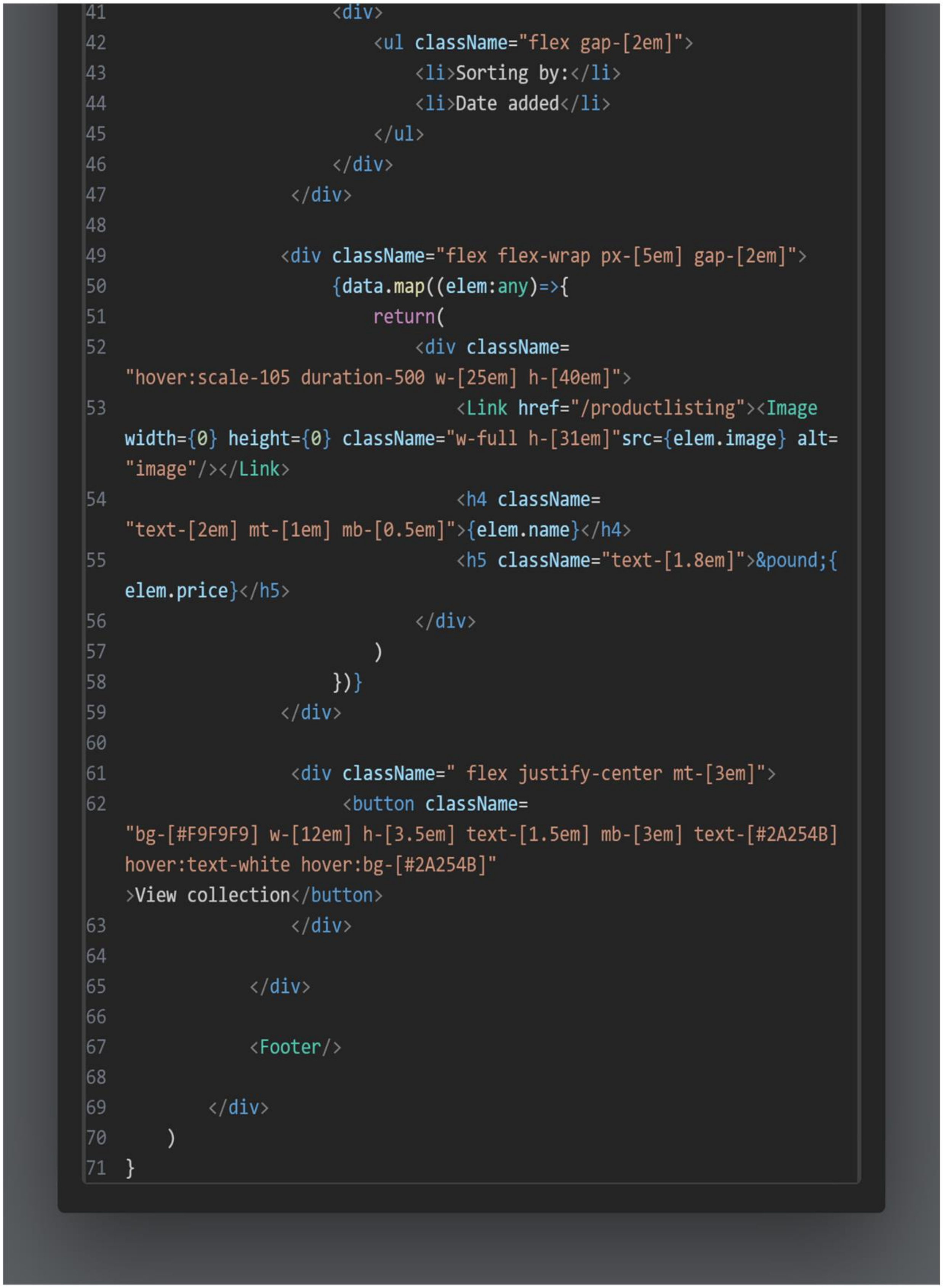
2.Fetching Data from Sanity for UI

In the **getData() function**:

- Data is fetched from the Sanity database using a query to retrieve product details like name, image, description, and price.
- This data is passed to the UI component.



```
import Link from "next/link";
import Image from "next/image";
import Header from "../components/header/page"
import Frame from "../image/Frame 143.png"
import rightside from "@/app/image/Right Image.png"
import photo from "@/app/image/Photo.png"
import photo1 from "@/app/image/Photo (1).png"
import photo2 from "@/app/image/Photo (2).png"
import photo4 from "@/app/image/Photo (4).png"
import photo5 from "@/app/image/Photo (5).png"
import photo6 from "@/app/image/Photo (6).png"
import photo7 from "@/app/image/Photo (7).png"
import Footer from "../components/footer/page";
import { client } from "@/sanity/lib/client";

async function getData (){
    let data = await client.fetch(
"*[_type == 'product']{name,'image':image.asset->url,description ,price}"
)
    return data
}



export default async function Productlisting2(){

    let data = await getData()

    return(
        <div>
            <Header/>
            <div className="mt-[1.5em] text-[10px]">
                <Image src={Frame} alt="frame"/>
                <div className=
"flex justify-between text-[#2A254B] m-[1em] text-[1.8em]">
                    <div className="">
                        <ul className="flex gap-[1.5em]">
                            <li>Category</li>
                            <li>Product type</li>
                            <li>Price</li>
                            <li>Brand</li>
                        </ul>
                    </div>
                    <div>
```



```
                    <div>
                        <ul className="flex gap-[2em]">
                            <li>Sorting by:</li>
                            <li>Date added</li>
                        </ul>
                    </div>
                </div>

               <div className="flex flex-wrap px-[5em] gap-[2em]">
                    {data.map((elem:any)=>{
                        return(
                            <div className="hover:scale-105 duration-500 w-[25em] h-[40em]">
                                <Link href="/productlisting"><Image width={0} height={0} className="w-full h-[31em]"src={elem.image} alt="image"/></Link>
                                <h4 className="text-[2em] mt-[1em] mb-[0.5em]">{elem.name}</h4>
                                <h5 className="text-[1.8em]">&pound;{elem.price}</h5>
                            </div>
                        )
                    })}
               </div>

                <div className=" flex justify-center mt-[3em]">
                     <button className="bg-[#F9F9F9] w-[12em] h-[3.5em] text-[1.5em] mb-[3em] text-[#2A254B] hover:text-white hover:bg-[#2A254B]">View collection</button>
                </div>

            </div>

            <Footer/>

        </div>
    )
}
```

3.Displaying Data in the Frontend

ProductListing2 component:

- It calls the getData() function to fetch products from Sanity.
- It maps through the fetched product data and displays:
- Product images (with links).
- Product names.
- Product prices.
- It includes options for sorting/filtering like Category, Product Type, Price, and Brand.

4.Flow of Data

1.API to Sanity:

- External API → Process data → Upload to Sanity.

2.Sanity to UI:

- Fetch product data from Sanity → Display on the website.