Step-by-Step Guide: Using Sanity API in Your Project

1. Introduction

Sanity is a headless CMS that provides powerful APIs for managing and querying
walks you through the process of setting up and using the Sanity API in your proj

---

2. Importing the Sanity API

Step 1: Install the Sanity Client

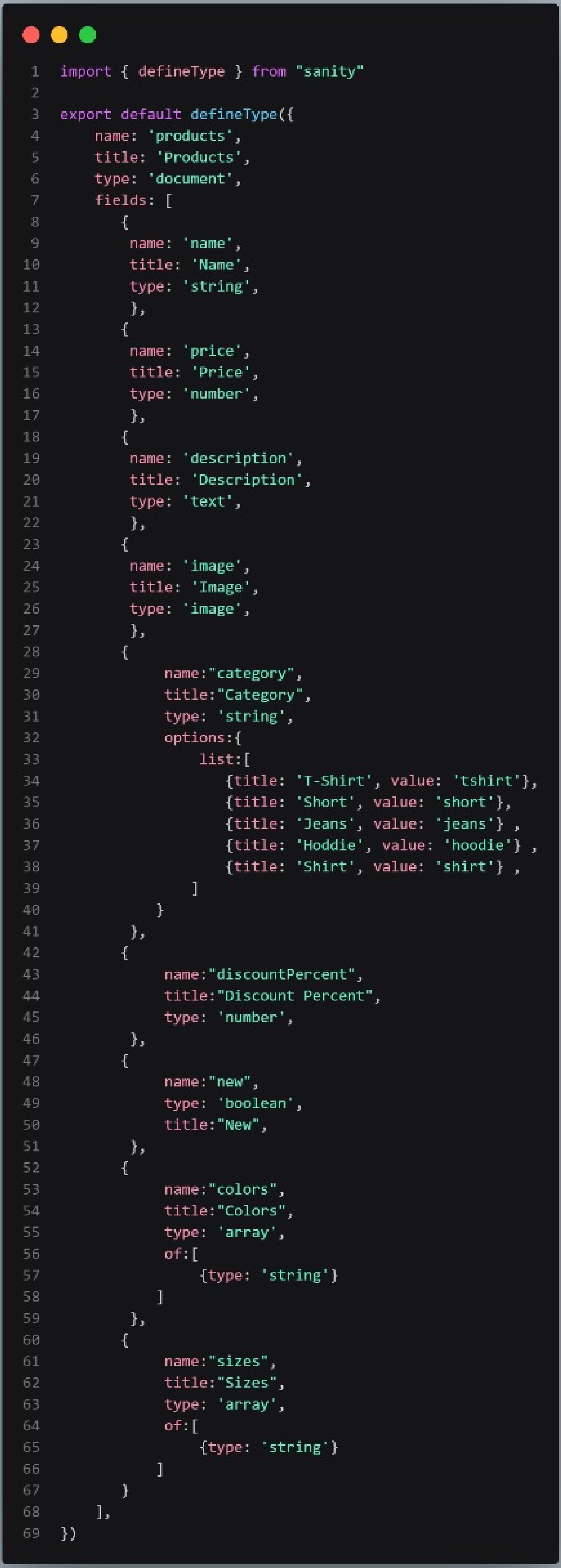
To start using the Sanity API, you need to install the Sanity client in your project.
application to communicate with your Sanity project.

Step 2: Configure the Sanity Client

Once installed, configure the client with your project details. These include your
name, API version, and whether to use the CDN for faster performance.

Step 3: Test the Connection

Verify the setup by testing the connection to your Sanity project. This ensures yo
correct and the client can successfully connect to the API.



```
import { defineType } from "sanity"

export default defineType({
    name: 'products',
    title: 'Products',
    type: 'document',
    fields: [
       {
        name: 'name',
        title: 'Name',
        type: 'string',
        },
       {
        name: 'price',
        title: 'Price',
        type: 'number',
        },
       {
        name: 'description',
        title: 'Description',
        type: 'text',
        },
       {
        name: 'image',
        title: 'Image',
        type: 'image',
        },
       {
            name:"category",
            title:"Category",
            type: 'string',
            options:{
                list:[
                   {title: 'T-Shirt', value: 'tshirt'},
                   {title: 'Short', value: 'short'},
                   {title: 'Jeans', value: 'jeans'} ,
                   {title: 'Hoddie', value: 'hoodie'} ,
                   {title: 'Shirt', value: 'shirt'} ,
                ]
            }
        },
       {
            name:"discountPercent",
            title:"Discount Percent",
            type: 'number',
        },
       {
            name:"new",
            type: 'boolean',
            title:"New",
        },
       {
            name:"colors",
            title:"Colors",
            type: 'array',
            of:[
                {type: 'string'}
            ]
        },
       {
            name:"sizes",
            title:"Sizes",
            type: 'array',
            of:[
                {type: 'string'}
            ]
       }
    ],
})
```

3. Fetching Data from Sanity

Step 1: Define Your Query

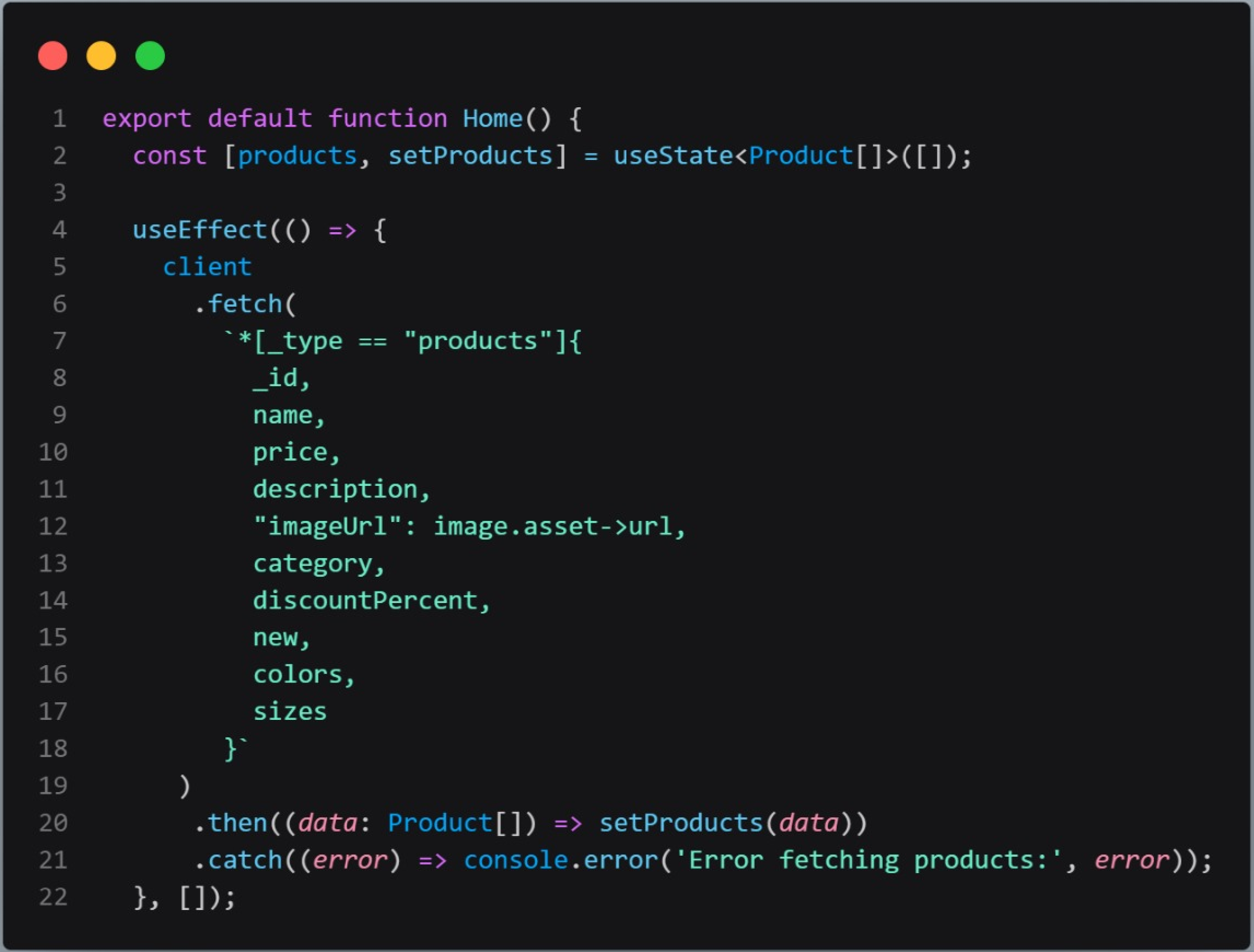
Use GROQ (Graph-Relational Object Queries) to define what data you want to fetch from your Sanity project. This allows you to specify the fields and types of content you need.

Step 2: Fetch Data

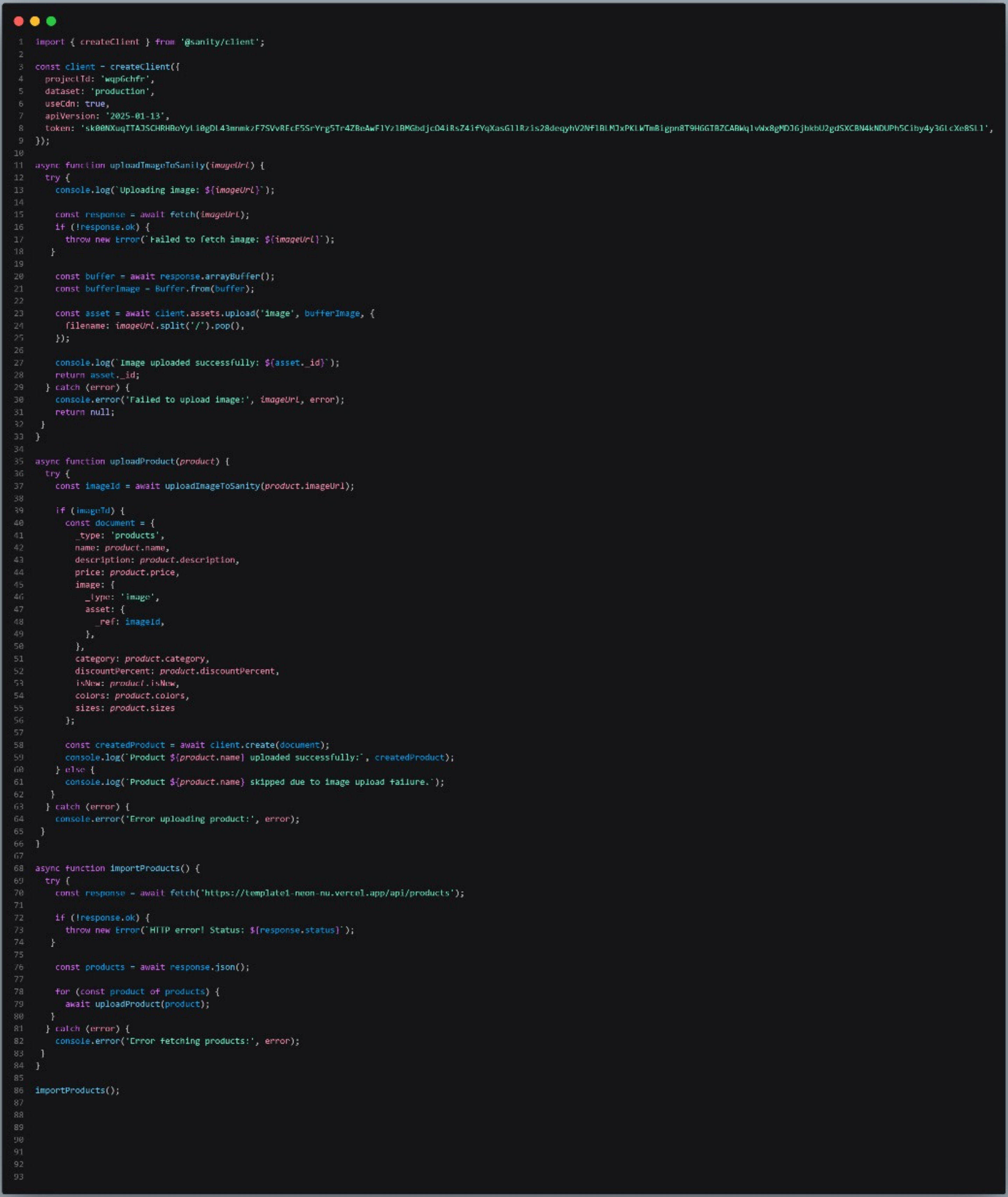
Execute your GROQ query using the Sanity client. Fetching the data involves sending the query and handling the response.

Step 3: Handle the Response

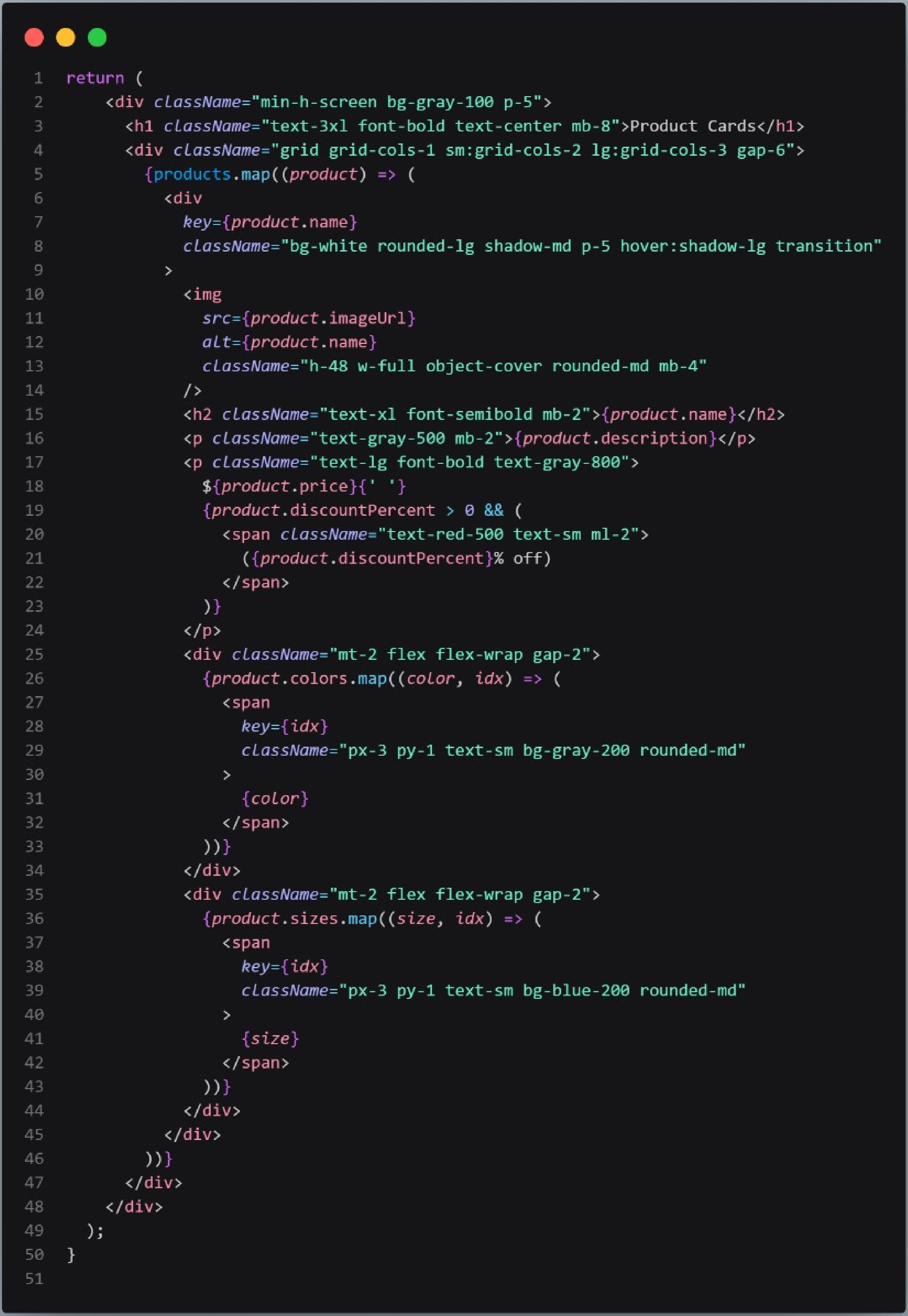
Process the fetched data and prepare it for use in your application. Ensure that the response is correctly formatted and error-handling mechanisms are in place.



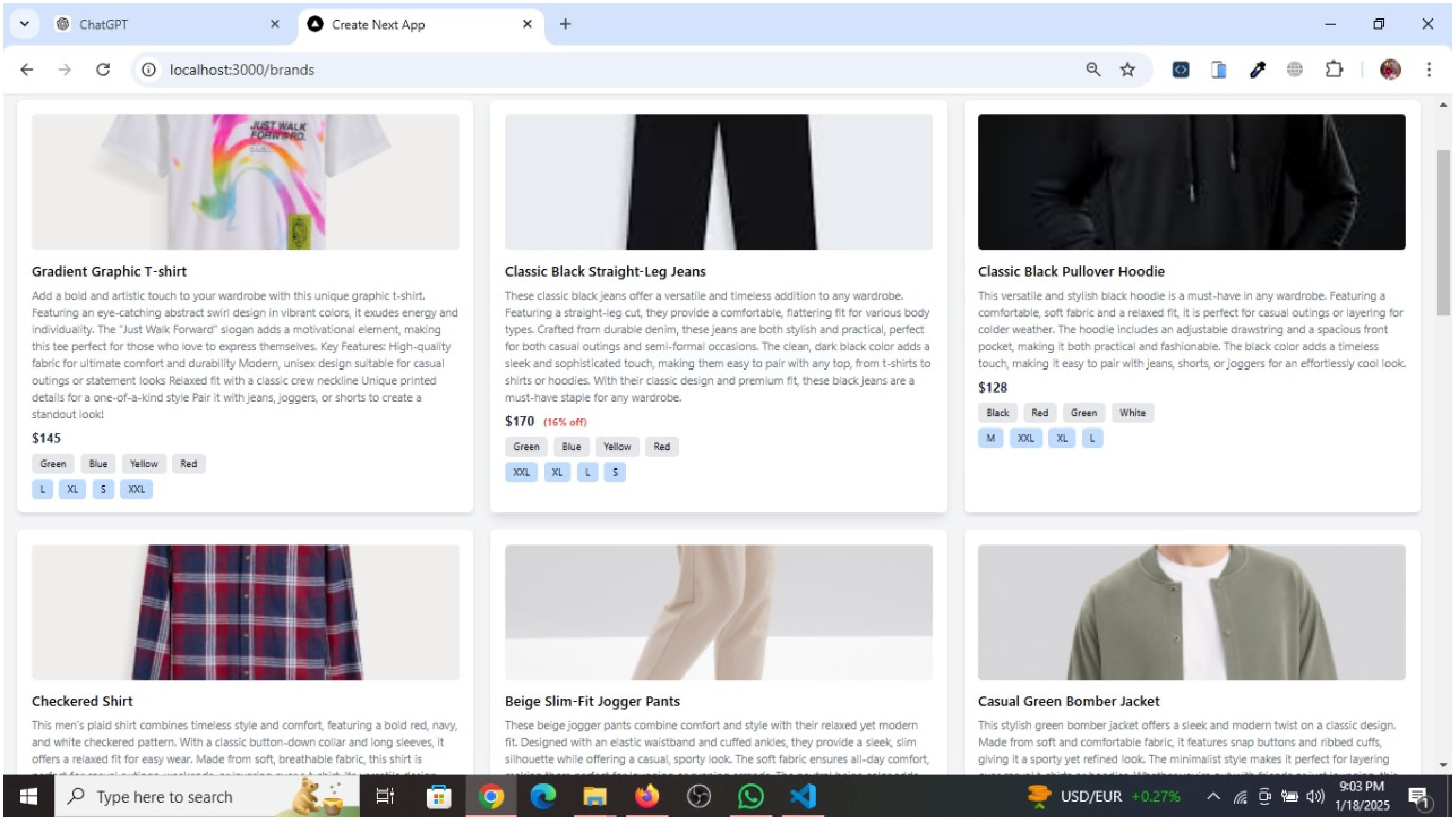
```
export default function Home() {
  const [products, setProducts] = useState<Product[]>([]);

  useEffect(() => {
    client
      .fetch(
        `*[_type == "products"]{
          _id,
          name,
          price,
          description,
          "imageUrl": image.asset->url,
          category,
          discountPercent,
          new,
          colors,
          sizes
        }`
      )
      .then((data: Product[]) => setProducts(data))
      .catch((error) => console.error('Error fetching products:', error));
  }, []);
```



```javascript
import { createClient } from '@sanity/client';

const client = createClient({
  projectId: 'wqp6chfr',
  dataset: 'production',
  useCdn: true,
  apiVersion: '2025-01-13',
  token: 'sk00NXuqITAJSCHRHBoYyLi0gDL43mnmkzF7SVvREcE5SrYrg5Tr4ZBeAwFlYzlBMGbdjcO4iRsZ4ifYqXasGllRzis28deqyhV2NflBLMJxPKLWTmBigpn8T9H6GIBZCABWqlvWx8gMDJ6jbkbU2gdSXCBN4kNDUPh5Ciby4y3GLcXe8SLl',
});

async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading image: ${imageUrl}`);

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

    const buffer = await response.arrayBuffer();
    const bufferImage = Buffer.from(buffer);

    const asset = await client.assets.upload('image', bufferImage, {
      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 uploadProduct(product) {
  try {
    const imageId = await uploadImageToSanity(product.imageUrl);

    if (imageId) {
      const document = {
        _type: 'products',
        name: product.name,
        description: product.description,
        price: product.price,
        image: {
          _type: 'image',
          asset: {
            _ref: imageId,
          },
        },
        category: product.category,
        discountPercent: product.discountPercent,
        isNew: product.isNew,
        colors: product.colors,
        sizes: product.sizes
      };

      const createdProduct = await client.create(document);
      console.log(`Product ${product.name} uploaded successfully:`, createdProduct);
    } else {
      console.log(`Product ${product.name} skipped due to image upload failure.`);
    }
  } catch (error) {
    console.error('Error uploading product:', error);
  }
}

async function importProducts() {
  try {
    const response = await fetch('https://template1-neon-nu.vercel.app/api/products');

    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }

    const products = await response.json();

    for (const product of products) {
      await uploadProduct(product);
    }
  } catch (error) {
    console.error('Error fetching products:', error);
  }
}

importProducts();
```



```jsx
return (
    <div className="min-h-screen bg-gray-100 p-5">
      <h1 className="text-3xl font-bold text-center mb-8">Product Cards</h1>
      <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-6">
        {products.map((product) => (
          <div
            key={product.name}
            className="bg-white rounded-lg shadow-md p-5 hover:shadow-lg transition"
          >
            <img
              src={product.imageUrl}
              alt={product.name}
              className="h-48 w-full object-cover rounded-md mb-4"
            />
            <h2 className="text-xl font-semibold mb-2">{product.name}</h2>
            <p className="text-gray-500 mb-2">{product.description}</p>
            <p className="text-lg font-bold text-gray-800">
              ${product.price}{' '}
              {product.discountPercent > 0 && (
                <span className="text-red-500 text-sm ml-2">
                  ({product.discountPercent}% off)
                </span>
              )}
            </p>
            <div className="mt-2 flex flex-wrap gap-2">
              {product.colors.map((color, idx) => (
                <span
                  key={idx}
                  className="px-3 py-1 text-sm bg-gray-200 rounded-md"
                >
                  {color}
                </span>
              ))}
            </div>
            <div className="mt-2 flex flex-wrap gap-2">
              {product.sizes.map((size, idx) => (
                <span
                  key={idx}
                  className="px-3 py-1 text-sm bg-blue-200 rounded-md"
                >
                  {size}
                </span>
              ))}
            </div>
          </div>
        ))}
      </div>
    </div>
  );
}

```



Gradient Graphic T-shirt

Add a bold and artistic touch to your wardrobe with this unique graphic t-shirt. Featuring an eye-catching abstract swirl design in vibrant colors, it exudes energy and individuality. The "Just Walk Forward" slogan adds a motivational element, making this tee perfect for those who love to express themselves. Key Features: High-quality fabric for ultimate comfort and durability Modern, unisex design suitable for casual outings or statement looks Relaxed fit with a classic crew neckline Unique printed details for a one-of-a-kind style Pair it with jeans, joggers, or shorts to create a standout look!

$145

Green Blue Yellow Red

L XL S XXL

Classic Black Straight-Leg Jeans

These classic black jeans offer a versatile and timeless addition to any wardrobe. Featuring a straight-leg cut, they provide a comfortable, flattering fit for various body types. Crafted from durable denim, these jeans are both stylish and practical, perfect for both casual outings and semi-formal occasions. The clean, dark black color adds a sleek and sophisticated touch, making them easy to pair with any top, from t-shirts to shirts or hoodies. With their classic design and premium fit, these black jeans are a must-have staple for any wardrobe.

$170 (16% off)

Green Blue Yellow Red

XXL XL L S

Classic Black Pullover Hoodie

This versatile and stylish black hoodie is a must-have in any wardrobe. Featuring a comfortable, soft fabric and a relaxed fit, it is perfect for casual outings or layering for colder weather. The hoodie includes an adjustable drawstring and a spacious front pocket, making it both practical and fashionable. The black color adds a timeless touch, making it easy to pair with jeans, shorts, or joggers for an effortlessly cool look.

$128

Black Red Green White

M XXL XL L

Checkered Shirt

This men's plaid shirt combines timeless style and comfort, featuring a bold red, navy, and white checkered pattern. With a classic button-down collar and long sleeves, it offers a relaxed fit for easy wear. Made from soft, breathable fabric, this shirt is

Beige Slim-Fit Jogger Pants

These beige jogger pants combine comfort and style with their relaxed yet modern fit. Designed with an elastic waistband and cuffed ankles, they provide a sleek, slim silhouette while offering a casual, sporty look. The soft fabric ensures all-day comfort,

Casual Green Bomber Jacket

This stylish green bomber jacket offers a sleek and modern twist on a classic design. Made from soft and comfortable fabric, it features snap buttons and ribbed cuffs, giving it a sporty yet refined look. The minimalist style makes it perfect for layering