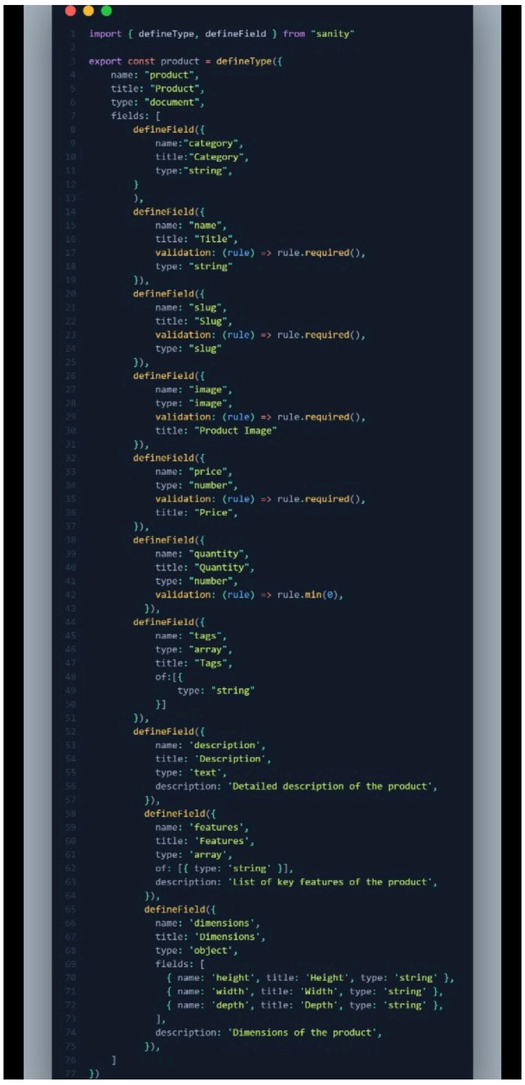
HACKATHON DAY#03 (DOCUMENTATION)

INTRODUCTION:

Our project integrates Sanity CMS as the backend and builds a dynamic frontend using a template. This documentation outlines Day #03's progress.

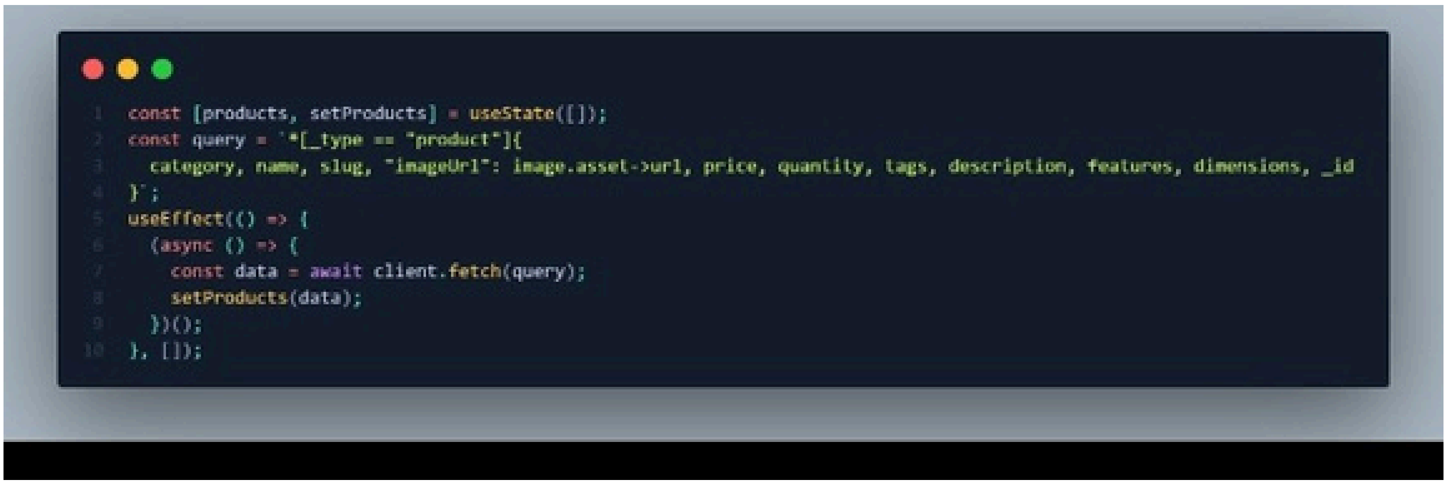
API SET-UP & INTEGRATION:

1. PRODUCTS SCHEMA:



```
import { defineType, defineField } from "sanity"

export const product = defineType({
    name: "product",
    title: "Product",
    type: "document",
    fields: [
        defineField({
            name:"category",
            title:"Category",
            type:"string",
        }
        ),
        defineField({
            name: "name",
            title: "Title",
            validation: (rule) => rule.required(),
            type: "string"
        }),
        defineField({
            name: "slug",
            title: "Slug",
            validation: (rule) => rule.required(),
            type: "slug"
        }),
        defineField({
            name: "image",
            type: "image",
            validation: (rule) => rule.required(),
            title: "Product Image"
        }),
        defineField({
            name: "price",
            type: "number",
            validation: (rule) => rule.required(),
            title: "Price",
        }),
        defineField({
            name: "quantity",
            title: "Quantity",
            type: "number",
            validation: (rule) => rule.min(0),
          }),
        defineField({
            name: "tags",
            type: "array",
            title: "Tags",
            of:[{
                type: "string"
            }]
        }),
        defineField({
            name: 'description',
            title: 'Description',
            type: 'text',
            description: 'Detailed description of the product',
          }),
          defineField({
            name: 'features',
            title: 'Features',
            type: 'array',
            of: [{ type: 'string' }],
            description: 'List of key features of the product',
          }),
          defineField({
            name: 'dimensions',
            title: 'Dimensions',
            type: 'object',
            fields: [
              { name: 'height', title: 'Height', type: 'string' },
              { name: 'width', title: 'Width', type: 'string' },
              { name: 'depth', title: 'Depth', type: 'string' },
            ],
            description: 'Dimensions of the product',
          }),
    ]
})
```

2. FETCH DATA: Implemented data fetching and mapping.



```
const [products, setProducts] = useState([]);
const query = `*[_type == "product"]{
  category, name, slug, "imageUrl": image.asset->url, price, quantity, tags, description, features, dimensions, _id
}`;
useEffect(() => {
  (async () => {
    const data = await client.fetch(query);
    setProducts(data);
  })();
}, []);
```

3. API INTEGRATION : Integrated API with frontend template.

=>MIGRATION:

[illegible]

1. ENVIRONMENT SETUP:

Set up the development environment, including Sanity CMS, API client library, and frontend template.

2. DATA FETCHING:

Implemented data fetching from Sanity CMS API endpoints.

3. _Image Upload_: Configured image upload functionality using Sanity CMS assets.

4. _Documentation Creation_: Created documentation for API endpoint setup, data fetching, and migration steps.

=>FINAL CHECKLIST

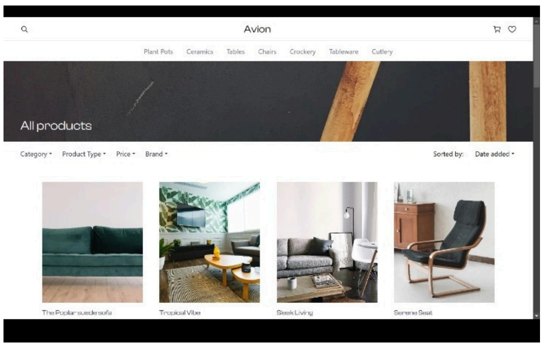
- API endpoint setup ✅

- Data fetching and mapping ✅

- API integration ✅

- Migration script ✅

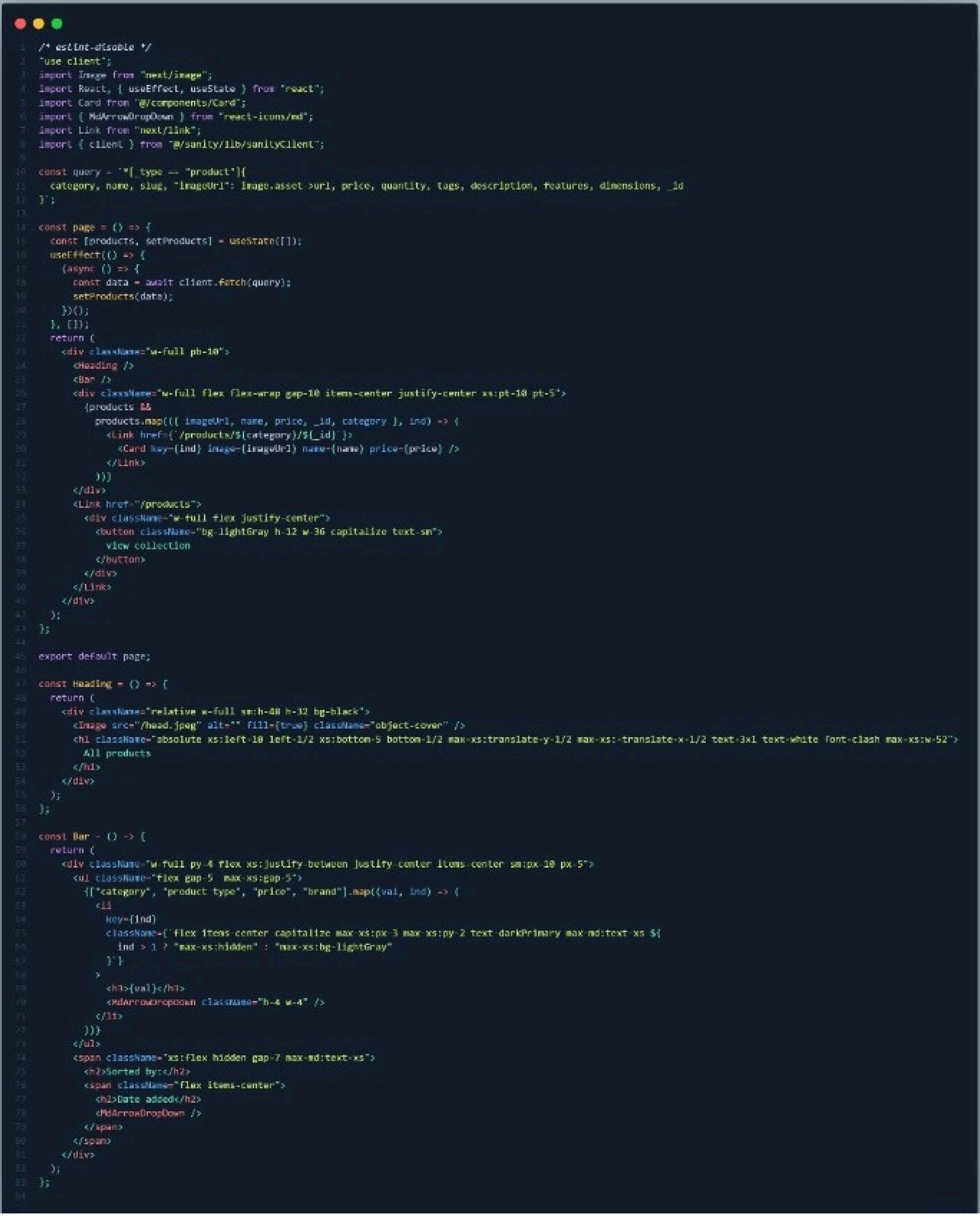
=>PRODUCT PAGE:



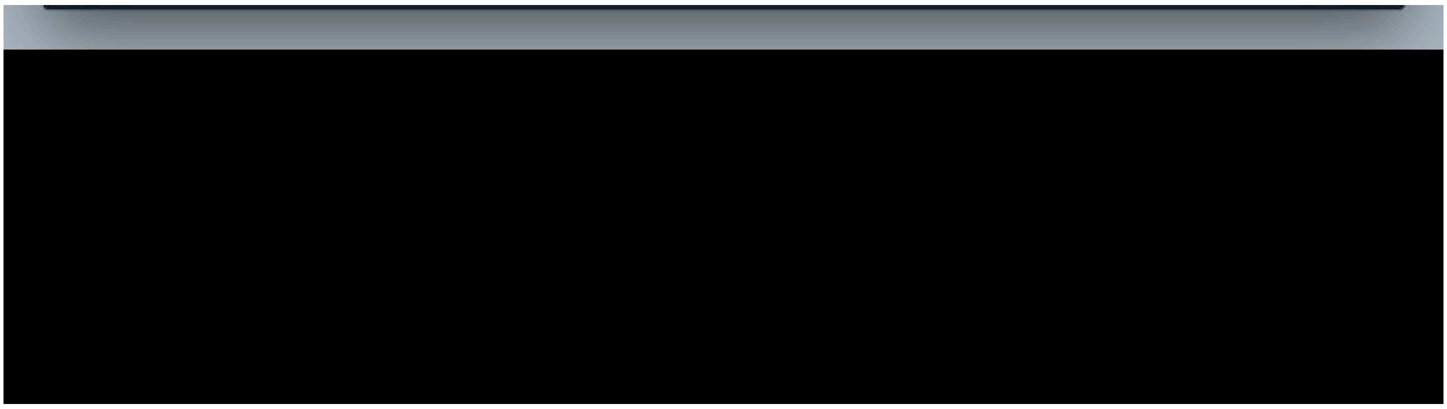
=>DYNAMIC COMPONENT:

Introduction:

Our dynamic component integrates seamlessly with Sanity CMS, enabling real-time updates and efficient content management.



```jsx
/* eslint-disable */
"use client";
import Image from "next/image";
import React, { useEffect, useState } from "react";
import Card from "@/components/Card";
import { MdArrowDropDown } from "react-icons/md";
import Link from "next/link";
import { client } from "@/sanity/lib/sanityClient";

const query = `*[_type == "product"]{
  category, name, slug, "imageUrl": image.asset->url, price, quantity, tags, description, features, dimensions, _id
}`;

const page = () => {
  const [products, setProducts] = useState([]);
  useEffect(() => {
    (async () => {
      const data = await client.fetch(query);
      setProducts(data);
    })();
  }, []);
  return (
    <div className="w-full pb-10">
      <Heading />
      <Bar />
      <div className="w-full flex flex-wrap gap-10 items-center justify-center xs:pt-10 pt-5">
        {products &&
          products.map(({ imageUrl, name, price, _id, category }, ind) => (
            <Link href={`/products/${category}/${_id}`}>
              <Card key={ind} image={imageUrl} name={name} price={price} />
            </Link>
          ))}
      </div>
      <Link href="/products">
        <div className="w-full flex justify-center">
          <button className="bg-lightGray h-12 w-36 capitalize text-sm">
            view collection
          </button>
        </div>
      </Link>
    </div>
  );
};

export default page;

const Heading = () => {
  return (
    <div className="relative w-full sm:h-48 h-32 bg-black">
      <Image src="/head.jpeg" alt="" fill={true} className="object-cover" />
      <h1 className="absolute xs:left-10 left-1/2 xs:bottom-5 bottom-1/2 max-xs:translate-y-1/2 max-xs:-translate-x-1/2 text-3xl text-white font-clash max-xs:w-52">
        All products
      </h1>
    </div>
  );
};

const Bar = () => {
  return (
    <div className="w-full py-4 flex xs:justify-between justify-center items-center sm:px-10 px-5">
      <ul className="flex gap-5  max-xs:gap-5">
        {["category", "product type", "price", "brand"].map((val, ind) => (
          <li
            key={ind}
            className={`flex items-center capitalize max-xs:px-3 max-xs:py-2 text-darkPrimary max-md:text-xs ${
              ind > 1 ? "max-xs:hidden" : "max-xs:bg-lightGray"
            }`}
          >
            <h3>{val}</h3>
            <MdArrowDropDown className="h-4 w-4" />
          </li>
        ))}
      </ul>
      <span className="xs:flex hidden gap-7 max-md:text-xs">
        <h2>Sorted by:</h2>
        <span className="flex items-center">
          <h2>Date added</h2>
          <MdArrowDropDown />
        </span>
      </span>
    </div>
  );
};
```



=>FEATURES:

- API-driven content rendering

- Real-time updates

- Efficient content management

- Scalable and customizable design

#CONCLUSION:

Day #03 focused on API setup, integration, and migration. The project is on track, with the next steps refining the frontend template and testing.