**Day 3 : API Integration And Migration Setup**

This document highlights the work completed for Day-03 Hackathon03 of the CarryIT MarketPlace hackathon .It covers Custom migration , Groq Query , Schema creation and Displaying data on the frontend.Each Page is filled with explanation of the particular images and the working done for integrating the API.

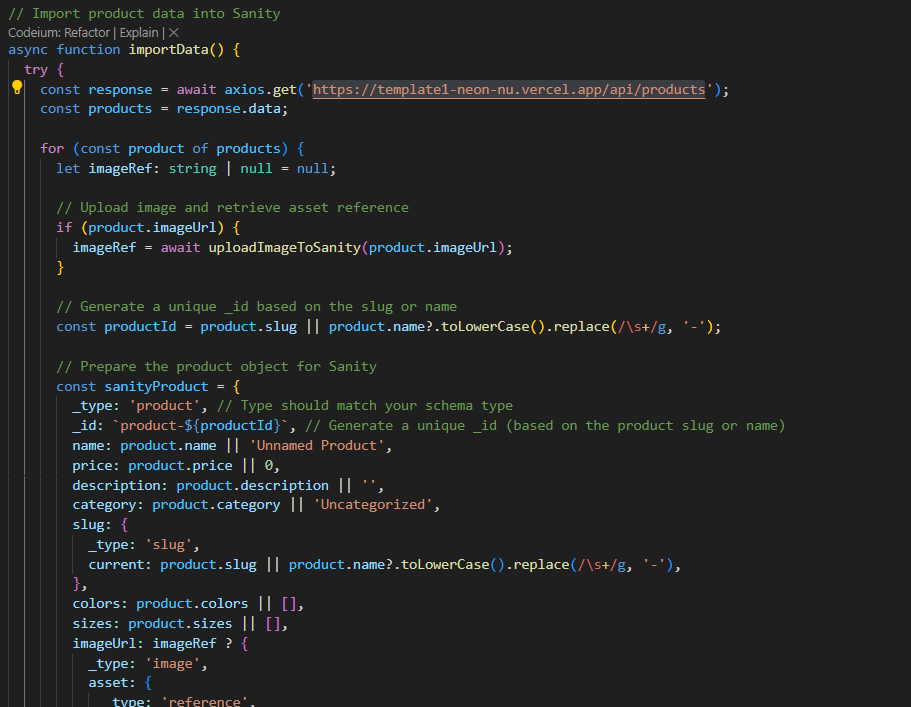
**LETS DIVE IN** :

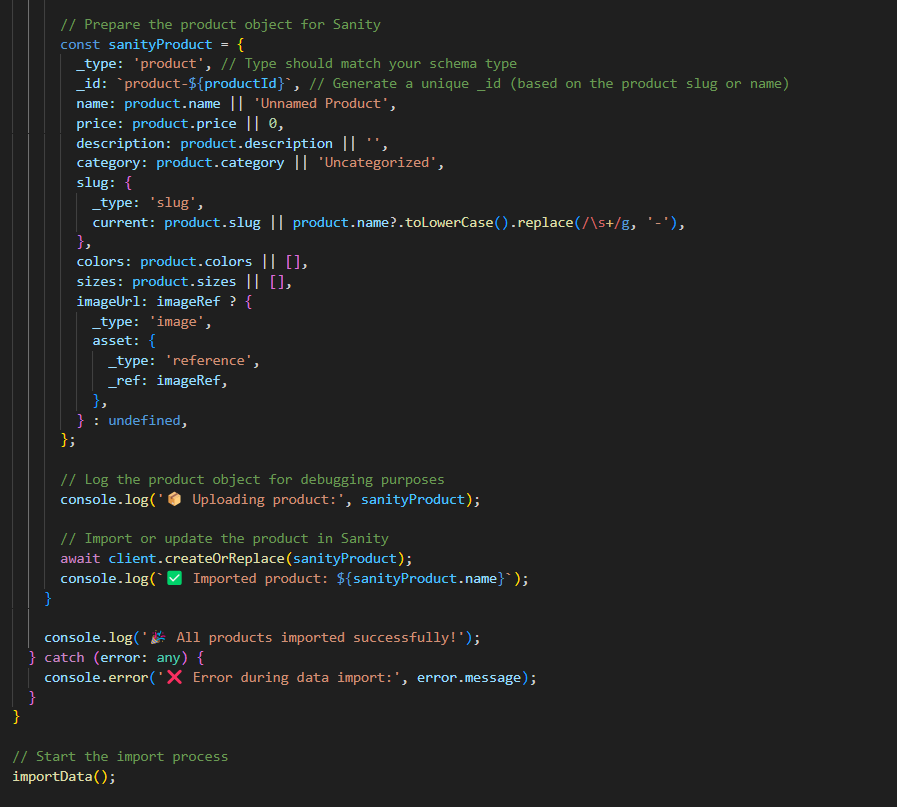
API Endpoint Setup : [**https://template1-neon-nu.vercel.app/api/products**](https://template1-neon-nu.vercel.app/api/products)

Axios is used to fetch data from the API

**Migration File** :

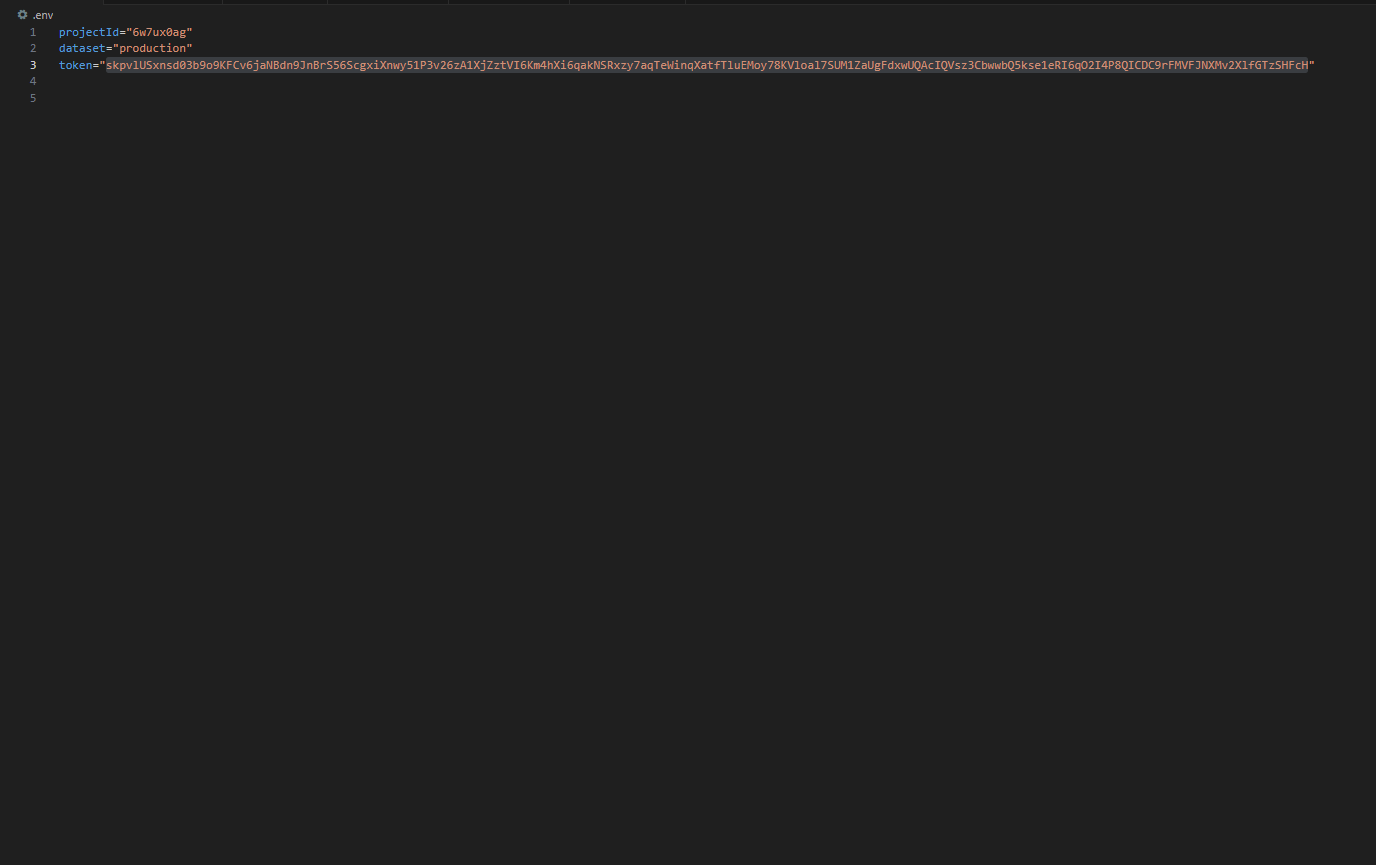
****

****

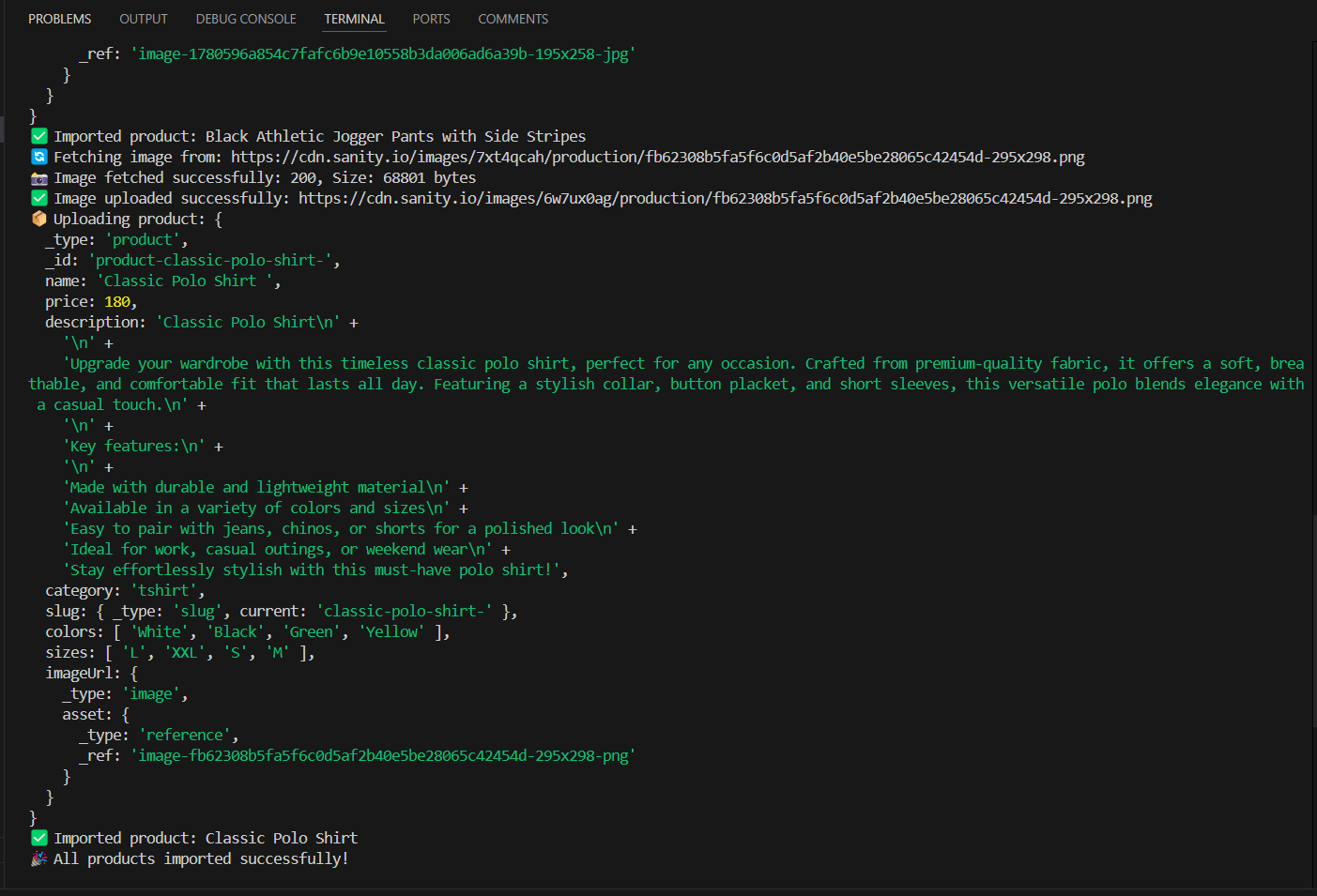
****

The Above Migration file is used for migrating the data successfully on Sanity   
**SANITY API SETUP** :

* The above migration file is used to connect to Sanity through Sanity Project ID , Project dataset and project token.
* The token is clearly private and should be kept in the .env.local file where it cannot be accessed by any third party ,they are sensitive and secure.
* In the File I have used the API Link where all the data was present and all we had to do was call the API endpoints.



**MIGRATION SUCCESS!!**



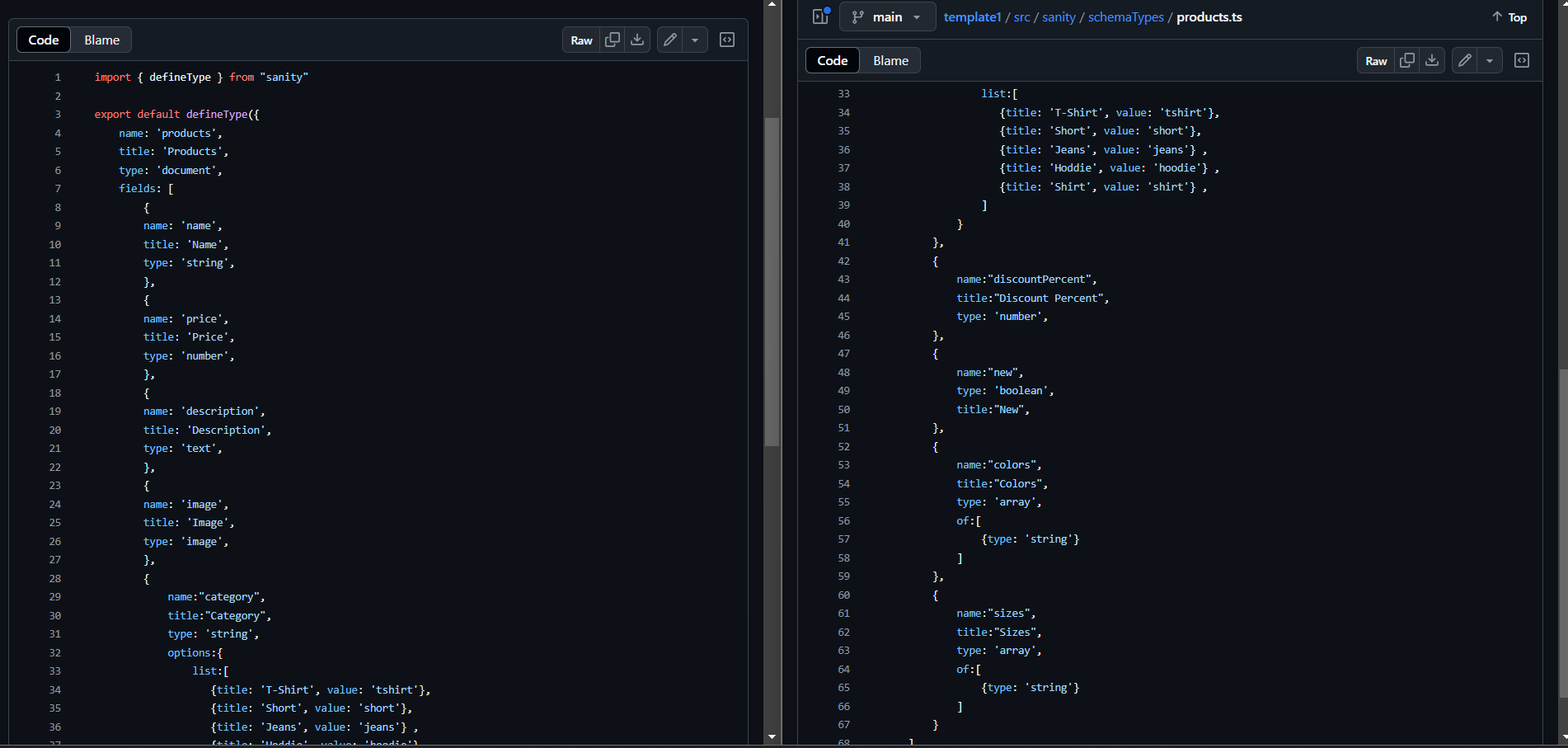
**HOW DID I FETCH THE DATA ON SANITY:**

* To Fetch the data on sanity, the project ID , Project dataset and token should match from the project created on Sanity

* In this way the data is all uploaded on the Sanity and to see the data, a schema should be defined in the SchemaTypes folder in the product.ts file.

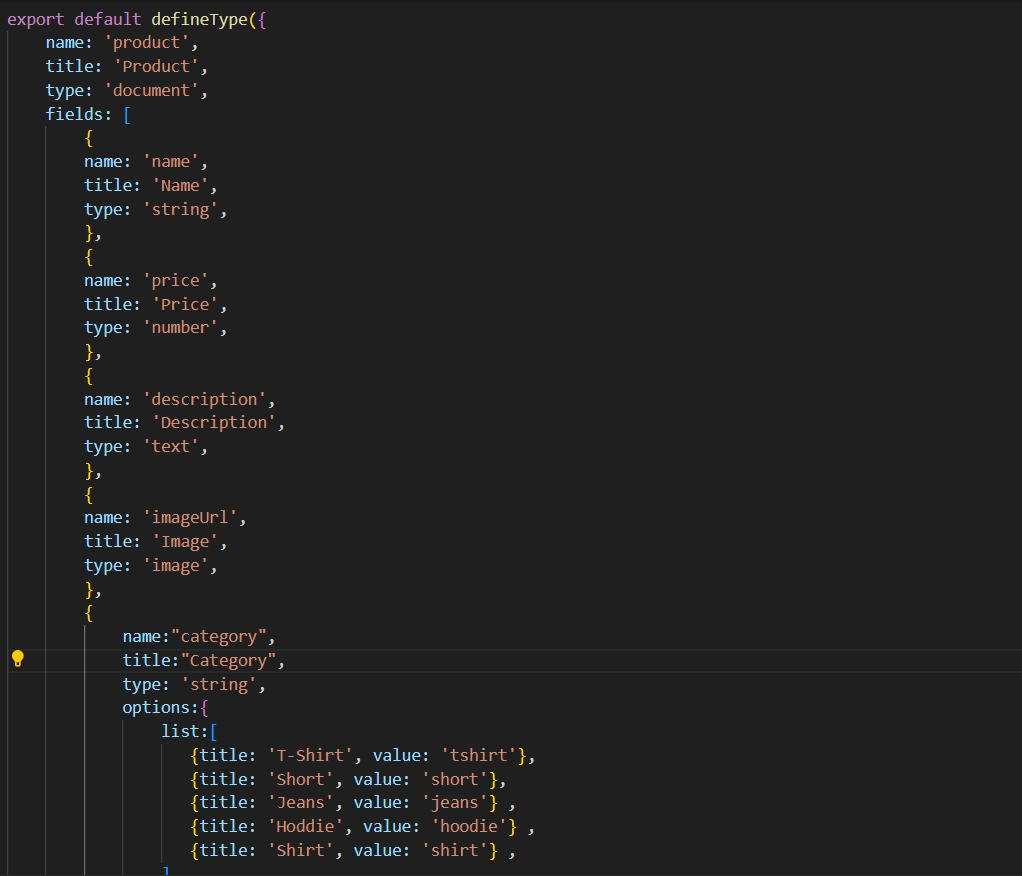
**ADJUSTMENTS MADE TO SCHEMA:**

**Previous Schema:**



* The old schema had issues in it where the name of an image was kept “image” but the name given in the API was “imageUrl”.
* This API was provided to call the data through the use of schema defined in the project.

**Updated Schema**:

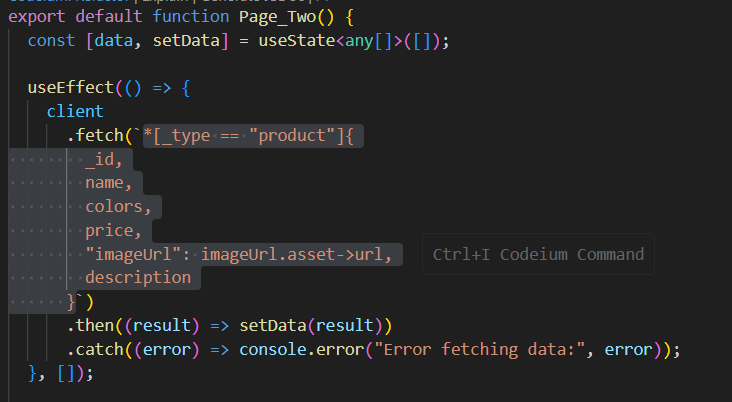


**GROQ QUERY :**

* A GROQ query is used to retrieve specific information from a JSON document database, primarily within the Sanity CMS, by allowing users to precisely define the data they need from various sets of documents, essentially acting as a powerful and flexible query language to extract relevant data from a complex, unstructured data structure like JSON
* Groq query is a powerful tool to extract all the unstructured data in a structured form and used to display the data on the frontend .

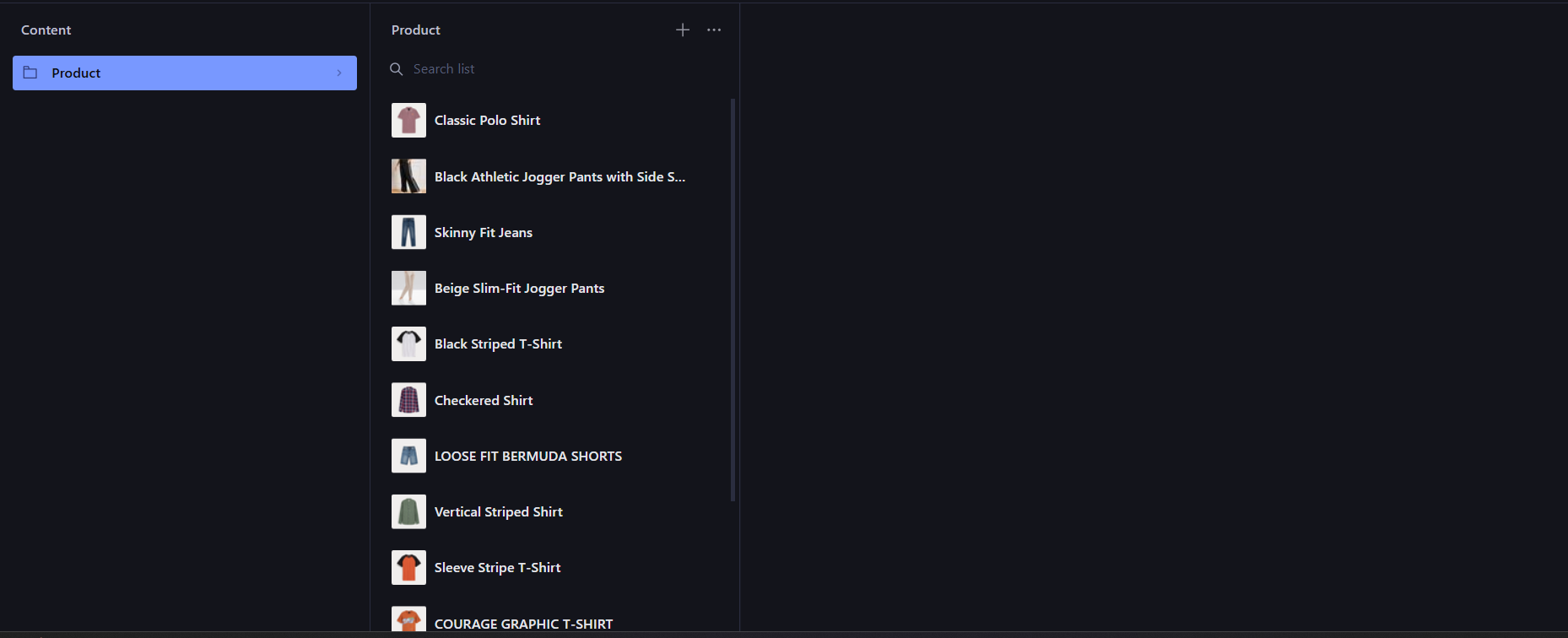


* The groq query extracted all the things I wanted to show on the display



* The data will be stored in the data useState and then it will be mapped to iterate through all the data in the Groq query.
* It will be iterated through each item in the array

**POPULATED SANITY CMS FIELDS:**



* The Products are fetched on the Sanity ,confirming that the data is fetched in the Sanity.

**DATA SUCCESSFULLY FETCHED ON THE FRONTEND:**

