Day 03 – Hackathon Documentation: API Integration and Data Migration

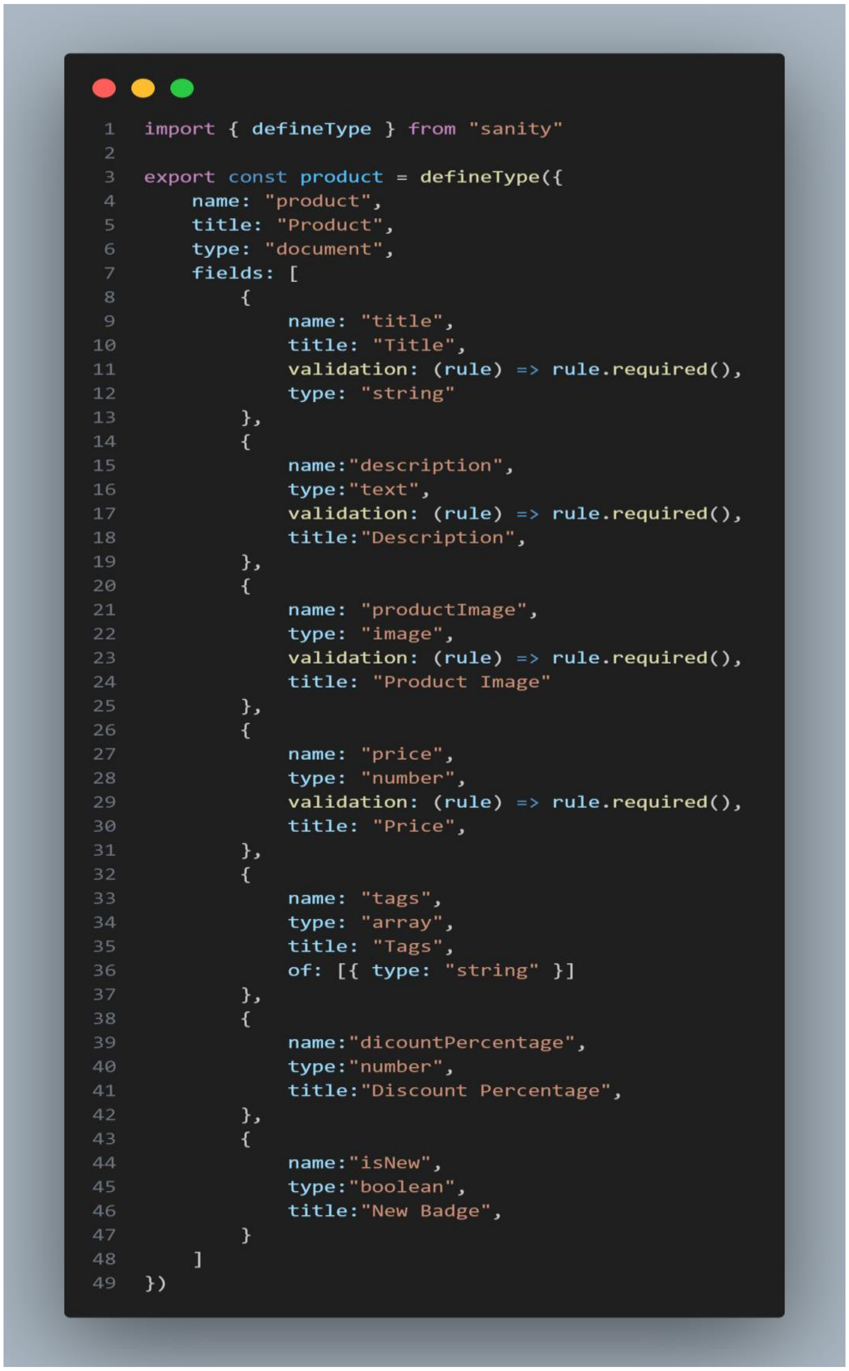
API Integration Process Overview

On Day 3 of the hackathon, we focused on integration APIs to fetch product data and migrating existing data into sanity CMS backend.

Steps For API Integration

1. Import API Call Logic
    - Implement API call logic in the importData.mjs script to fetch and push data to sanity CMS.
2. Schema Adjustment
    - Updated the Product.ts Schema in the index.ts file within the SchemaTypes folder

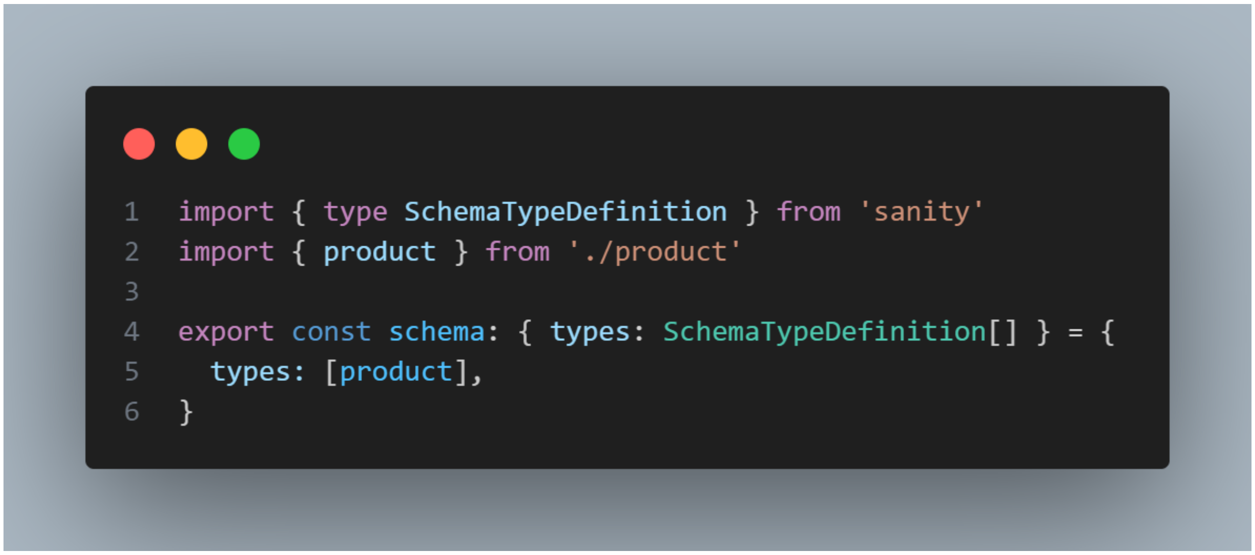
1. Product Schema:



```
import { defineType } from "sanity"

export const product = defineType({
    name: "product",
    title: "Product",
    type: "document",
    fields: [
        {
            name: "title",
            title: "Title",
            validation: (rule) => rule.required(),
            type: "string"
        },
        {
            name:"description",
            type:"text",
            validation: (rule) => rule.required(),
            title:"Description",
        },
        {
            name: "productImage",
            type: "image",
            validation: (rule) => rule.required(),
            title: "Product Image"
        },
        {
            name: "price",
            type: "number",
            validation: (rule) => rule.required(),
            title: "Price",
        },
        {
            name: "tags",
            type: "array",
            title: "Tags",
            of: [{ type: "string" }]
        },
        {
            name:"dicountPercentage",
            type:"number",
            title:"Discount Percentage",
        },
        {
            name:"isNew",
            type:"boolean",
            title:"New Badge",
        }
    ]
})
```

2. Updated Schema:

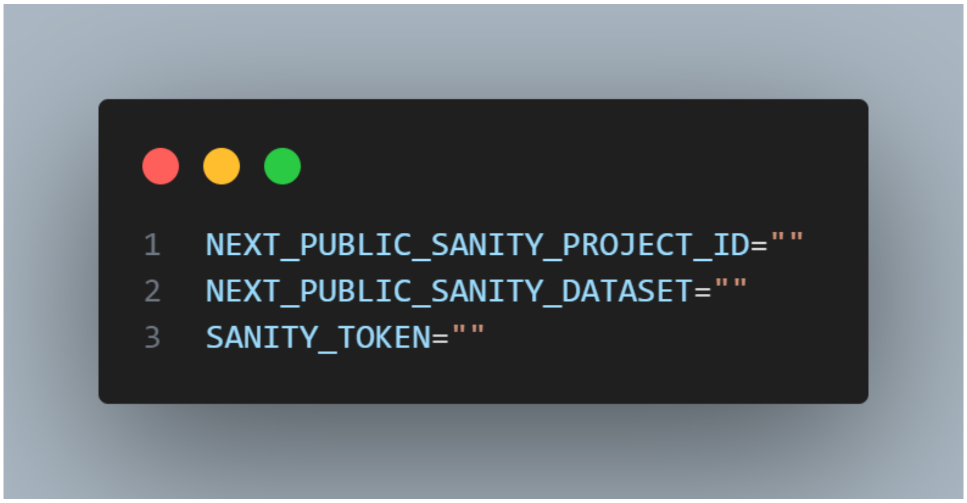


```
import { type SchemaTypeDefinition } from 'sanity'
import { product } from './product'

export const schema: { types: SchemaTypeDefinition[] } = {
  types: [product],
}
```

3. Configure Environment Variable:

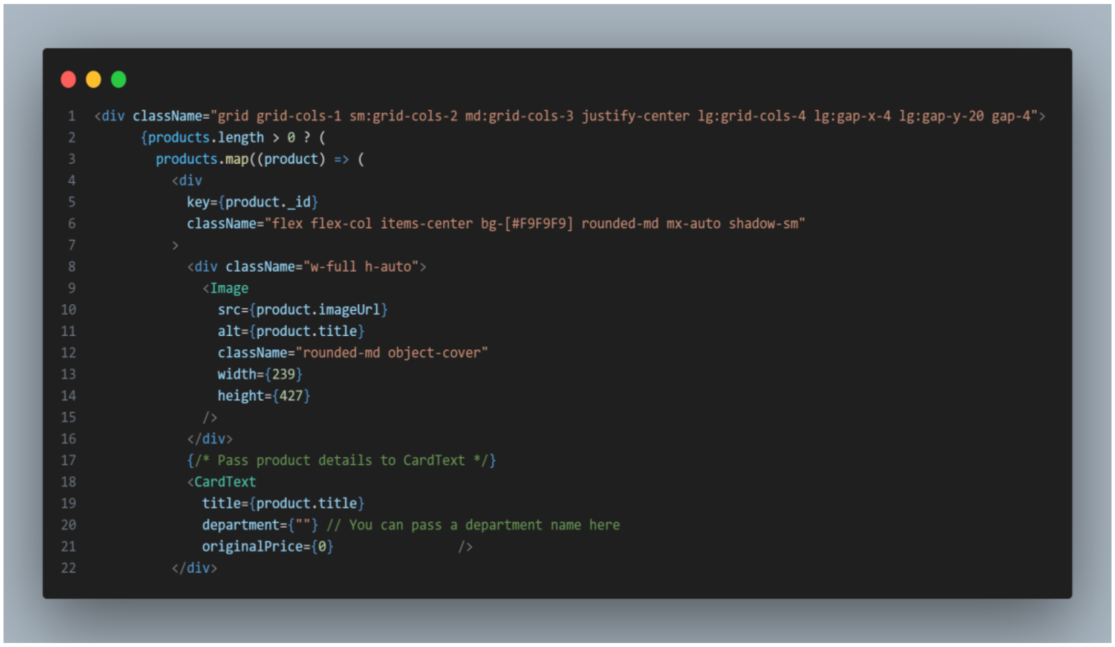
To securely store your sanity project credentials, configure the environment variable inn a. env File. Add the following entities:



```
NEXT_PUBLIC_SANITY_PROJECT_ID=""
NEXT_PUBLIC_SANITY_DATASET=""
SANITY_TOKEN=""
```

4. Data Processing:

Map the fetched data to match the desired schema structure for sanity CMS.



```
<div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 justify-center lg:grid-cols-4 lg:gap-x-4 lg:gap-y-20 gap-4">
      {products.length > 0 ? (
        products.map((product) => (
          <div
            key={product._id}
            className="flex flex-col items-center bg-[#F9F9F9] rounded-md mx-auto shadow-sm"
          >
            <div className="w-full h-auto">
              <Image
                src={product.imageUrl}
                alt={product.title}
                className="rounded-md object-cover"
                width={239}
                height={427}
              />
            </div>
            {/* Pass product details to CardText */}
            <CardText
              title={product.title}
              department={""} // You can pass a department name here
              originalPrice={0}                />
          </div>
```

5. Error Handling

Ensure smooth data processing with robust error handling and data validation.

Key Learning Outcomes:

- Understand API integration techniques in a Next.js Project.
- Learn data migration method to import API data into sanity CMS.
- Gain practical experience in handling schema and ensuring compatibility with data sources.
- Develop robust error-handling strategy.