Hackathon Day 3

API Integration Report

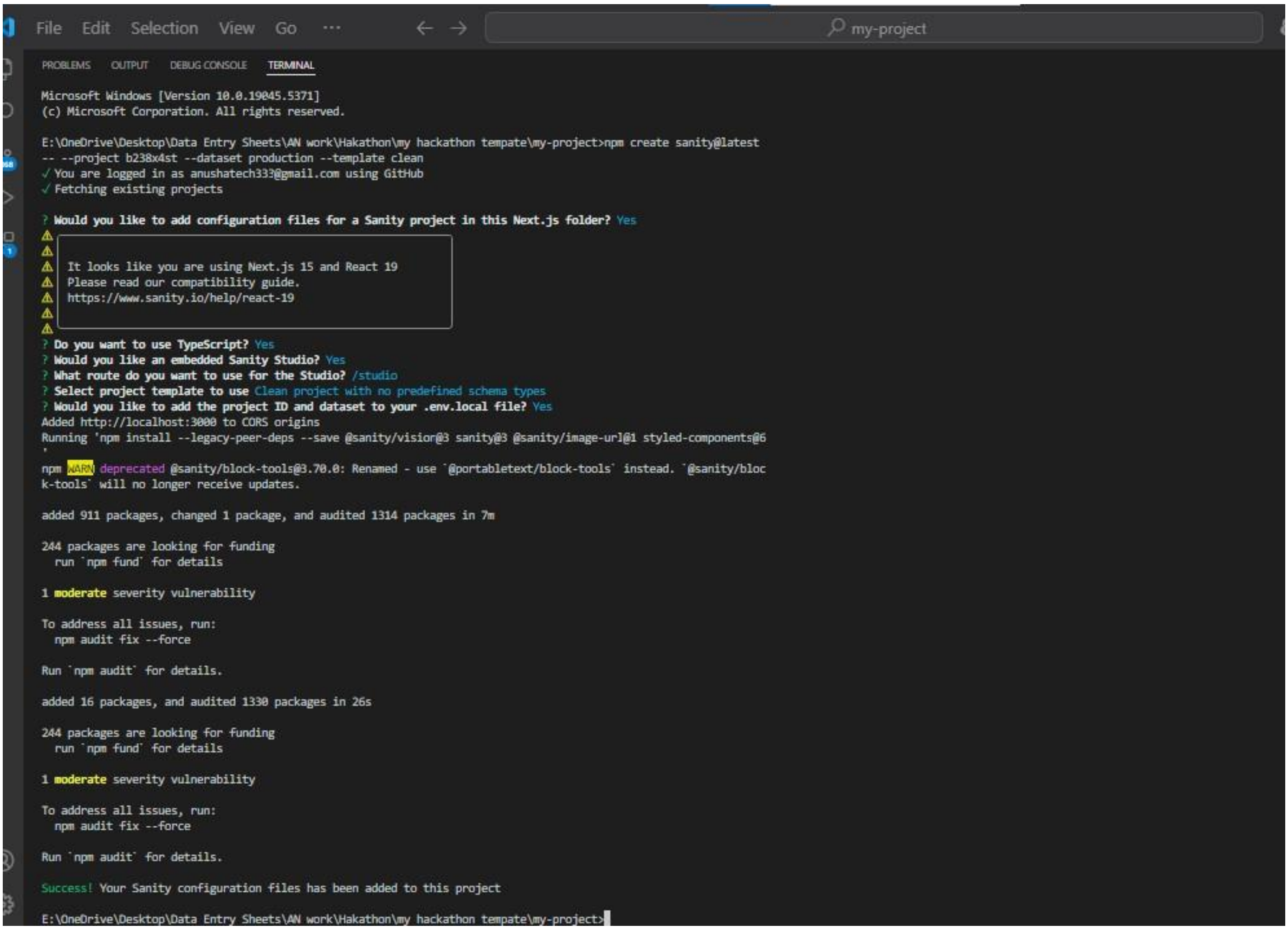
This report provides a detailed account of the API integration process and data migration for the project. It outlines the steps taken to integrate an external API, adjustments made to schemas, the migration strategy, and the tools used during implementation.

Step-by-Step Process

Setting Up Sanity

After initializing the Next.js application, the next step is installing **Sanity CMS**. To do this, run the following command,

npm create sanity@latest



```
Microsoft Windows [Version 10.0.19045.5371]
(c) Microsoft Corporation. All rights reserved.

E:\OneDrive\Desktop\Data Entry Sheets\AN work\Hakathon\my hackathon tempate\my-project>npm create sanity@latest
-- --project b238x4st --dataset production --template clean
✓ You are logged in as anushatech333@gmail.com using GitHub
✓ Fetching existing projects

? Would you like to add configuration files for a Sanity project in this Next.js folder? Yes
⚠ ┌──────────────────────────────────────────────┐
⚠ │                                              │
⚠ │ It looks like you are using Next.js 15 and React 19
⚠ │ Please read our compatibility guide.         │
⚠ │ https://www.sanity.io/help/react-19          │
⚠ │                                              │
⚠ └──────────────────────────────────────────────┘
? Do you want to use TypeScript? Yes
? Would you like an embedded Sanity Studio? Yes
? What route do you want to use for the Studio? /studio
? Select project template to use Clean project with no predefined schema types
? Would you like to add the project ID and dataset to your .env.local file? Yes
Added http://localhost:3000 to CORS origins
Running 'npm install --legacy-peer-deps --save @sanity/vision@3 sanity@3 @sanity/image-url@1 styled-components@6
'
npm WARN deprecated @sanity/block-tools@3.70.0: Renamed - use `@portabletext/block-tools` instead. `@sanity/bloc
k-tools` will no longer receive updates.

added 911 packages, changed 1 package, and audited 1314 packages in 7m

244 packages are looking for funding
  run `npm fund` for details

1 moderate severity vulnerability

To address all issues, run:
  npm audit fix --force

Run `npm audit` for details.

added 16 packages, and audited 1330 packages in 26s

244 packages are looking for funding
  run `npm fund` for details

1 moderate severity vulnerability

To address all issues, run:
  npm audit fix --force

Run `npm audit` for details.

Success! Your Sanity configuration files has been added to this project

E:\OneDrive\Desktop\Data Entry Sheets\AN work\Hakathon\my hackathon tempate\my-project>
```

This command sets up **Sanity CMS** in your project, allowing you to manage and structure content efficiently.

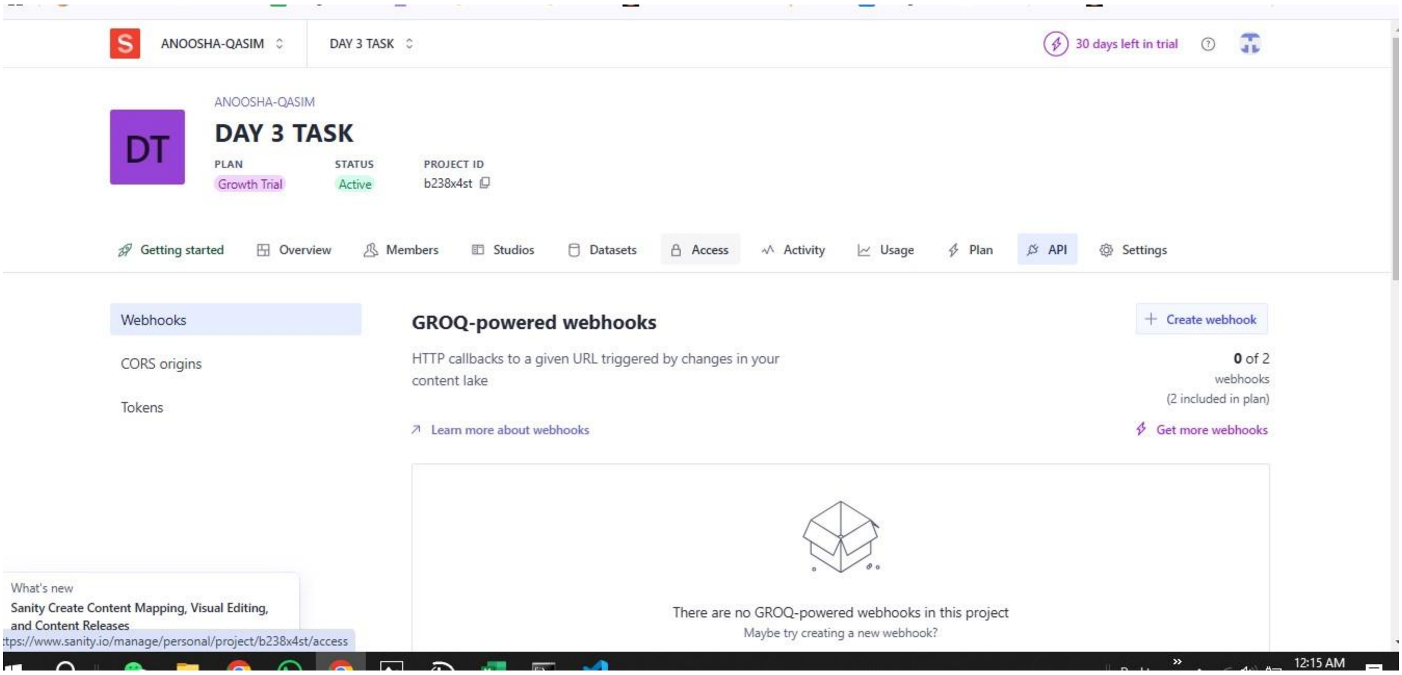
Important Note:

Ensure that Sanity CMS is installed in the root directory of your Next.js project. Setting it up correctly within the right directory ensures a real-world development experience, making API integration and data migration smoother.

Setting Up Sanity Project ID and Token

Create a Sanity Project and Get the Credentials

First, sign up or log in to **sanity.io** and create a new project. After creating the project, you will get a **Project ID** and a **Token**. These credentials are required for integrating Sanity with your application



ANOOSHA-QASIM

DAY 3 TASK

PLAN: Growth Trial

STATUS: Active

PROJECT ID: b238x4st

GROQ-powered webhooks

HTTP callbacks to a given URL triggered by changes in your content lake

0 of 2 webhooks (2 included in plan)

There are no GROQ-powered webhooks in this project

Maybe try creating a new webhook?

Sanity Create Content Mapping, Visual Editing, and Content Releases

ttps://www.sanity.io/manage/personal/project/b238x4st/access

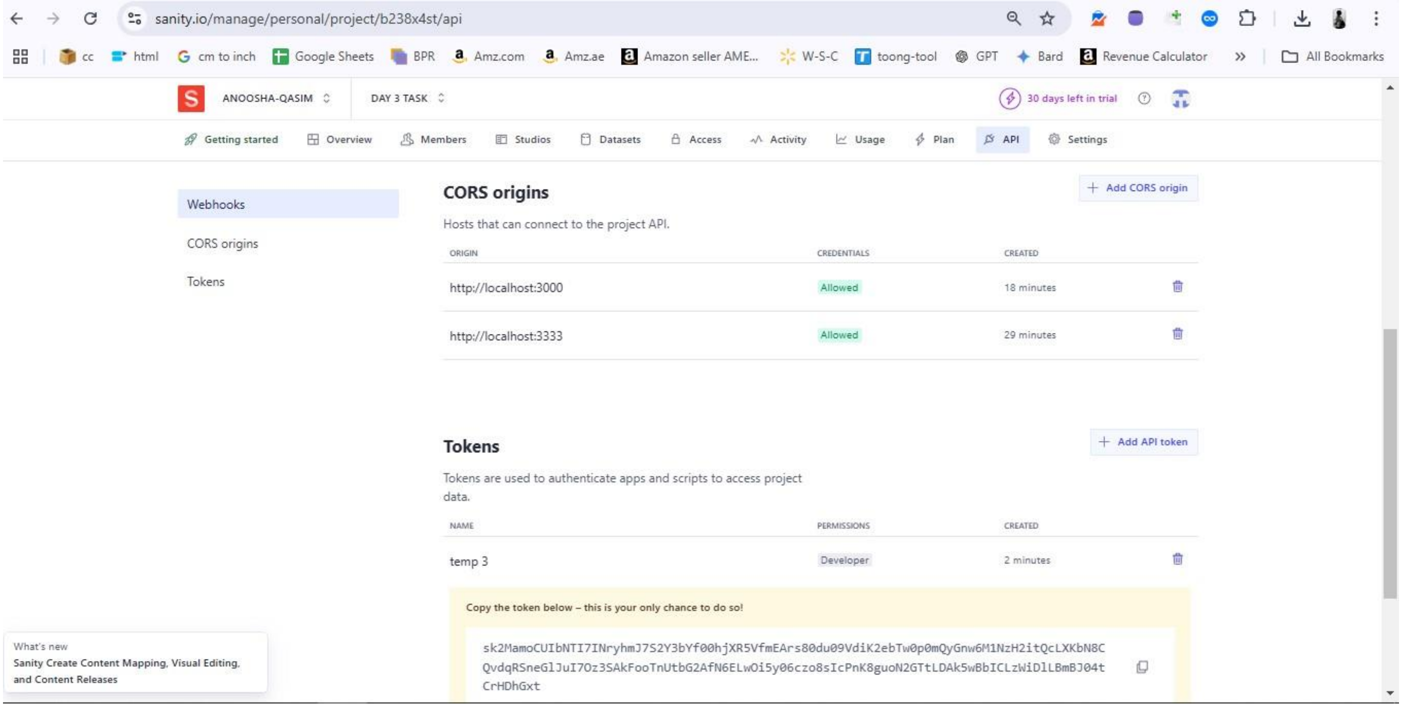
Generate an Authentication Token

To enable secure access, generate an **API token** from the **Sanity dashboard**. This token is necessary for connecting your application to Sanity's content management system.

Store Credentials Securely

After generating the token, store the Project ID and Token in your project's **environment variables**:

- Add them to the .env.local file in your Next.js project.
- Use them in importData.js or other necessary configuration files to ensure seamless data integration.



CORS origins

Hosts that can connect to the project API.

ORIGIN	CREDENTIALS	CREATED
http://localhost:3000	Allowed	18 minutes
http://localhost:3333	Allowed	29 minutes

Tokens

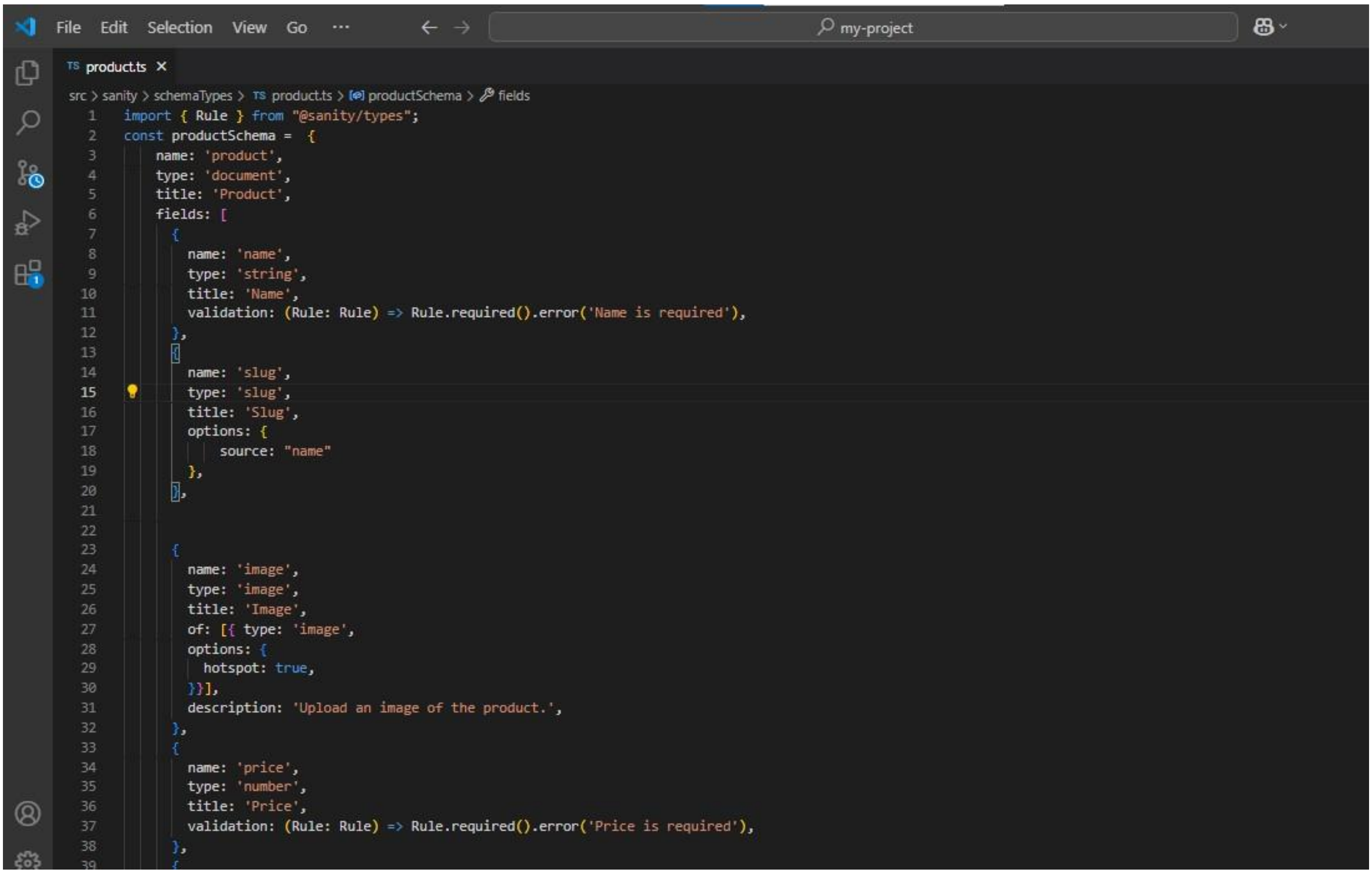
Tokens are used to authenticate apps and scripts to access project data.

NAME	PERMISSIONS	CREATED
temp 3	Developer	2 minutes

Copy the token below – this is your only chance to do so!

```
sk2MamoCUIbNTI7INryhmJ7S2Y3bYf00hjXR5VfmEArs80du09VdiK2ebTw0p0mQyGnw6M1NzH2itQcLXKbN8C
QvdqRSneGlJuI7Oz3SAkFooTnUtbG2AfN6ELwOi5y06czo8sIcPnK8guoN2GTtLDAk5wBbICLzWiDlLBmBJ04t
CrHDhGxt
```

Add product schema


File
Edit
Selection
View
Go
my-project
product.ts
src > sanity > schemaTypes > product.ts > productSchema > fields

```
import { Rule } from "@sanity/types";
const productSchema =  {
    name: 'product',
    type: 'document',
    title: 'Product',
    fields: [
      {
        name: 'name',
        type: 'string',
        title: 'Name',
        validation: (Rule: Rule) => Rule.required().error('Name is required'),
      },
      {
        name: 'slug',
        type: 'slug',
        title: 'Slug',
        options: {
            source: "name"
        },
      },


      {
        name: 'image',
        type: 'image',
        title: 'Image',
        of: [{ type: 'image',
        options: {
          hotspot: true,
        }}],
        description: 'Upload an image of the product.',
      },
      {
        name: 'price',
        type: 'number',
        title: 'Price',
        validation: (Rule: Rule) => Rule.required().error('Price is required'),
      },
      {
```

Understanding Schema in Sanity CMS

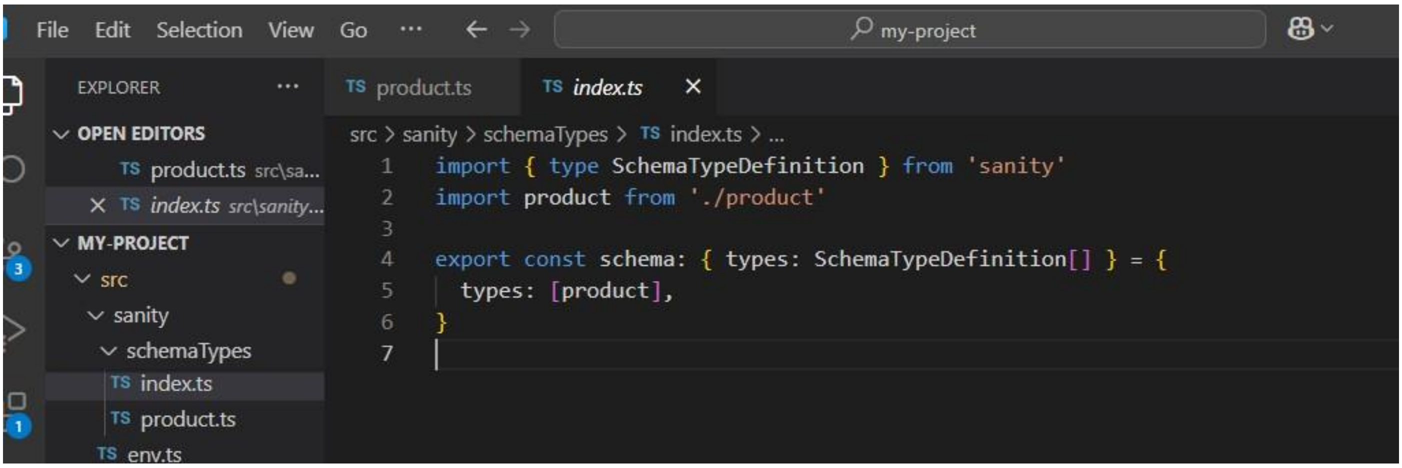
The schema provided by the instructor helps establish a clear connection between **API integration** and Sanity CMS data storage. It ensures that data received from the API is structured correctly and stored efficiently in Sanity CMS.

Setting Up the Schema

Once the schema is created, the next step is to import and render the product schema in the index.ts file inside the sanity Types directory. This step is crucial as it registers the schema within Sanity.

Code Implementation

In the index.ts file, import the necessary dependencies and register the product schema:

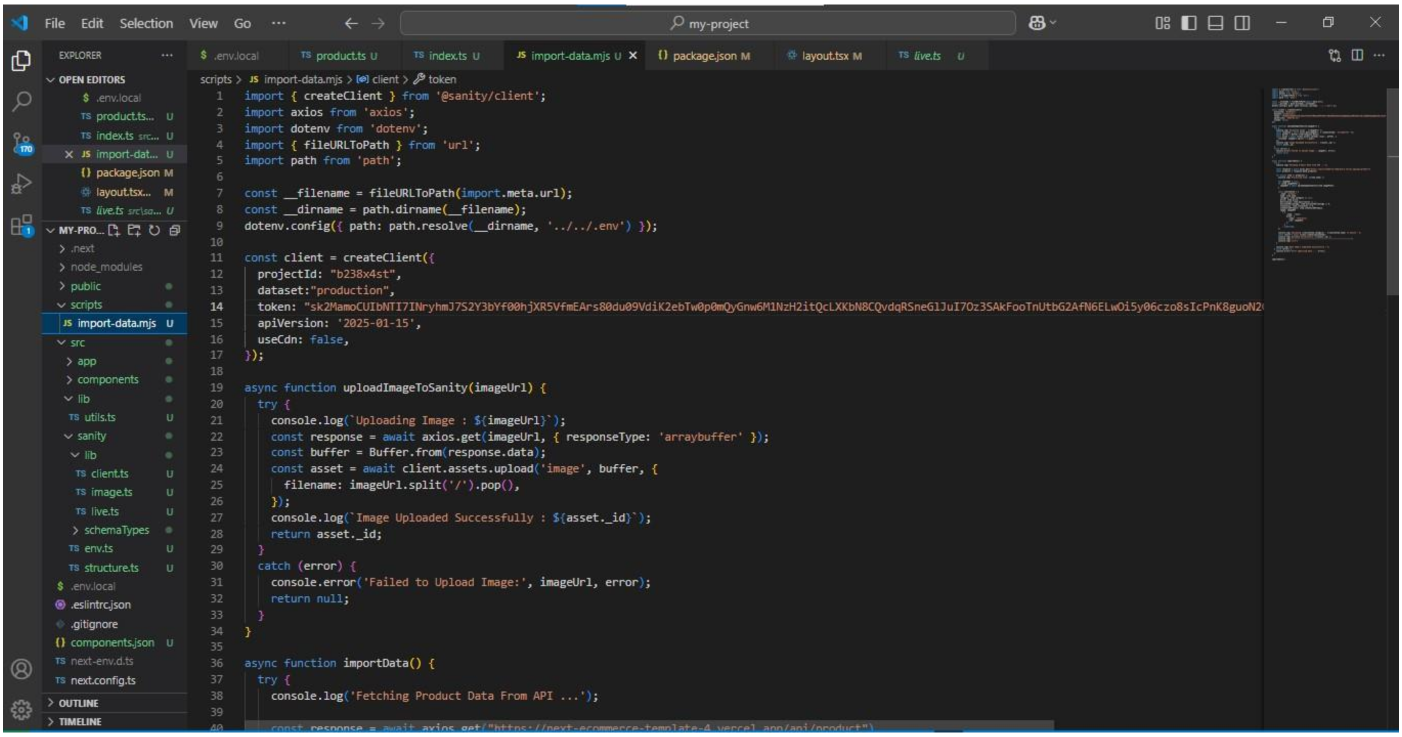


```ts
import { type SchemaTypeDefinition } from 'sanity'
import product from './product'

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

```

This ensures that the **product schema** is recognized and displayed in Sanity CMS.

Data Migration Process

To efficiently move data from an API to **Sanity CMS**, we use a **data migration script**. This script helps in fetching, transforming, and structuring the data properly, ensuring it aligns with the schema set in Sanity.

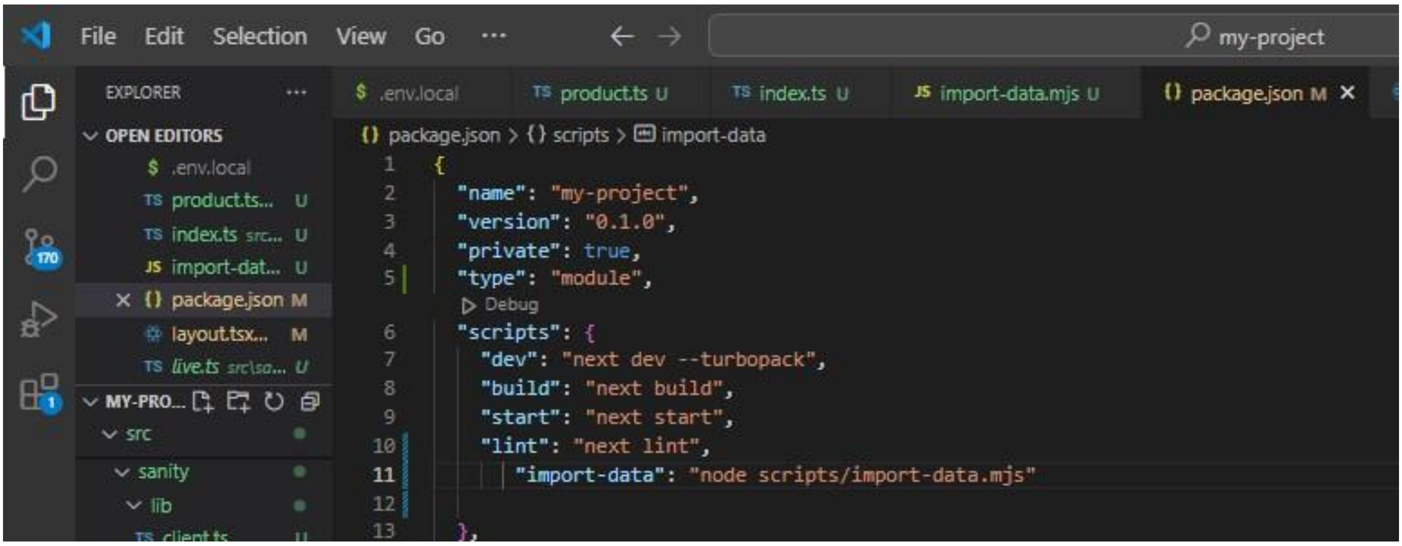


```javascript
import { createClient } from '@sanity/client';
import axios from 'axios';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';
import path from 'path';

const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);
dotenv.config({ path: path.resolve(__dirname, '../../.env') });

const client = createClient({
  projectId: "b238x4st",
  dataset:"production",
  token: "sk2MamoCUIbNTI7INryhmJ7S2Y3bYf00hjXR5VfmEArs80du09VdiK2ebTw0p0mQyGnw6M1NzH2itQcLXKbN8CQvdqRSneGlJuI7Oz3SAkFooTnUtbG2AfN6ELwOi5y06czo8sIcPnK8guoN2
  apiVersion: '2025-01-15',
  useCdn: false,
});

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 importData() {
  try {
    console.log('Fetching Product Data From API ...');

    const response = await axios.get("https://next-ecommerce-template-4.vercel.app/api/product")
```

By utilizing this script, we gain hands-on experience with configuration setup, handling errors, and ensuring smooth data transfer. After understanding the schema, I successfully fetched data from the API and stored it in **Sanity CMS** using this approach.

Key Benefits of Using the Migration Script:

- Ensures that the imported data is accurate and properly structured according to the predefined schema.
- Before importing large datasets, it is essential to create a backup of the Sanity project to prevent data loss. This is a best practice, especially when handling marketplace-related data.
- Working with this migration process has provided me with real-world experience in handling data migration across different sources, making API-driven content management more efficient.



```json
{
  "name": "my-project",
  "version": "0.1.0",
  "private": true,
  "type": "module",
  "scripts": {
    "dev": "next dev --turbopack",
    "build": "next build",
    "start": "next start",
    "lint": "next lint",
      "import-data": "node scripts/import-data.mjs"

  },
```

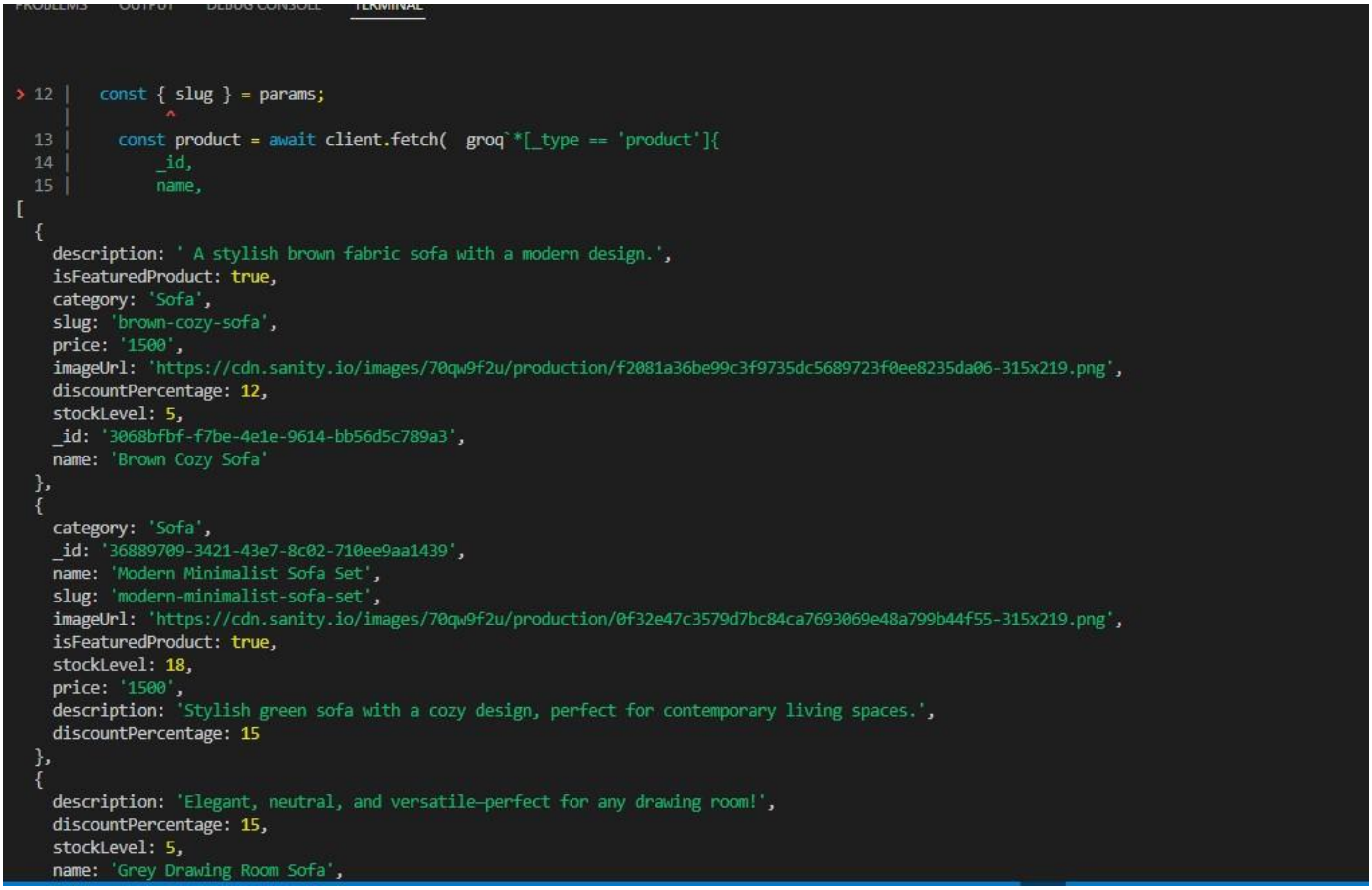
Running the Data Import Script

To execute the **data migration process**, follow these steps:

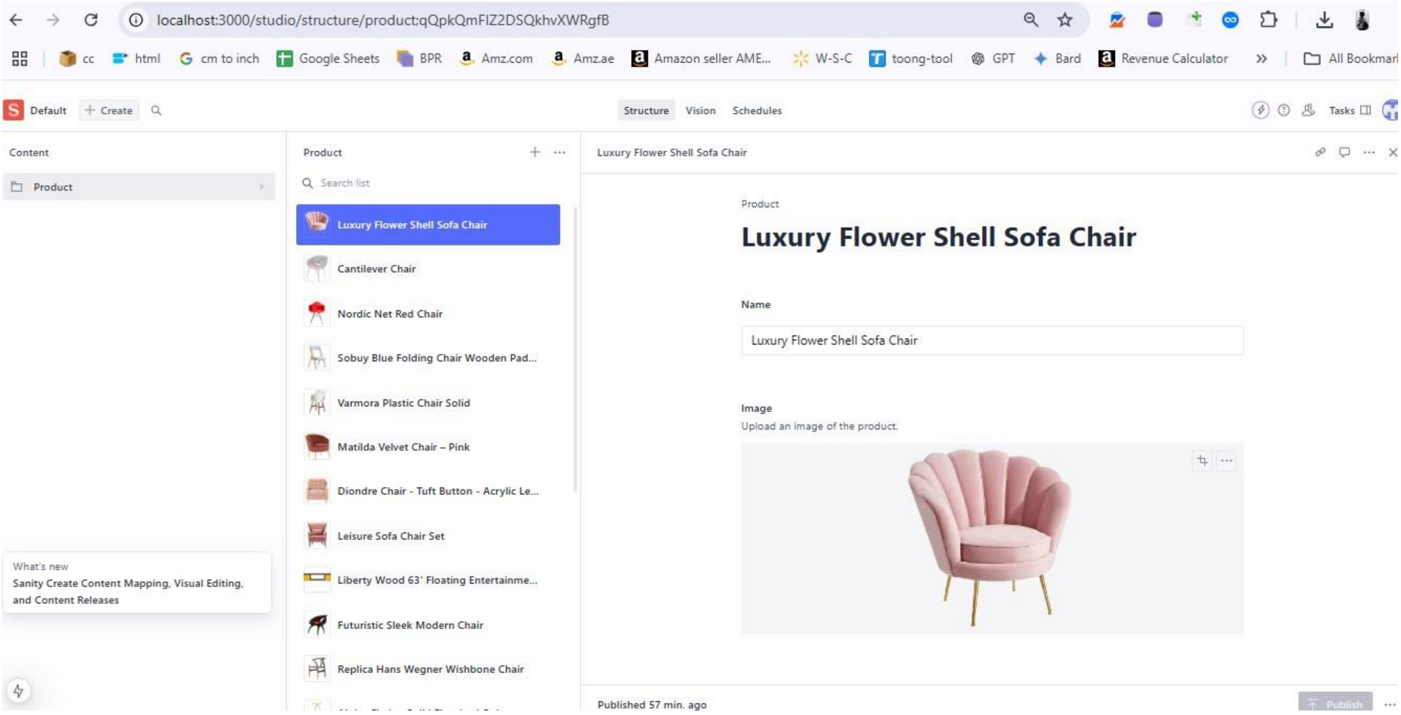
Open the terminal and run the following command:

node importData.js

This will initiate the script to fetch and import API data into Sanity CMS.



```
> 12 |   const { slug } = params;
     |           ^
  13 |     const product = await client.fetch(  groq`*[_type == 'product']{
  14 |         _id,
  15 |         name,
[
  {
    description: ' A stylish brown fabric sofa with a modern design.',
    isFeaturedProduct: true,
    category: 'Sofa',
    slug: 'brown-cozy-sofa',
    price: '1500',
    imageUrl: 'https://cdn.sanity.io/images/70qw9f2u/production/f2081a36be99c3f9735dc5689723f0ee8235da06-315x219.png',
    discountPercentage: 12,
    stockLevel: 5,
    _id: '3068bfbf-f7be-4e1e-9614-bb56d5c789a3',
    name: 'Brown Cozy Sofa'
  },
  {
    category: 'Sofa',
    _id: '36889709-3421-43e7-8c02-710ee9aa1439',
    name: 'Modern Minimalist Sofa Set',
    slug: 'modern-minimalist-sofa-set',
    imageUrl: 'https://cdn.sanity.io/images/70qw9f2u/production/0f32e47c3579d7bc84ca7693069e48a799b44f55-315x219.png',
    isFeaturedProduct: true,
    stockLevel: 18,
    price: '1500',
    description: 'Stylish green sofa with a cozy design, perfect for contemporary living spaces.',
    discountPercentage: 15
  },
  {
    description: 'Elegant, neutral, and versatile—perfect for any drawing room!',
    discountPercentage: 15,
    stockLevel: 5,
    name: 'Grey Drawing Room Sofa',
```



Product

- Luxury Flower Shell Sofa Chair
- Cantilever Chair
- Nordic Net Red Chair
- Sobuy Blue Folding Chair Wooden Pad...
- Varmora Plastic Chair Solid
- Matilda Velvet Chair – Pink
- Diondre Chair - Tuft Button - Acrylic Le...
- Leisure Sofa Chair Set
- Liberty Wood 63' Floating Entertainme...
- Futuristic Sleek Modern Chair
- Replica Hans Wegner Wishbone Chair

Luxury Flower Shell Sofa Chair

Product

Luxury Flower Shell Sofa Chair

Name

Luxury Flower Shell Sofa Chair

Image

Upload an image of the product.

Published 57 min. ago

Once the data is successfully imported into Sanity, you can retrieve it on the frontend using queries. This allows dynamic content management and seamless integration with your application.

Day 3 Checklist: Self-Validation Checklist:

API Understanding: ✓

Schema Validation: ✓

Data Migration: ✓

API Integration in Next.js: ✓

Submission Preparation: ✓