MarketPlace-Builder-Hackathon

Day-3: API Integration and Data Migration in Sanity

1/18/2025
GIAIC: Sunday(2pm to 5pm)
Qaimudin Khuwaja
00161235
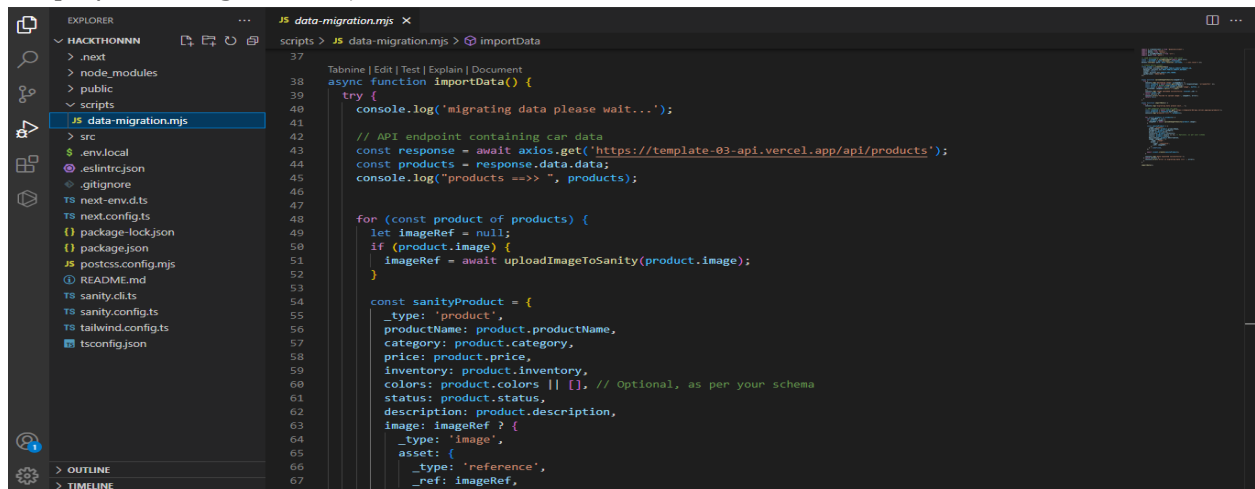
EXPLANATION OF MIGRATING DATA FROM API TO SANITY IN NEXTJS

The focus for Day 3 of the hackathon was on API Integration and Data Migration. The task was to integrate external APIs into the marketplace project.I implemented a solution that allows us to migrate and import data from an external API into sanity CMS. By the end of the day, the objective was to have a fully integrated marketplace with dynamic data fetched from APIs and correctly displayed on the frontend. Below is step by step explanation how I achieved it:
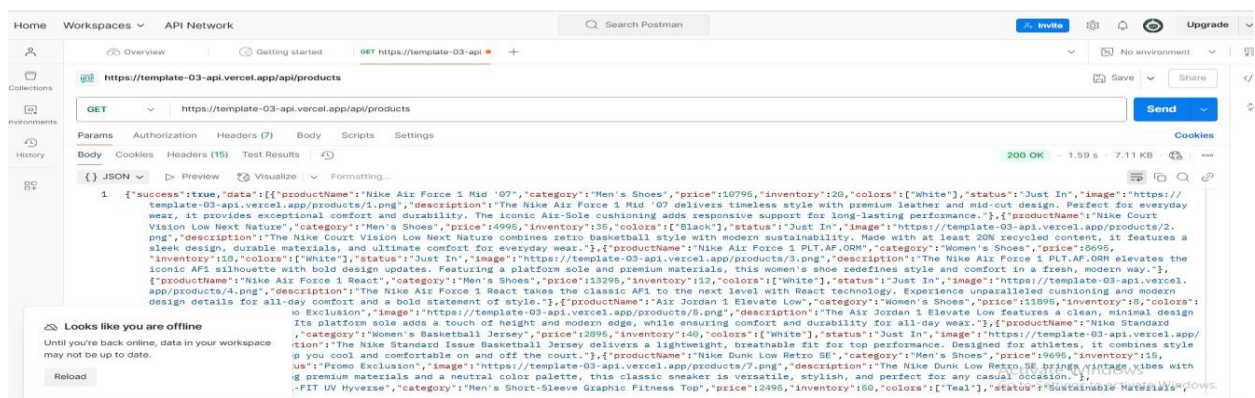
1. <u>Fetching Data from External APIs</u>

The first step in this process was to fetch data from the external API. In my Next.js application, where the frontend code was already present, I created a folder named "scripts" and within that, I created a file called "data-migration.mjs" where I wrote the code to fetch data from the API. Afterward, I tested it both in the application and through Postman API, and my data was successfully fetched.

scripts/data-migration.mjs

2. Comparing API data with Sanity Schema

After the data was fetched, the next task was to compare the data structure of the API with the Sanity schema. The Sanity schema will be used to handle the product data. The main purpose of this step was to ensure that the data coming from the API matches the Sanity schema. This step ensures that the data from the API perfectly matches the Sanity schema.

3. Data Migration Script

After confirming the schema and the API data structure, the next step was to write a migration script in existing file data-migration.mjs that wood automate the process of importing the fetched data into Sanity. A script was created that would:

1. Fetch data from API
2. Validate data with sanity schema
3. Use the sanity client to push data into sanity

```js
22  async function uploadImageToSanity(imageUrl) {
23    try {
24      console.log(`Uploading image: ${imageUrl}`);
25      const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
26      const buffer = Buffer.from(response.data);
27      const asset = await client.assets.upload('image', buffer, {
28        filename: imageUrl.split('/').pop()
29      });
30      console.log(`Image uploaded successfully: ${asset._id}`);
31      return asset._id;
32    } catch (error) {
33      console.error('Failed to upload image:', imageUrl, error);
34      return null;
35    }
36  }
37
38  async function importData() {
39    try {
40      console.log('migrating data please wait...');
41
42      // API endpoint containing car data
43      const response = await axios.get('https://template-03-api.vercel.app/api/products');
44      const products = response.data.data;
45      console.log("products ==>> ", products);
46
47
48      for (const product of products) {
49        let imageRef = null;
50        if (product.image) {
51          imageRef = await uploadImageToSanity(product.image);
52        }
```

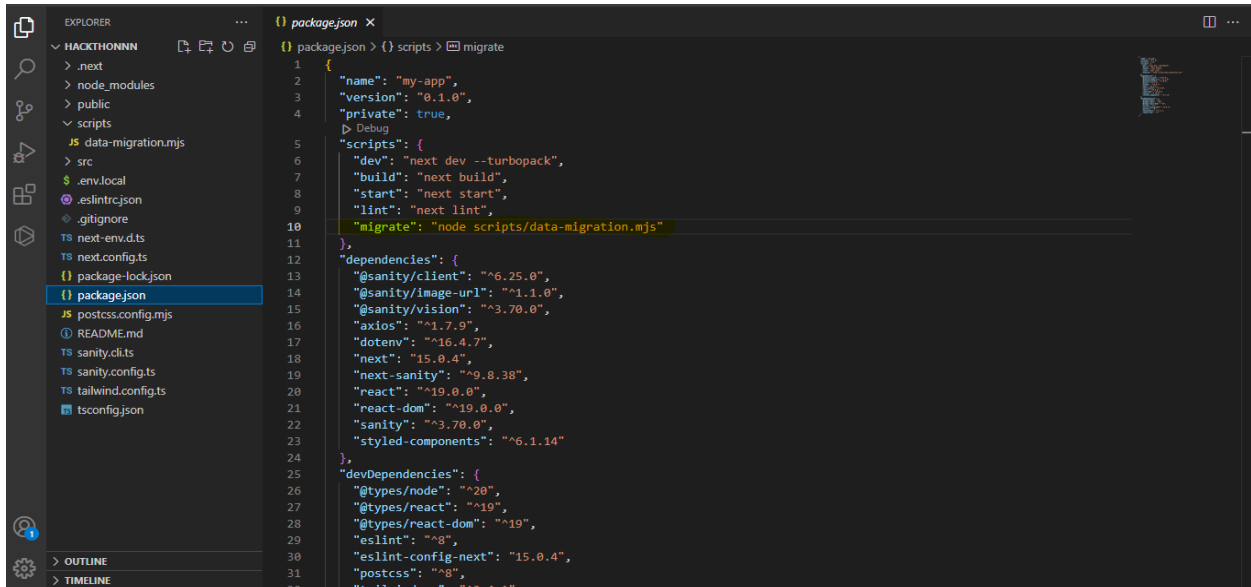# 4. Setting Up The Environment

Once the script was written, the next step was to configure Sanity. For the configuration, a token for the project needs to be generated from Sanity's official website, and by looking at the ID, DATASET, and TOKEN, it needs to be written in the .env.local file, and also provided in the data-migration.mjs file.


```js
1   import { createClient } from '@sanity/client';
2   import axios from 'axios';
3   import dotenv from 'dotenv';
4   import { fileURLToPath } from 'url';
5   import path from 'path';
6
7   // Load environment variables from .env.local
8   const __filename = fileURLToPath(import.meta.url);
9   const __dirname = path.dirname(__filename);
10  dotenv.config({ path: path.resolve(__dirname, '../.env.local') });
11
12  // Create Sanity client
13  const client = createClient({
14    projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
15    dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
16    useCdn: false,
17    token: process.env.SANITY_API_TOKEN,
18    apiVersion: '2021-08-31'
19  });
20
21
22  async function uploadImageToSanity(imageUrl) {
23    try {
24      console.log(`Uploading image: ${imageUrl}`);
25      const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
26      const buffer = Buffer.from(response.data);
27      const asset = await client.assets.upload('image', buffer, {
28        filename: imageUrl.split('/').pop()
29      });
30      console.log(`Image uploaded successfully: ${asset._id}`);
31      return asset._id;
```

5. Modifying package.json

To run the migration script, the necessary npm package was edit to the package.json file under the scripts section. This made it easier to run the script through the command line.
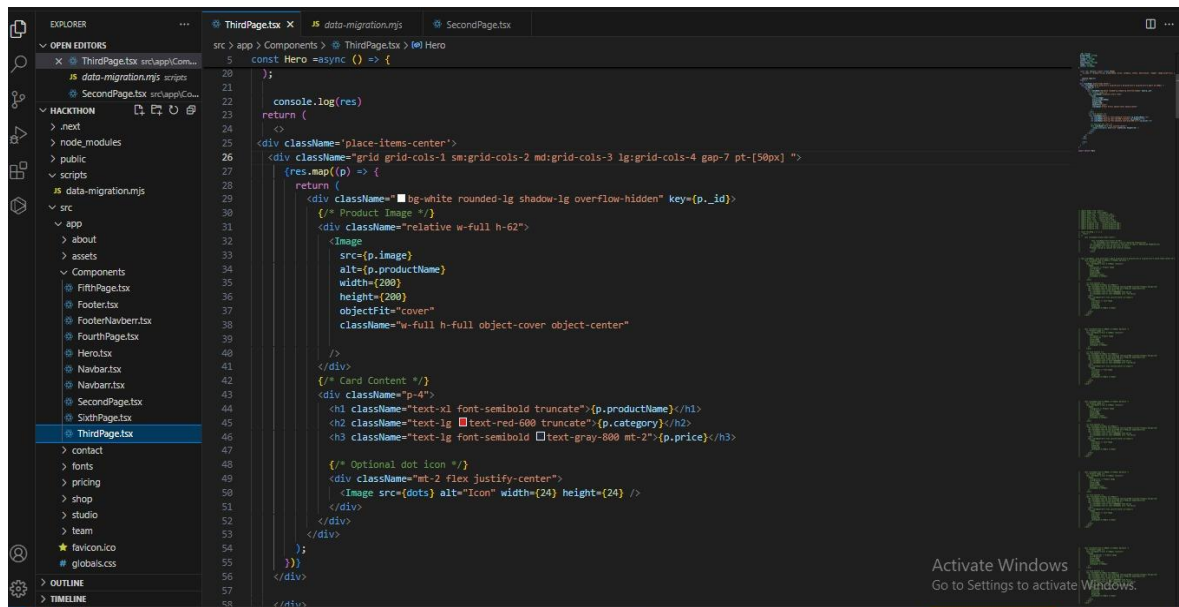

6. Importing Data into Sanity

With everything setup, I executed the migration script to send data into sanity by running the following command:
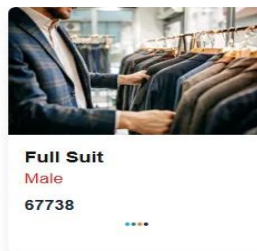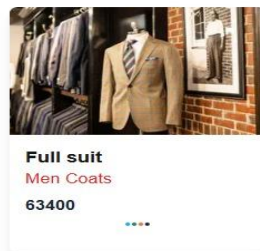
npm run migrate

Sanity Studio (Structure)

Default + Create

Structure Vision Schedules Tasks

Content

Product

Product

Search list

Men Cloths — Male
Shirts & Pents — Male
Upper Coats — Men Coats
white T-Shirt — Boys T-Shirt
Men Shirts — Boy's Shirt
The T-Shirt — Child Shirts
Boys Kurta — Men Full Suit

Men Cloths

Product

Men Cloths

Product Name

Men Cloths

Category

Male

Price

11895

Activate Windows
Go to Settings to activate Windows.

Published 15 hr. ago   Publish

VS Code — page.tsx

EXPLORER

HACKTHONNN
- src
  - app
    - Component
    - fonts
    - Help
    - Joinus
    - ProductDetail
    - Products
      - page.tsx
    - Signin
    - studio
    - favicon.ico
    - globals.css
    - layout.tsx
    - not-found.js
    - page.tsx
  - sanity
  - .env.local
  - .eslintrc.json
  - .gitignore
  - next-env.d.ts
  - next.config.ts
  - package-lock.json
  - package.json
  - postcss.config.mjs
  - README.md

OUTLINE
TIMELINE

src > app > Products > page.tsx > [@] Products > [@] data

```tsx
import React from 'react'
import { client } from '@/sanity/lib/client'
import Image from "next/image"

const Products =async () => {

interface Idata{
  productName:string,
  price:number,
  _id:number,
  inventory:number,
  category:string,
  status:string,
  colors:string,
  description:string,
  image:string
}
  const data:Idata[] = await client.fetch(`*[_type=="product"]{
productName,
  price,
  _id,
  inventory,
  category,
  status,
  colors,
  description,
  "image":image.asset->url,

}`)
  console.log(data)
  return (
```
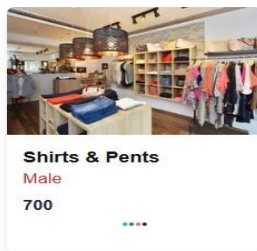
7. Displaying data in frontend

After fetching data from Sanity, I mapped it and printed it in cards, and successfully my data was displayed on the frontend.

Conclusion

This process successfully automated the migration of data from an external API to Sanity CMS. The Key steps included:

1. Fetching Data from External APIs
2. Comparing API data with Sanity Schema
3. Data Migration Script
4. Setting Up The Environment
5. Modifying package.json
6. Importing Data into Sanity
7. **Displaying data in frontend**

API Understanding	✔
Schema Validation	✔
Data Migration	✔
API Integration in Next.js	✔
Submission Preparation	✔

Prepared by: Qaimudin Khuwaja