

## **Day 3 - API Integration Report - Medical Scrubs and Stethoscopes Marketplace**

### **Introduction:**

As part of integrating APIs into our e-commerce website for medical scrubs and stethoscopes, we have used Sanity CMS, Visual Studio Code (VSCode), and the slugify command for generating clean and SEO-friendly URLs.

### **1. API Integration Process:**

- We set up Sanity as the headless CMS for managing content related to medical scrubs and stethoscopes, such as product details, images, and pricing.
- The API endpoints were created in Sanity, and I integrated them into the e-commerce website template using JavaScript and asynchronous functions to fetch product data.
- Once the data was retrieved from Sanity's API, it was displayed on the product pages of the website, ensuring smooth user experience and product accessibility.

### **2. Adjustments Made to Schemas:**

- Adjustments to the Sanity schemas were made to fit the structure of our e-commerce platform. This involved adding specific fields for medical scrubs and stethoscopes, such as product name, price, description, image, and category.
- To ensure that product URLs were clean and SEO-friendly, I used the **slugify** command to create unique slugs for each product, ensuring proper indexing by search engines.

### **3. Migration Steps and Tools Used:**

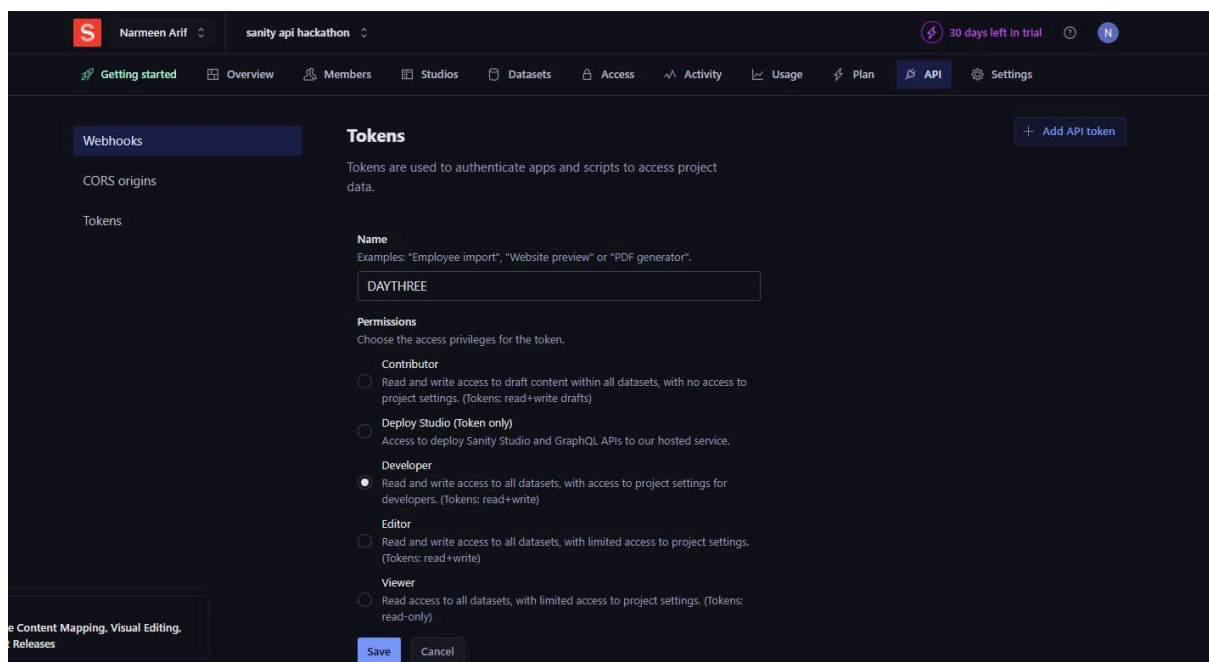
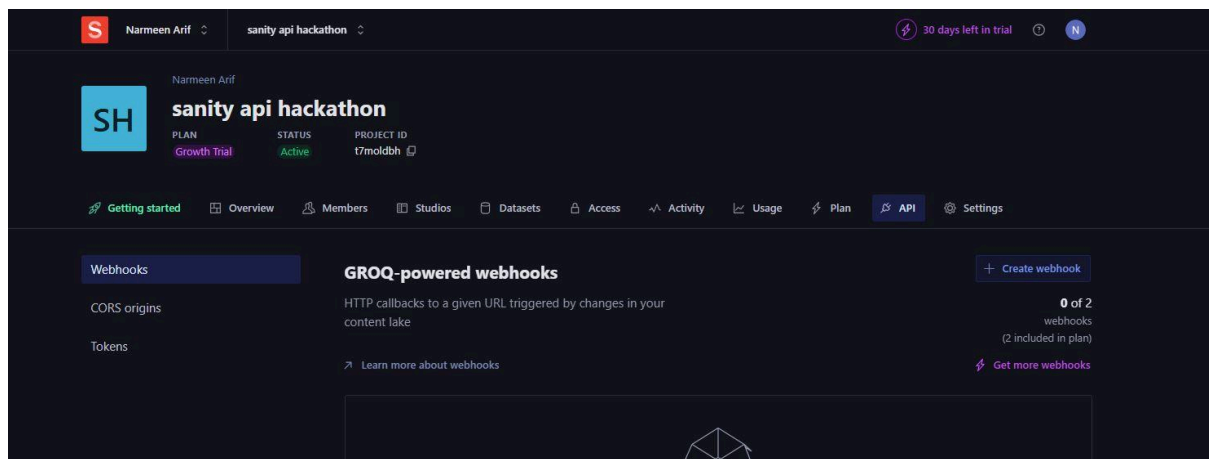
- I migrated the product data into Sanity CMS, ensuring that all relevant information was organized correctly in the backend.
- Using VSCode, I updated the frontend of the e-commerce site to fetch and display this data via the integrated API.
- The **slugify** command was used to generate slugs for each product, ensuring that URLs were both clean and descriptive.

## Conclusion:

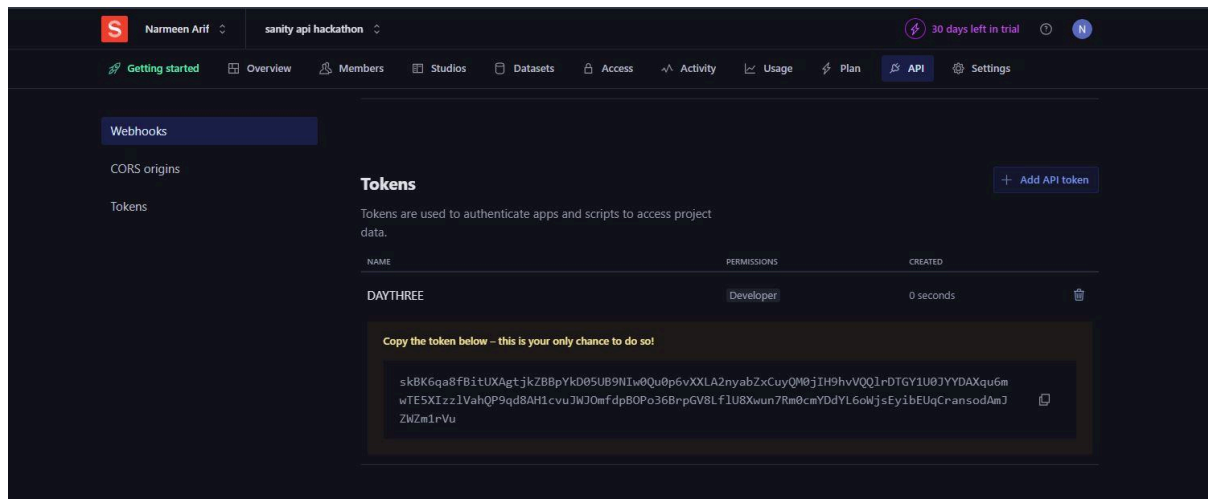
This was my first experience with API integration, and I learned a lot about retrieving and displaying dynamic data from a CMS. The integration of Sanity API has allowed the e-commerce site to be more scalable and maintainable. The use of the slugify command improved the product URLs, making them more search engine optimized. Overall, the integration was successful, and I look forward to enhancing the site with additional features.

---

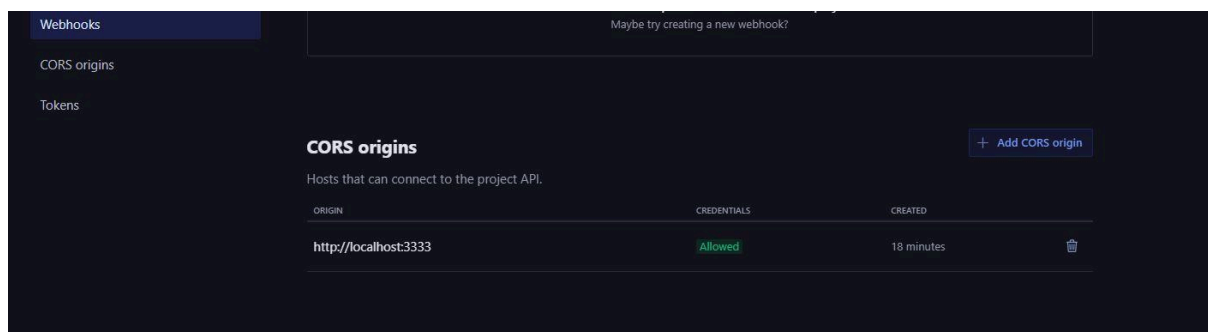
## Some screenshots of the work are below:



# SANITY TOKEN GENERATE



The screenshot shows the Sanity API Tokens page. The left sidebar has a menu with 'Webhooks', 'CORS origins', and 'Tokens'. The 'Tokens' section is active. The main content area is titled 'Tokens' and includes a '+ Add API token' button. Below the title, there is a table with columns 'NAME', 'PERMISSIONS', and 'CREATED'. The table contains one entry: 'DAYTHREE' with permissions 'Developer' and '0 seconds'. A warning box states: 'Copy the token below - this is your only chance to do so!'. The token is displayed as a long alphanumeric string: 'skBK6qa8fBitUXAgTjkZBBpYkD05UB9NIw0Qu0p6vXXLA2nyabZxCuyQM0jIH9hvVQQ1rDTGY1U0JYYDAXqu6m wTE5XIzz1VahQP9qd8AH1cvuJWJ0mfdpBOPo36B8rpGV8Lf1U8Xwun7Rm0cmYDdYL6oWjsEyibEUqCransodAmJ ZWZm1rVu'.



The screenshot shows the Sanity API CORS origins page. The left sidebar has a menu with 'Webhooks', 'CORS origins', and 'Tokens'. The 'CORS origins' section is active. The main content area is titled 'CORS origins' and includes a '+ Add CORS origin' button. Below the title, there is a table with columns 'ORIGIN', 'CREDENTIALS', and 'CREATED'. The table contains one entry: 'http://localhost:3333' with credentials 'Allowed' and '18 minutes'.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.5371]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ARIF\Desktop\7dayhackathon>code .

C:\Users\ARIF\Desktop\7dayhackathon>npm create sanity@latest
You are logged in as syedanarameen20@gmail.com using Google
Fetching existing projects

? Create a new project or select an existing one Create new project
? Your project name: sanity api hackathon
Your content will be stored in a dataset that can be public or private, depending on
whether you want to query your content with or without authentication.
The default dataset configuration has a public dataset named "production".
? Use the default dataset configuration? Yes
? Creating dataset
? 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'
```

## SANITY INSTALLED

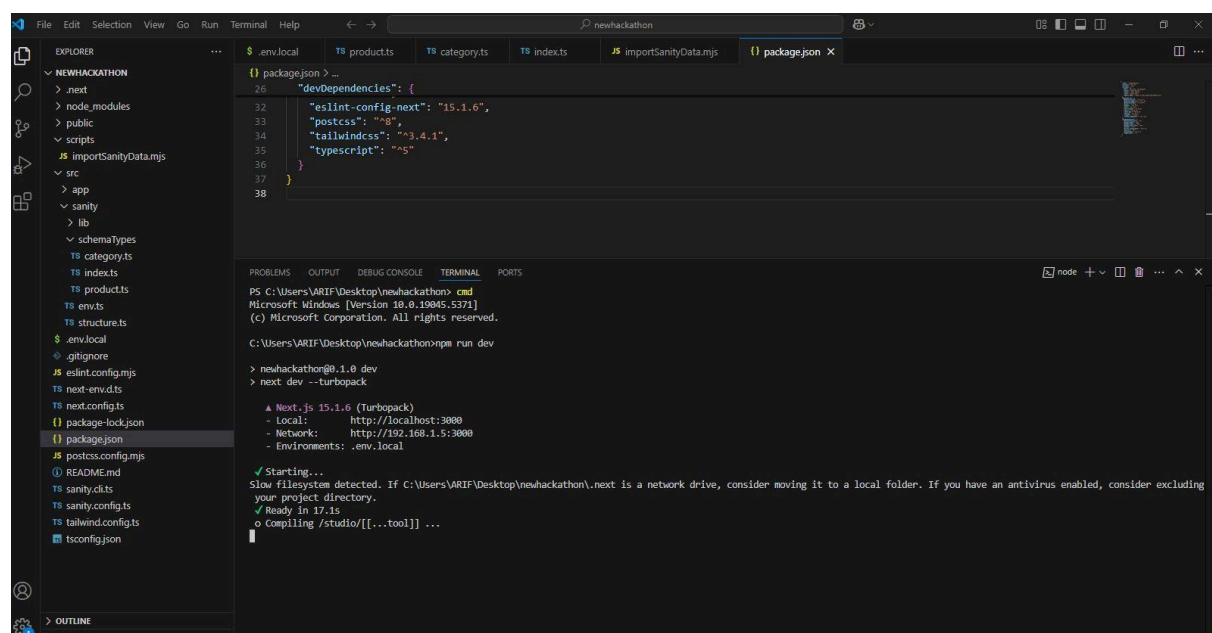
```
247 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

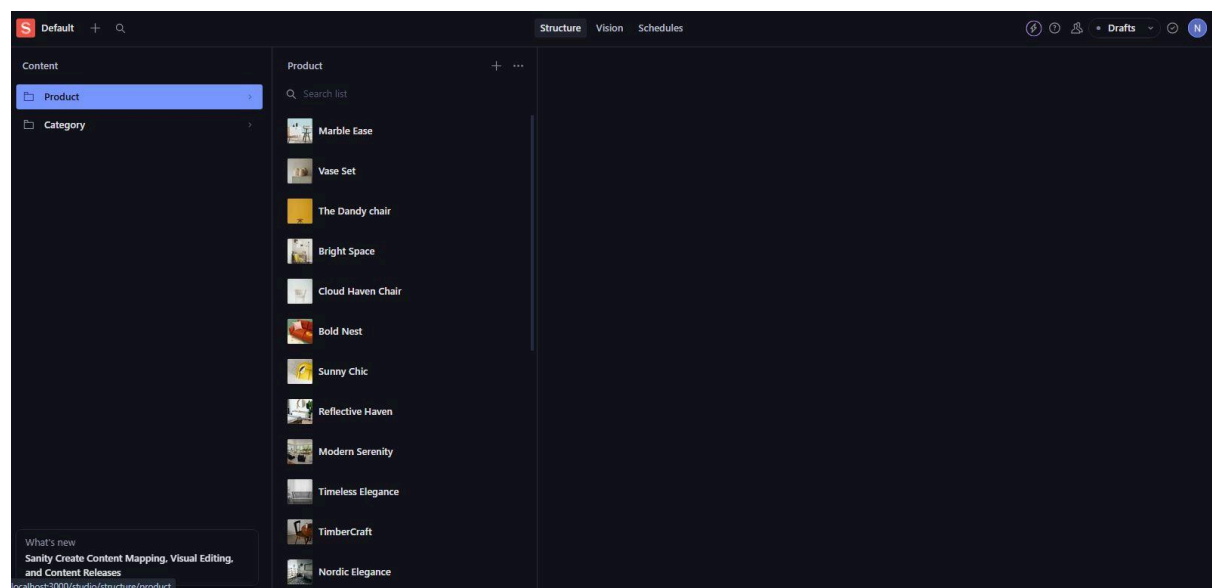
Success! Your Sanity configuration files has been added to this project

C:\Users\ARIF\Desktop\7dayhackathon>
```

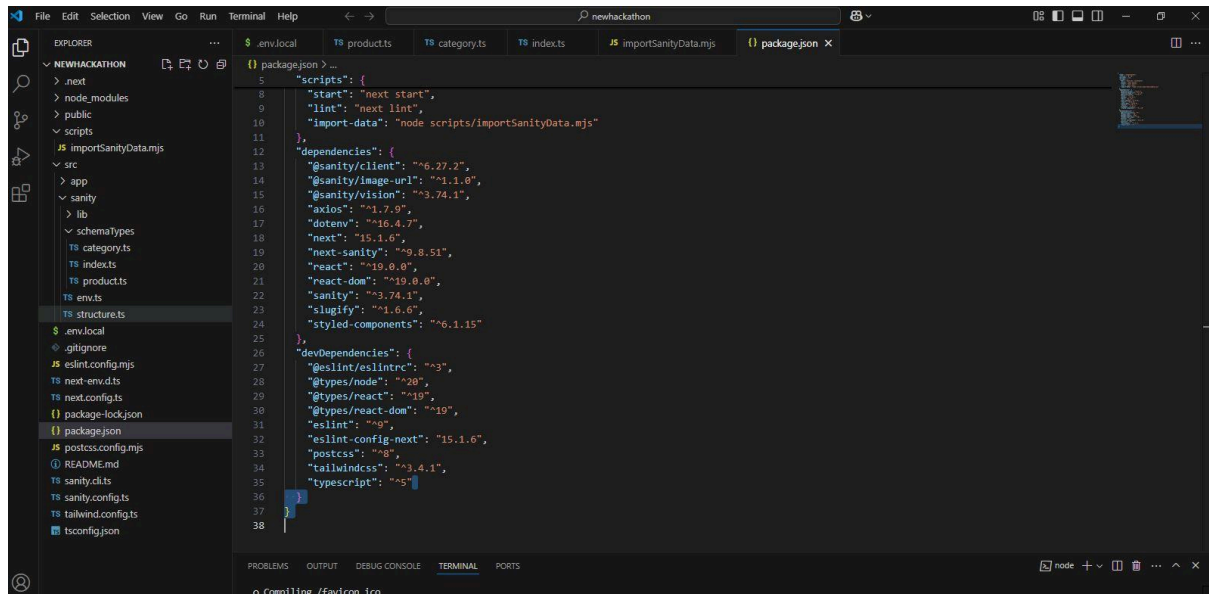
## FINAL COMMAND



## DATA FETCH

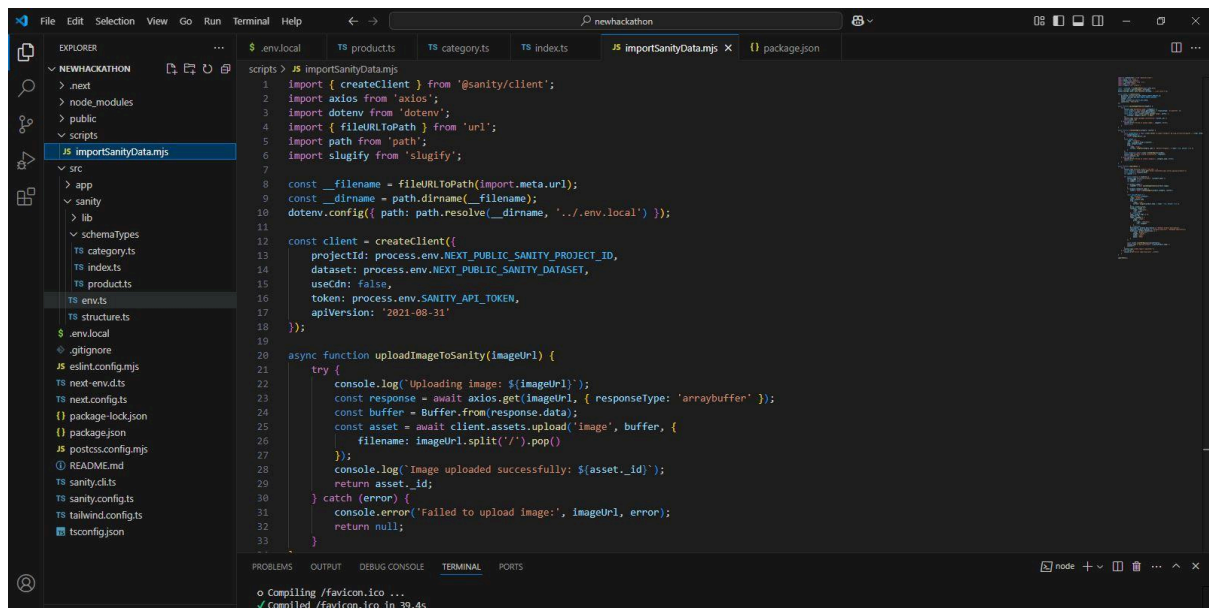


# PACKAGE.JSON



```
{} package.json > ...
5
8 "scripts": {
9   "start": "next start",
10  "lint": "next lint",
11  "import-data": "node scripts/importSanityData.mjs"
12 },
13 "dependencies": {
14   "@sanity/client": "^6.27.2",
15   "@sanity/image-url": "^1.1.0",
16   "@sanity/vision": "^3.74.1",
17   "axios": "^1.7.9",
18   "dotenv": "^16.4.7",
19   "next": "15.1.6",
20   "next-sanity": "^9.8.51",
21   "react": "19.0.0",
22   "react-dom": "19.0.0",
23   "sanity": "^3.74.1",
24   "slugify": "^1.6.6",
25   "styled-components": "^6.1.15"
26 },
27 "devDependencies": {
28   "@eslint/eslintrc": "^3",
29   "@types/node": "^20",
30   "@types/react": "19",
31   "@types/react-dom": "19",
32   "eslint": "8",
33   "eslint-config-next": "15.1.6",
34   "postcss": "8",
35   "tailwindcss": "3.4.1",
36   "typescript": "5"
37 }
38
```

# MIGRATION API



```
scripts > JS importSanityData.mjs
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 import slugify from 'slugify';
7
8 const __filename = fileURLToPath(import.meta.url);
9 const __dirname = path.dirname(__filename);
10 dotenv.config({ path: path.resolve(__dirname, '../.env.local') });
11
12 const client = createClient({
13   projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
14   dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
15   useCdn: false,
16   token: process.env.SANITY_API_TOKEN,
17   apiVersion: '2021-08-31'
18 });
19
20
21 async function uploadImageToSanity(imageUrl) {
22   try {
23     console.log('Uploading image: ${imageUrl}');
24     const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
25     const buffer = Buffer.from(response.data);
26     const asset = await client.assets.upload('image', buffer, {
27       filename: imageUrl.split('/').pop()
28     });
29     console.log('Image uploaded successfully: ${asset._id}');
30     return asset._id;
31   } catch (error) {
32     console.error('Failed to upload image:', imageUrl, error);
33     return null;
34   }
35 }
36
```