

DAY 3

NAME: ALI HASAN

ROLL NO: 0044644

TIMING: SAT 9-12AM

DAY 03 – API INTEGRATION REPORT OF HECTO

Introduction

In this project, we were tasked with importing data from API into Sanity using migration scripts. We successfully completed this task and then fetched the data to display in our user interface.

Steps Involved

1. API Integration
 - We received API from which we needed to import data into Sanity. The data from the API was crucial for our project.
2. Schema Creation
 - We created a schema for the product and categories. This schema defines how the data will be structured in Sanity.
 - The schema is essential for organizing and querying the data effectively.
3. Migration Scripts
 - We developed migration scripts to import data from the APIs into Sanity.
 - These scripts ensure that the data is correctly formatted and inserted into the Sanity database.
4. Adding Migration to package.json
 - We added the migration script to the package.json file. This allows us to run the migration using npm commands.
 - Specifically, we used `npm run migrate` to execute the migration script.
5. Importing Data into Sanity
 - After running the migration script, the data from the APIs was successfully imported into Sanity.
 - This step ensured that our Sanity database was populated with the necessary data.
6. Fetching Data with GROQ
 - We used GROQ (Sanity's query language) to fetch the imported data from Sanity. GROQ queries allow us to specify exactly what data we need, making it efficient for displaying in our UI.
7. Displaying Data in UI
 - Finally, we displayed the fetched data in our user interface.
 - This completes the full cycle of importing data into Sanity and then using it in our application.
 -

Conclusion

This project involved integrating data from external API into Sanity using migration scripts and then querying that data for display in our UI. The process was successful, and we were able to efficiently manage and display the data as needed. This report outlines the steps you took in a clear and simple manner, making it easy to understand the process involved.

```
C:\Users\Dell\Downloads\Figma-to-code-Hackathon-Project>npm run import-data

> ali-hecto-commerce@0.1.0 import-data
> node scripts/importSanityData.mjs

Fetching Product Data From API ...
Processing Item: Tribù Elio Chair
Uploading Image : https://next-ecommerce-template-4.vercel.app/product/Chair (5).png
Image Uploaded Successfully : image-9967004db2d5e1235db7525ed72aaa32bc4f95a7-720x522-png
Uploading Chair - Tribù Elio Chair to Sanity !
Uploaded Successfully: 8rHgms8cRnJSVtFj3yeyjq
-----

Processing Item: Armchair Tortuga
Uploading Image : https://next-ecommerce-template-4.vercel.app/product/Chair (35).png
Image Uploaded Successfully : image-d6aed5e3bb1c45542716cb31036c740f1113625e-1398x1194-png
Uploading Chair - Armchair Tortuga to Sanity !
Uploaded Successfully: 2CvAB00fgTAGzXYTMB01Io
-----

Processing Item: Uchiwa Quilted Lounge Chair
Uploading Image : https://next-ecommerce-template-4.vercel.app/product/Chair (6).png
Image Uploaded Successfully : image-2e1f600d17d0908be26d3768263fa9b3d7fe4ead-402x402-png
Uploading Chair - Uchiwa Quilted Lounge Chair to Sanity !
Uploaded Successfully: 2CvAB00fgTAGzXYTMB0mKG
-----

Processing Item: Stylish Golden Metal Legs Mint Blue Fabric Velvet Sofa Leisure Armchair
Uploading Image : https://next-ecommerce-template-4.vercel.app/product/Chair (3).png
Image Uploaded Successfully : image-7a9f255386e1ee47303b1a578d6b7d43029b469b-1018x1100-png
Uploading Sofa - Stylish Golden Metal Legs Mint Blue Fabric Velvet Sofa Leisure Armchair to Sanity !
Uploaded Successfully: 8rHgms8cRnJSVtFj3yf4kE
-----

Processing Item: Hans Wegner Style Three-Legged Shell Chair
```

← → ↺ local:3333/structure/product

☆ 📦 🔄 ⚠️ Relaunch to update

template-4 + Create 🔍


Structure Vision Schedules


Content


Product


Product


Search list


Luxury Flower Shell Sofa Chair


Cantilever Chair


Nordic Net Red Chair


Sobuy Blue Folding Chair Wooden P...


Varmora Plastic Chair Solid

Matilda Velvet Chair - Pink

Diondre Chair - Tuft Button - Acrylic ...

Leisure Sofa Chair Set

Liberty Wood 63' Floating Entertain...

Futuristic Sleek Modern Chair

What's new

Sanity Create Content Mapping, Visual Editing, and Content Releases

Latest Products

New Arrival

Best Seller

Featured

Special Offer



[Comfort Handy Craft](#)

\$42.00 ~~\$65.00~~



[Comfort Handy Craft](#)

\$42.00 ~~\$65.00~~



[Comfort Handy Craft](#)

\$42.00 ~~\$65.00~~



[Comfort Handy Craft](#)

\$42.00 ~~\$65.00~~



[Comfort Handy Craft](#)

\$42.00 ~~\$65.00~~



[Comfort Handy Craft](#)

\$42.00 ~~\$65.00~~

EXPLORER

PROJ HACKATHON

next

documentation

public

scripts

src

components

pages

services

styles

latest.tsx

template-4

.gitignore

my-preset.js

next-env.d.ts

package-lock.json

package.json

postcss.config.js

tailwind.config.js

tsconfig.json

yarn.lock

JS importSanityData.mjs U X

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 dotenv.config();

7 const __filename = fileURLToPath(import.meta.url);

8 const __dirname = path.dirname(__filename);

9 dotenv.config({ path: path.resolve(__dirname, '../.env') });

10

11 const client = createClient({

12 projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,

13 dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,

14 token: process.env.SANITY_API_TOKEN,

15 apiVersion: '2025-01-15',

16 useCdn: false,

17 });

18

19 async function uploadImageToSanity(imageUrl) {

20 try {

21 console.log(`Uploading Image : \${imageUrl}`);

22 const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });

23 const buffer = Buffer.from(response.data);

24 const asset = await client.assets.upload('image', buffer, {

25 filename: imageUrl.split('/').pop(),

26 });

27 console.log(`Image Uploaded Successfully : \${asset._id}`);

28 return asset._id;

29 }

30 catch (error) {

31 console.error('Failed to Upload Image:', imageUrl, error);

32 return null;

33 }

Tabnine | Edit | Test | Explain | Document | Codeium: Refactor | Explain | Generate JSDoc | X

EXPLORER

PROJ HACKATHON

next

documentation

public

scripts

src

components

pages

services

styles

latest.tsx

template-4

.gitignore

my-preset.js

next-env.d.ts

package-lock.json

package.json

postcss.config.js

tailwind.config.js

tsconfig.json

yarn.lock

JS importSanityData.mjs U

scripts > JS importSanityData.mjs > ...

```
36 async function importData() {
37   try {
38     console.log('Fetching Product Data From API ...');
39
40     const response = await axios.get("https://next-ecommerce-template-4.vercel.app/api/product");
41     const products = response.data.products;
42
43     for (const item of products) {
44       console.log('Processing Item: ${item.name}');
45
46       let imageRef = null;
47       if (item.imagePath) {
48         imageRef = await uploadImageToSanity(item.imagePath);
49       }
50
51       const sanityItem = {
52         _type: 'product',
53         name: item.name,
54         category: item.category || null,
55         price: item.price,
56         description: item.description || '',
57         discountPercentage: item.discountPercentage || 0,
58         stockLevel: item.stockLevel || 0,
59         isFeaturedProduct: item.isFeaturedProduct,
60         image: imageRef
61         ? {
62           _type: 'image',
63           asset: {
64             _type: 'reference',
65             _ref: imageRef,
66           },
67         }
68         : undefined;
```

TS products.ts X

TS products.ts

```
1 export default {
2   name: 'product',
3   type: 'document',
4   title: 'Product',
5   fields: [
6     {
7       name: 'name',
8       type: 'string',
9       title: 'Name',
10      validation: (Rule: any) => Rule.required().error('Name is required'),
11    },
12    {
13      name: 'image',
14      type: 'image',
15      title: 'Image',
16      options: {
17        hotspot: true,
18      },
19      description: 'Upload an image of the product.',
20    },
21    {
22      name: 'price',
23      type: 'string',
24      title: 'Price',
25      validation: (Rule: any) => Rule.required().error('Price is required'),
26    },
27    {
28      name: 'description',
29      type: 'text',
30      title: 'Description',
31      validation: (Rule: any) =>
32        Rule.max(150).warning('Keep the description under 150 characters.'),
```