

# Day 3 - API Integration Report - Comforty

## 1. Introduction

Today, I focused on integrating APIs and migrating data into Sanity CMS to set up the marketplace backend. It was a hands-on experience where I learned how to validate schemas, migrate data, and work with APIs effectively. This process gave me practical insights into building a scalable solution while handling real-world data challenges.

## 2. Objectives

The main objectives for today were:

1. Integrating APIs into the Next.js project to fetch and display data.
2. Migrating data into Sanity CMS to ensure the backend is functional and aligned with the marketplace requirements.
3. Validating and refining schemas to maintain compatibility with the data sources.

## 3. API Integration Process

For API integration, I used two main API endpoints: /products and /categories. I used the Sanity API to display product data, ensuring a smooth integration between the external APIs and the Sanity CMS.

## 4. Adjustments Made to Schemas

To ensure compatibility with the API data structure, I made several adjustments to the schemas in Sanity CMS:

1. Field Mapping: Renamed and aligned fields such as product\_title to name, and product\_price to price, ensuring consistency with the API response.
2. Data Types: Updated field types where necessary, such as ensuring price was stored as a number and categories were linked as references.
3. Relationships: Defined relationships between products and categories to maintain proper data hierarchy and enable efficient querying.

## 5. Data Migration Steps

The data migration process was involved the following steps:

1. Sanity Setup: I installed Sanity and configured the environment variables NEXT\_PUBLIC\_SANITY\_PROJECT\_ID, NEXT\_PUBLIC\_SANITY\_DATASET, and NEXT\_PUBLIC\_SANITY\_AUTH\_TOKEN to connect my project to Sanity CMS.
2. Schema Creation: I created schemas for products and categories and imported them into schemaTypes to define the structure of my data.
3. Migration Script: In the scripts folder, I implemented a data migration script to automate the transfer of data from the provided REST APIs into my Sanity dataset.
4. Environment Setup: I installed the dotenv package using the command npm install dotenv to manage environment variables securely.

## 6. API calls.

Here are the screenshots of my API calls:

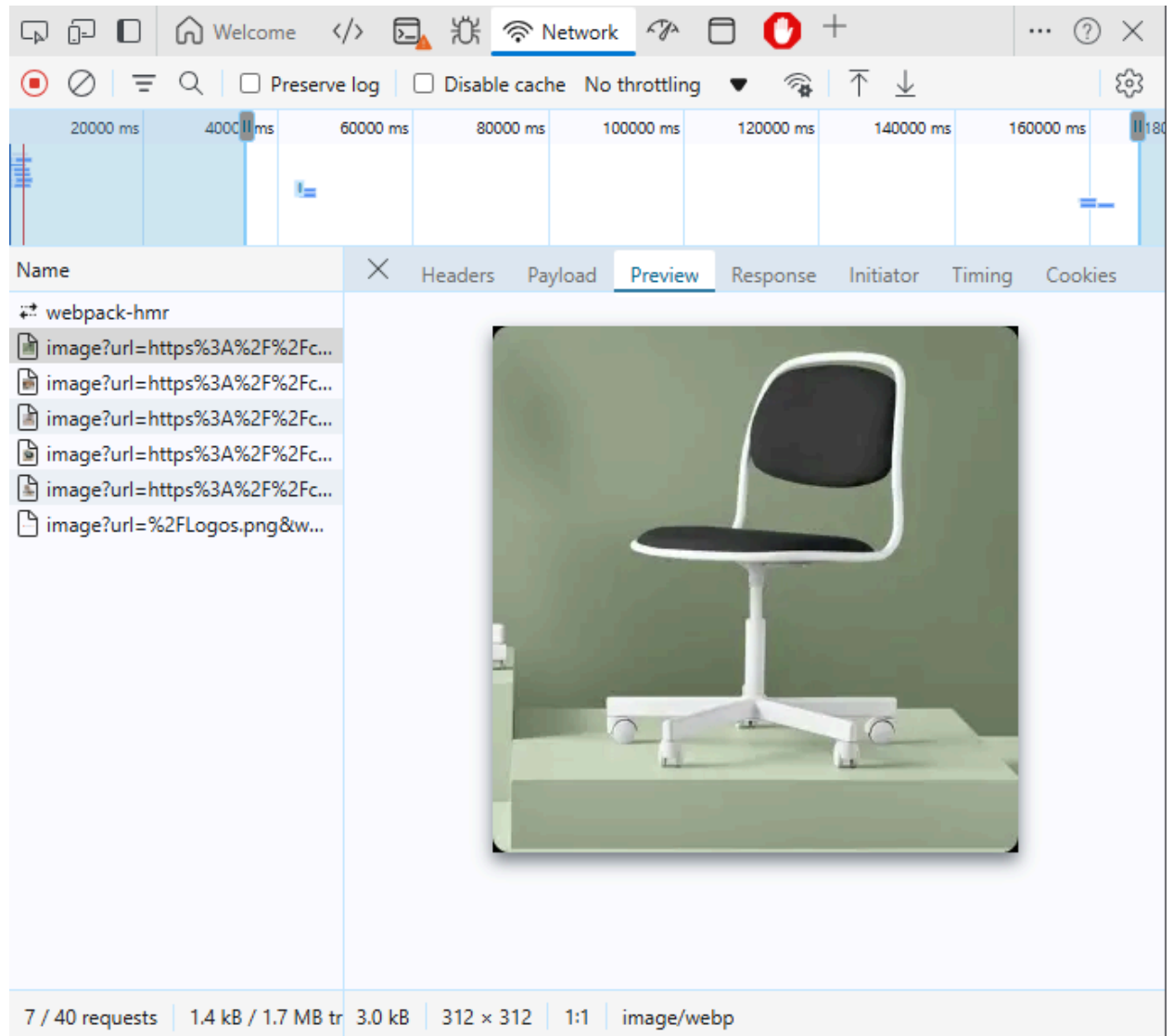
**Headers:**

The screenshot shows the Chrome DevTools Network tab with a list of requests. The selected request is an image request to a Sanity image endpoint. The 'Headers' tab is active, showing the following details:

Name	Value
Request URL:	http://localhost:3000/_next/image?url=https%3A%2F%2Fcdn.sanity.io%2Fimages%2Fnracvwl9%2Fproduction%2F81a5b7de166f930870a82f8f3e661b38a70de9f4-312x312.png&w=640&q=75
Request Method:	GET
Status Code:	304 Not Modified
Remote Address:	[::1]:3000
Referrer Policy:	strict-origin-when-cross-origin
Response Headers:	<p>Cache-Control: public, max-age=0, must-revalidate</p> <p>Connection: keep-alive</p> <p>Date: Sun, 19 Jan 2025 14:22:36 GMT</p> <p>Etag: 8fw5-KW159LpftdqygeboxqbzFmKjLyd0OCqRW2KjOQ=</p> <p>Keep-Alive: timeout=5</p> <p>Vary: Accept</p>
Request Headers:	

At the bottom of the network tab, it shows '7 / 40 requests' and '1.4 kB / 1.7 MB tr'.

## Preview:



## 7. Data Displayed on Frontend.

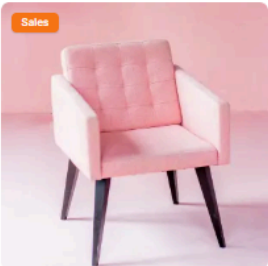
Here are the screenshots of my data display:

Home Page:




Featured Products

Sale



Rose Luxe Armchair  
\$20 ~~\$30~~



Citrus Edge  
\$20


New



Library Stool Chair  
\$20

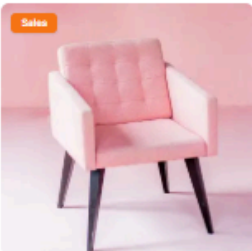
Citrus Edge  
\$20

Our Products




SleekSpin  
\$20

Sale



Rose Luxe Armchair  
\$20 ~~\$30~~



Scandi Dip Set  
\$40



SleekSpin  
\$20



Gray Elegance  
\$8



Rose Luxe Armchair  
\$20 ~~\$30~~



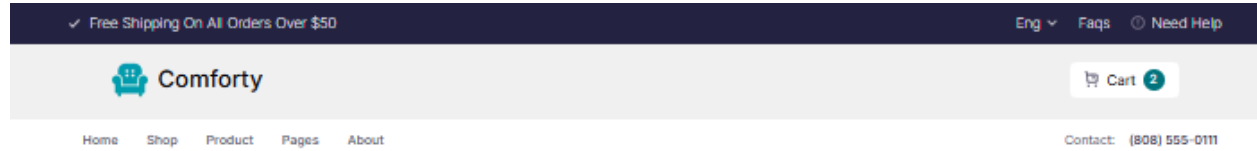
Modern Cozy  
\$20

New

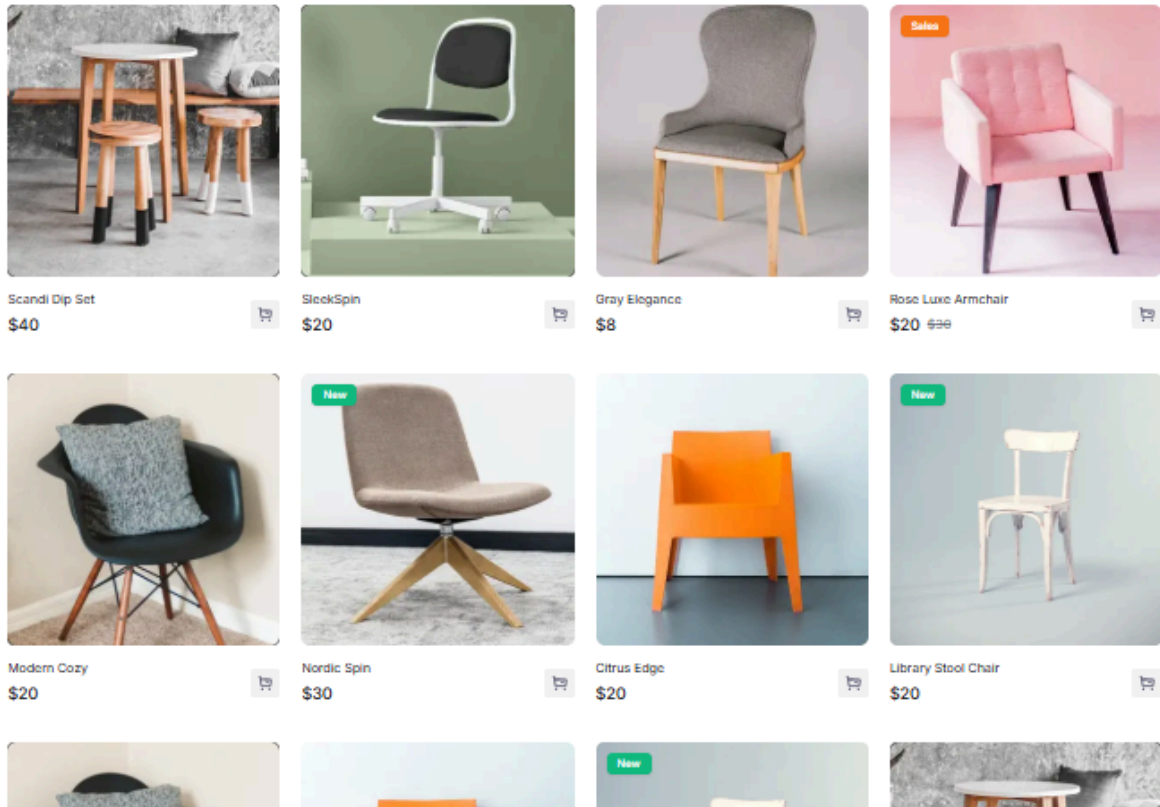


Nordic Spin  
\$30

## Products Page:



### All Products



## 8. Populated Sanity CMS fields

Here are the screenshots of my Sanity CMS populated fields:

Categories:

Comforty

Cart 2

HomeShopProductPagesAbout

Contact: (808) 555-0111

DefaultCreate

StructureVisionSchedules

Content

Products

Categories

Categories

Search list

Desk Chair

Wooden Chair


Wing Chair

Wooden Chair

Categories

Wooden Chair

Category Image



What's new

Sanity Create Content Mapping, Visual Editing, and Content Releases

Products:

Comforty

Cart 2

HomeShopProductPagesAbout

Contact: (808) 555-0111

DefaultCreate

StructureVisionSchedules

Content

Products

Categories

Products

Search list

Ivory Charm

Gray Elegance

Nordic Spin

Scandi Dip Set

Modern Cozy

Library Stool Chair

SleekSpin

Rose Luxe Armchair

Citrus Edge

Scandi Dip Set

Rose Luxe Armchair

Products

Rose Luxe Armchair

Product Title

Rose Luxe Armchair

Price

20

Price without Discount

30

Badge

Sales

What's new

Sanity Create Content Mapping, Visual Editing, and Content Releases

## 9. Code Snippets

API Integration :

Snippet 1

```
src > sanity > lib > TS queries.ts > ...
1  import { defineQuery } from "next-sanity";
2
3  export const allproducts = defineQuery(`
4    *[_type == "products"][2..13]{
5      _id,
6      title,
7      price,
8      priceWithoutDiscount,
9      badge,
10     "imageUrl": image.asset->url,
11     tags
12   }
13 `);
14 export const productdetails = defineQuery(`
15   *[_type == "products"]{
16     _id,
17     title,
18     price,
19     priceWithoutDiscount,
20     badge,
21     "imageUrl": image.asset->url,
22     "category": category->title,
23     description,
24     inventory,
25     tags
26   }
27 `);
```

## Snippet 2

```
src > sanity > lib > TS queries.ts > ...  
28 export const featuredproducts = defineQuery(`  
29   *[_type == "products" && "featured" in tags][1..4]{  
30     _id,  
31     title,  
32     price,  
33     priceWithoutDiscount,  
34     badge,  
35     "imageUrl": image.asset->url,  
36     tags  
37   }  
38 `);  
39 export const Ourproducts = defineQuery(`  
40   *[_type == "products"][0..7]{  
41     _id,  
42     title,  
43     price,  
44     priceWithoutDiscount,  
45     badge,  
46     "imageUrl": image.asset->url,  
47     tags  
48   }  
49 `);  
50 export const categories = defineQuery(`  
51   *[_type == "products"][2..13]{  
52     _id,  
53     title,  
54     price,  
55     priceWithoutDiscount,  
56     badge,  
57     "imageUrl": image.asset->url,  
58     tags  
59   }
```



## 2. Migration Script:

```
scripts > $S migrate.mjs > ...
1 // Import environment variables from .env.local
2 import "dotenv/config";
3 // Import the Sanity client to interact with the Sanity backend
4 import { createClient } from "sanity/client";
5 // Load required environment variables
6 const {
7   NEXT_PUBLIC_SANITY_PROJECT_ID, // Sanity project ID
8   NEXT_PUBLIC_SANITY_DATASET, // Sanity dataset (e.g., "production")
9   NEXT_PUBLIC_SANITY_AUTH_TOKEN, // Sanity API token
10   BASE_URL = "https://gala-hackathon-template-08.vercel.app", // API base URL
11   // for products and categories
12 } = process.env;
13
14 // Check if the required environment variables are provided
15 if (!NEXT_PUBLIC_SANITY_PROJECT_ID || !NEXT_PUBLIC_SANITY_AUTH_TOKEN) {
16   console.error(
17     "Missing required environment variables. Please check your .env.local file."
18   );
19   process.exit(1); // Stop execution if variables are missing
20 }
21
22 // Create a Sanity client instance to interact with the target Sanity dataset
23 const targetClient = createClient({
24   projectId: NEXT_PUBLIC_SANITY_PROJECT_ID, // Your Sanity project ID
25   dataset: NEXT_PUBLIC_SANITY_DATASET || "production", // Default to
26   // "production" if not set
27   useCdn: false, // Disable CDN for real-time updates
28   apiVersion: "2023-01-01", // Sanity API version
29   token: NEXT_PUBLIC_SANITY_AUTH_TOKEN, // API token for authentication
30 });
31
32 // Function to upload an image to Sanity
33 async function uploadImageToSanity(imageUrl) {
34   try {
35     // Fetch the image from the provided URL
36     const response = await fetch(imageUrl);
37     if (!response.ok) throw new Error("Failed to fetch image: ${imageUrl}");
38
39     // Convert the image to a buffer (binary format)
40     const buffer = await response.arrayBuffer();
41
42     // Upload the image to Sanity and get its asset ID
43     const uploadedAsset = await targetClient.assets.upload(
44       "image",
45       Buffer.from(buffer),
46       {
47         filename: imageUrl.split("/").pop(), // Use the file name from the URL
48       }
49     );
50
51     return uploadedAsset.id; // Return the asset ID
52   } catch (error) {
53     console.error("Error uploading image:", error.message);
54     return null; // Return null if the upload fails
55   }
56 }
57
58 // Main function to migrate data from REST API to Sanity
59 async function migrateData() {
60   console.log("Starting data migration...");
61
62   try {
63     // Fetch categories from the REST API
64     const categoriesResponse = await fetch(`${BASE_URL}/api/categories`);
65     if (!categoriesResponse.ok) throw new Error("Failed to fetch categories.");
66     const categoriesData = await categoriesResponse.json(); // Parse response
67     // to JSON
68
69     // Fetch products from the REST API
70     const productsResponse = await fetch(`${BASE_URL}/api/products`);
71     if (!productsResponse.ok) throw new Error("Failed to fetch products.");
72     const productsData = await productsResponse.json(); // Parse response to
73     // JSON
74
75     const categoryIdMap = {}; // Map to store migrated category IDs
76
77     // Migrate categories
78     for (const category of categoriesData) {
79       console.log("Migrating category: ${category.title}");
80       const imageId = await uploadImageToSanity(category.imageUrl); // Upload
81       // category image
82
83       // Prepare the new category object
84       const newCategory = {
85         _id: category.id, // Use the same ID for reference mapping
86         _type: "categories",
87         title: category.title,
88         image: imageId
89         ? { _type: "image", asset: { _ref: imageId } }
90         : undefined, // Add image if uploaded
91       };
92
93       // Save the category to Sanity
94       const result = await targetClient.createOrReplace(newCategory);
95       categoryIdMap[category.id] = result._id; // Store the new category ID
96       console.log("Migrated category: ${category.title} (ID: ${result._id})");
97     }
98
99     // Migrate products
100     for (const product of productsData) {
101       console.log("Migrating product: ${product.title}");
102       const imageId = await uploadImageToSanity(product.imageUrl); // Upload
103       // product image
104
105       // Prepare the new product object
106       const newProduct = {
107         _type: "products",
108         title: product.title,
109         price: product.price,
110         priceWithoutDiscount: product.priceWithoutDiscount,
111         badge: product.badge,
112         image: imageId
113         ? { _type: "image", asset: { _ref: imageId } }
114         : undefined, // Add image if uploaded
115         category: {
116           _type: "reference",
117           _ref: categoryIdMap[product.category_id], // Use the migrated
118           // category ID
119         },
120         description: product.description,
121         inventory: product.inventory,
122         tags: product.tags,
123       };
124
125       // Save the product to Sanity
126       const result = await targetClient.create(newProduct);
127       console.log("Migrated product: ${product.title} (ID: ${result._id})");
128     }
129
130     console.log("Data migration completed successfully!");
131   } catch (error) {
132     console.error("Error during migration:", error.message);
133     process.exit(1); // Stop execution if an error occurs
134   }
135 }
136
137 // Start the migration process
138 migrateData();
```

## 10. Day 03 Checklist

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