Day 3 Hackathon Task Report

API Integration & Data Migration

Table of Contents

Project Overview

On Day 3 of the hackathon, I focused on integrating an external API with my existing Sanity project to populate and manage data dynamically. This task involved extending my previously created website template, defining schemas in Sanity, and writing a custom script to fetch data from an external API and import it into my Sanity dataset.

Steps Completed

1. Website Template Setup

I started with a pre-existing website template that I had created earlier. This template already had the required structure for a front-end website built using **Next.js** and was integrated with **Sanity** for CMS functionalities.

2. Sanity Project Setup

I ensured that my Sanity project was installed and configured in the template:

3. Schema Definition and Customization

In the Sanity project, I created and customized the following schemas:

- **Product Schema**
  Defined the schema for storing product details, including fields such as:

```
export default
 { name: 'product',
 title: 'Product',
 type: 'document',
fields: [
{ name: 'title', type: 'string', title: 'Product Title' },
```

{ name: 'description', type: 'text', title: 'Description' },

{ name: 'price', type: 'number', title: 'Price' },

{ name: 'image', type: 'image', title: 'Product Image' },

{ name: 'category', type: 'reference', to: [{ type: 'category' }],

title: 'Category' },], };

Category Schema

```
export default {
  name: 'category',
  title: 'Category',
  type: 'document',
  fields: [
    { name: 'title', type: 'string', title: 'Category Title' },
    { name: 'description', type: 'text', title: 'Description' },
  ],
};
```

Both schemas were added to the schemas/ directory and included in the schemaTypes array of the Sanity configuration.

4. Created a scripts Folder

To manage automation scripts, I created a scripts/ folder in the root directory of my project. This folder is intended for storing and executing scripts related to data migration or integration.

5. Created importSanityData.mjs Script

Within the scripts folder, I created a file named importSanityData.mjs. This script was responsible for fetching data from an external API and importing it into the Sanity dataset.

6. Integrated External API

In the importSanityData.mjs file, I wrote the logic for fetching data from an external API (Template 2) and inserting it into the Sanity project:

- **API Integration and Data Fetching** Used the node-fetch library to fetch data from the external API:
- **7. Executed the Script**
- To populate the Sanity dataset with the external API data, I ran the script using Node.js:

```
node scripts/importSanityData.mjs
```

This successfully fetched data from the external API and imported it into the Sanity project.

Outcomes

1. Successfully fetched data from an external API (Template 2).
2. Imported the fetched data into the Sanity dataset using a custom script.
3. Verified that the data appeared correctly in the Sanity Studio dashboard, mapped to the appropriate product and category schemas.

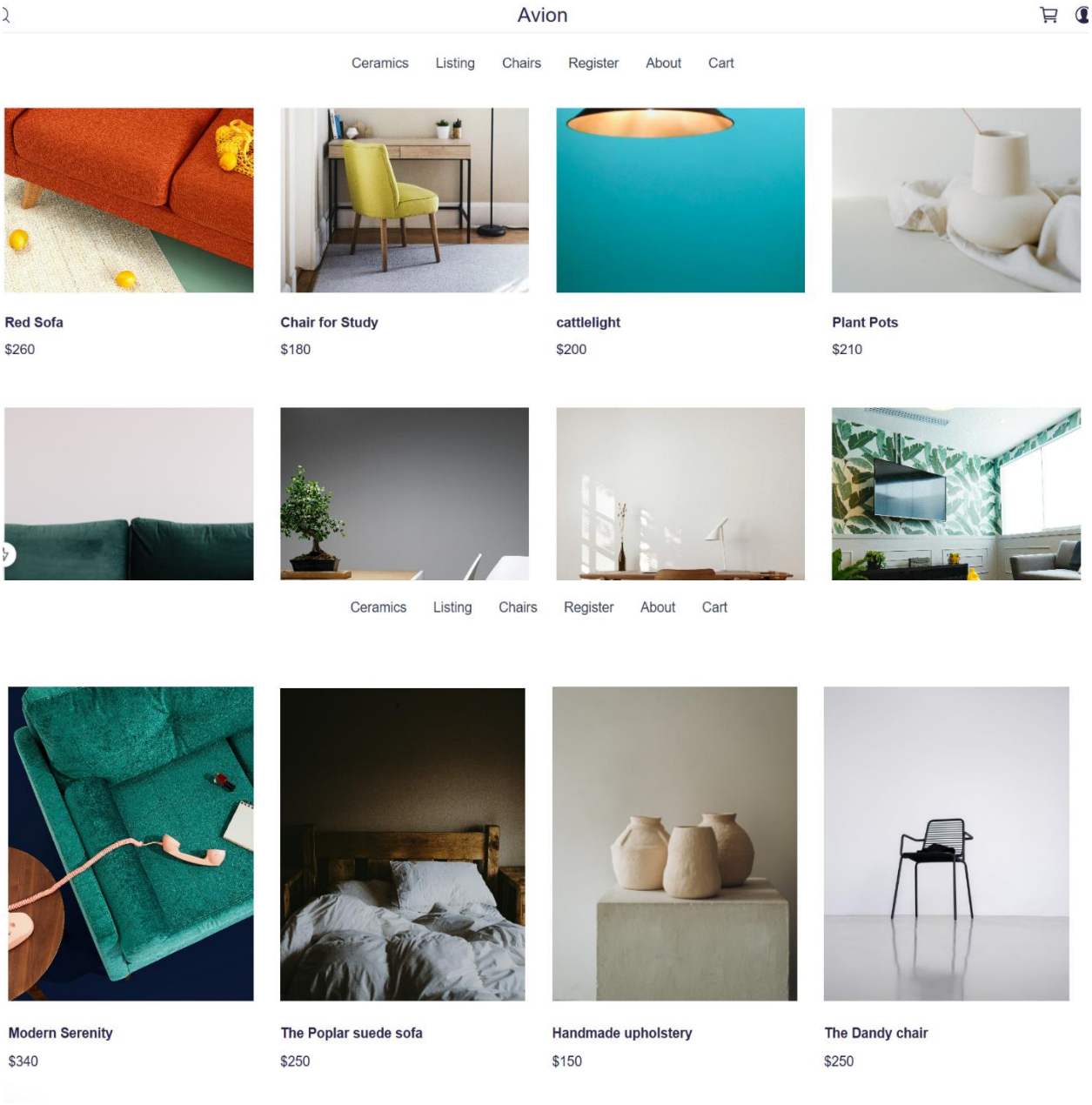
Learnings

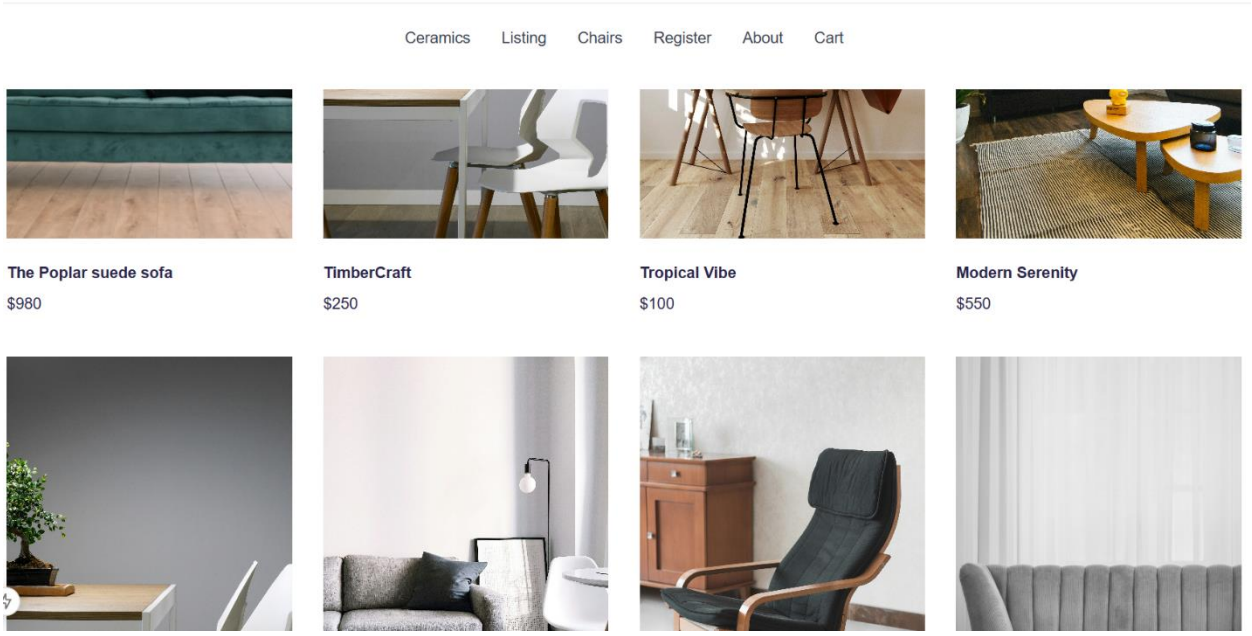
- Writing automation scripts in **Node.js** is an efficient way to integrate external data sources with a CMS like Sanity.
- Proper schema design is critical for structuring imported data and ensuring compatibility with the front-end application.

Next Steps

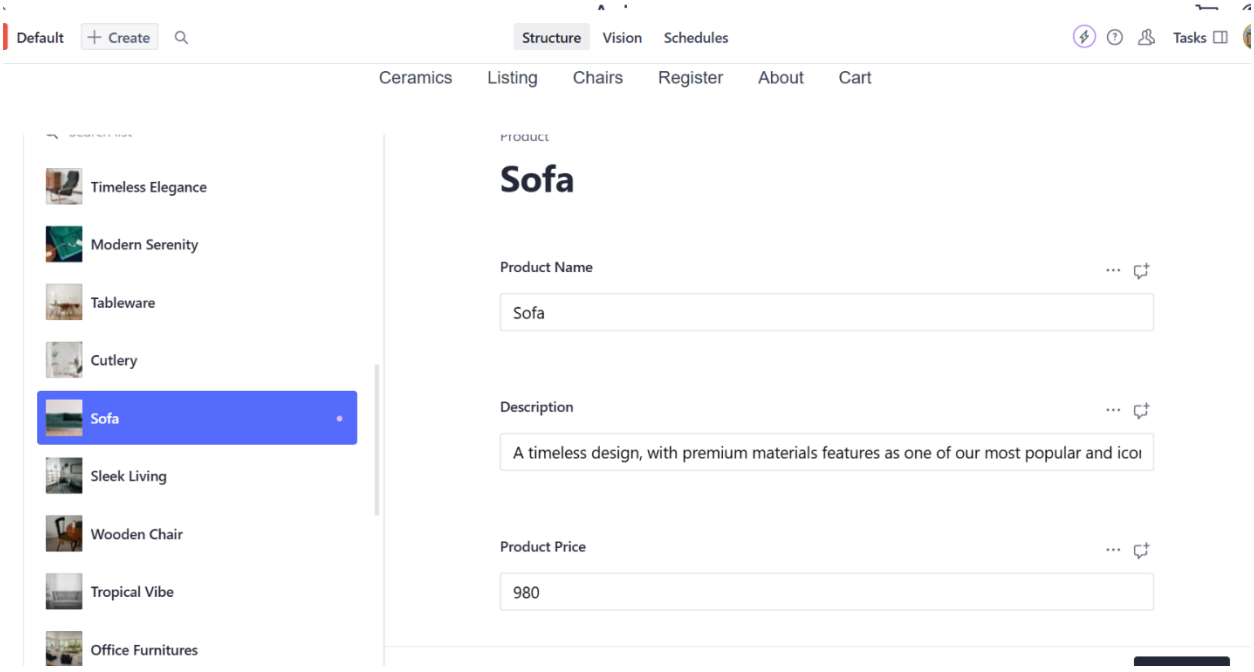
- Use the imported data in the front-end application to dynamically render product listings.
- Optimize the API integration process for handling large datasets.
- Add error handling and logging in the import script for better reliability.

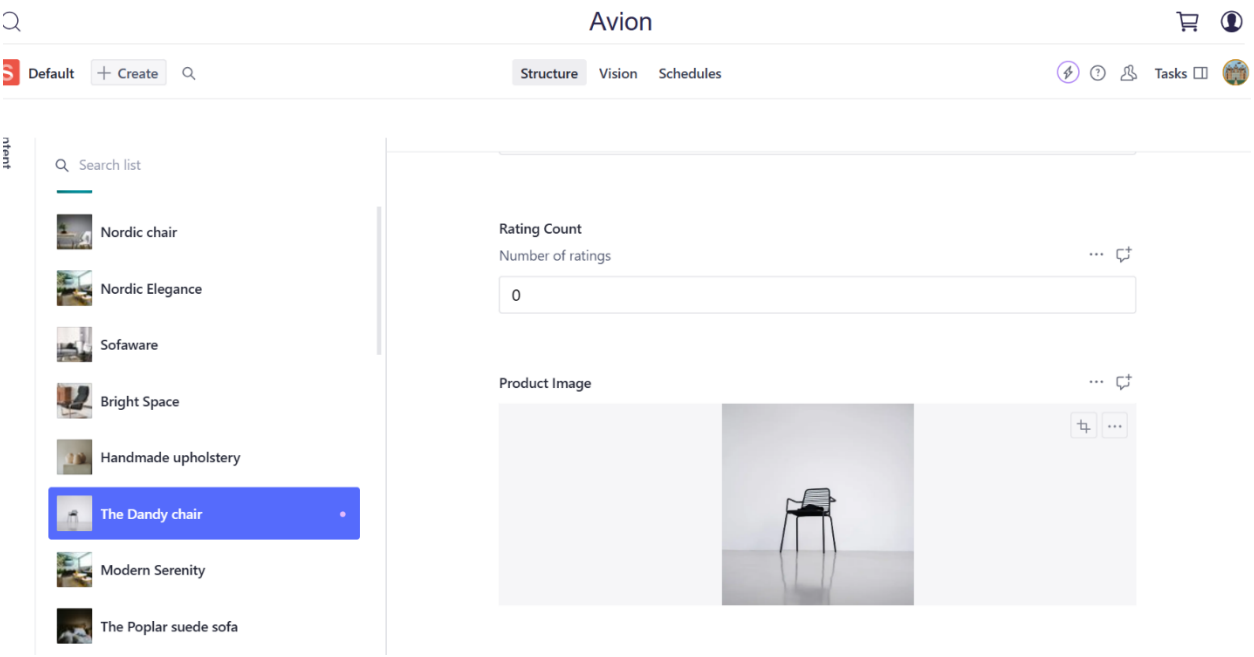
Screenshots of data displayed on frontend



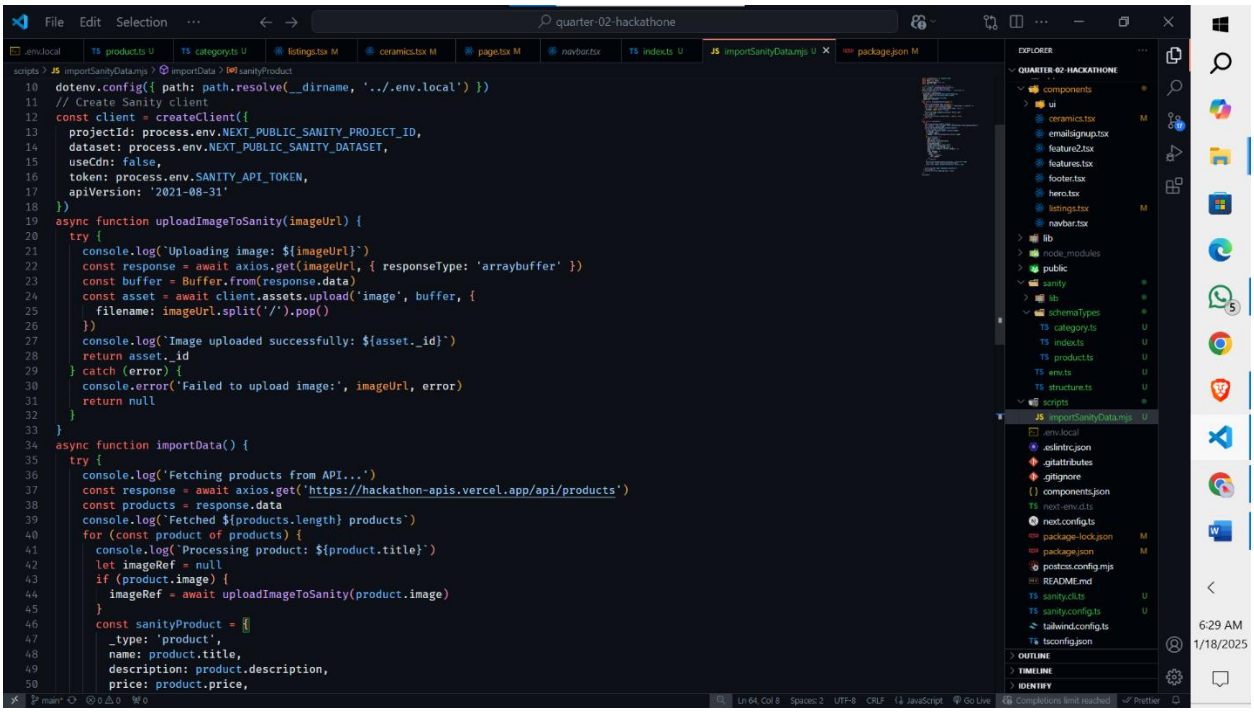


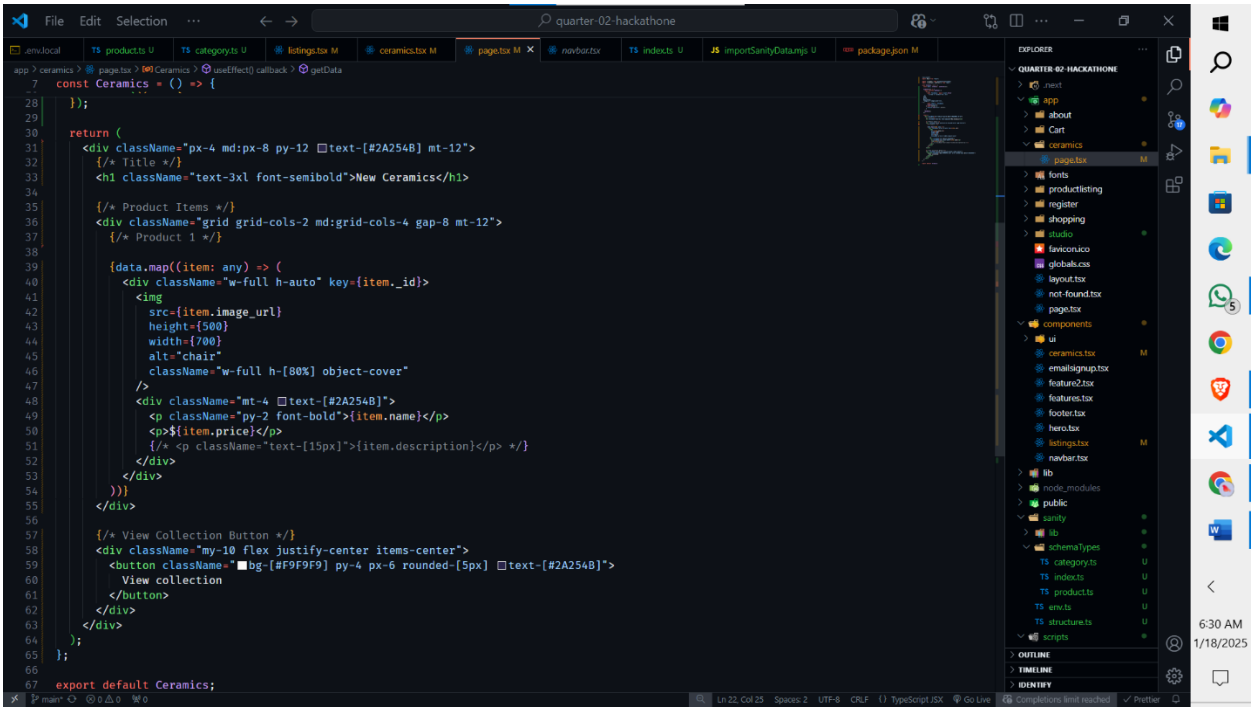
Screenshots of populated sanity CMS fields





Screenshots of API Calls & code snippets for API Integration and migration scripts




File Edit Selection
quarter-02-hackathone
page.tsx M
const Ceramics = () => {
return (
<h1 className="text-3xl font-semibold">New Ceramics</h1>
{/* Product Items */}
{/* Product 1 */}
{data.map((item: any) => (
${item.price}
View collection
export default Ceramics;
EXPLORER
QUARTER-02-HACKATHONE
app
about
Cart
ceramics
page.tsx
fonts
productlisting
register
shopping
studio
components
ui
lib
public
sanity
schemaTypes
OUTLINE
TIMELINE
IDENTIFY
6:30 AM
1/18/2025

The end