DAY 3 - API INTEGRATION REPORT - FURNITURE MARKETPLACE

Prepared by: Shama Farooq
Date: [18th Jan 2024]

---

1. Introduction

This report outlines the work completed on Day 3 of the hackathon for the **Furniture Marketplace** project. The main objective of this phase was to integrate the Sanity CMS API for migrating product data, as well as to adjust the schema to organize the product information. The process includes setting up the API connection, creating the schema for the product page, and running the data migration script to populate the local database with product data.

---

2. Sanity API Integration Process

The process of integrating the Sanity API was carried out to connect the Sanity CMS with the local project and facilitate the seamless migration of product data. The key steps are outlined below:

API Setup

- The **Product ID** and **API Token** were obtained from the Sanity CMS.
- These credentials were stored securely in environment variables to ensure confidentiality and security during the API connection.

Schema Configuration

To manage the product data efficiently, a single schema file was created:

1. **product.ts** – This schema file is designed to manage product-specific data such as product name, description, price, and images.

The schema ensures that product data is structured properly for easy querying and retrieval.

---

3. Data Migration

The goal of the data migration was to import product data from Sanity CMS to the local application. Below are the steps taken to ensure the migration was completed successfully:

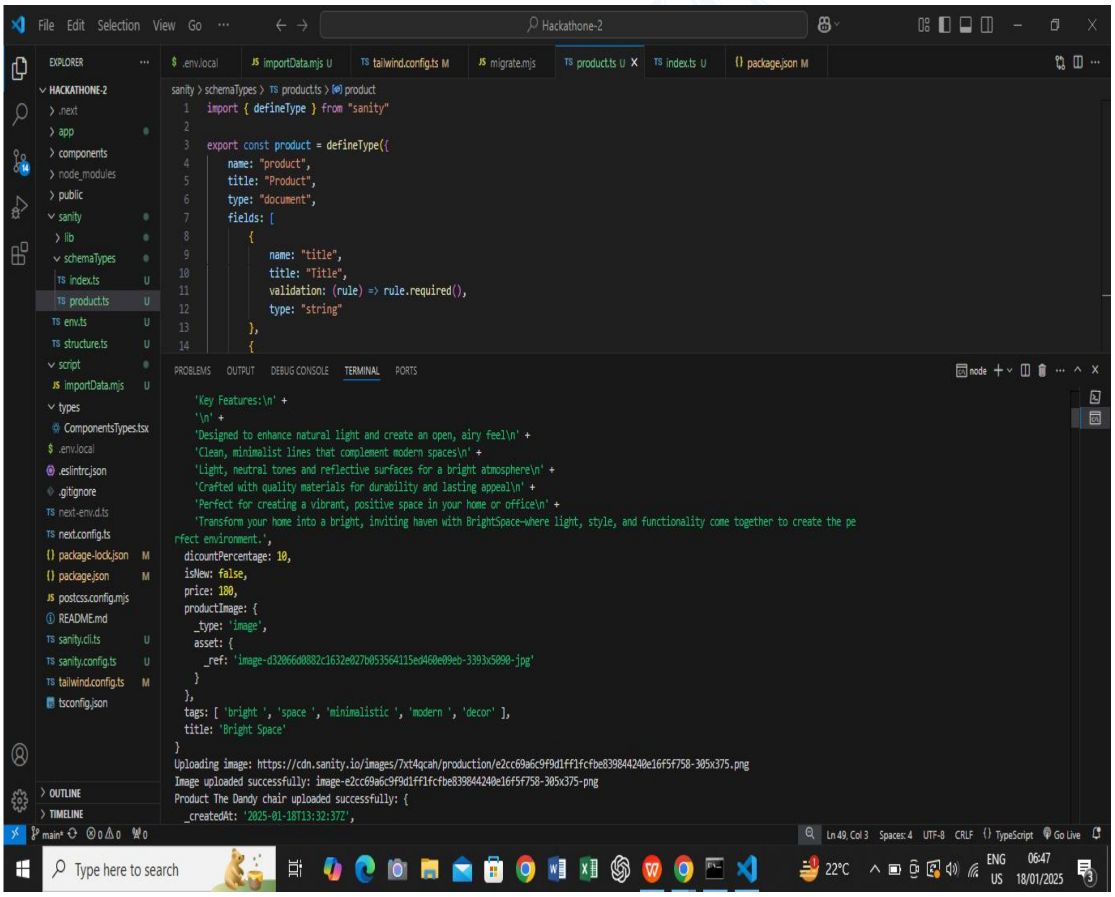
Script Creation

A folder named scripts was created at the root of the project directory to house the migration script. The script, named **importData.mjs**, was responsible for automating the process of importing product data into the local database.

Migration Execution

The data migration was triggered using the following terminal command:

npm run importData



```
import { defineType } from "sanity"

export const product = defineType({
    name: "product",
    title: "Product",
    type: "document",
    fields: [
        {
            name: "title",
            title: "Title",
            validation: (rule) => rule.required(),
            type: "string"
        },
        {
```

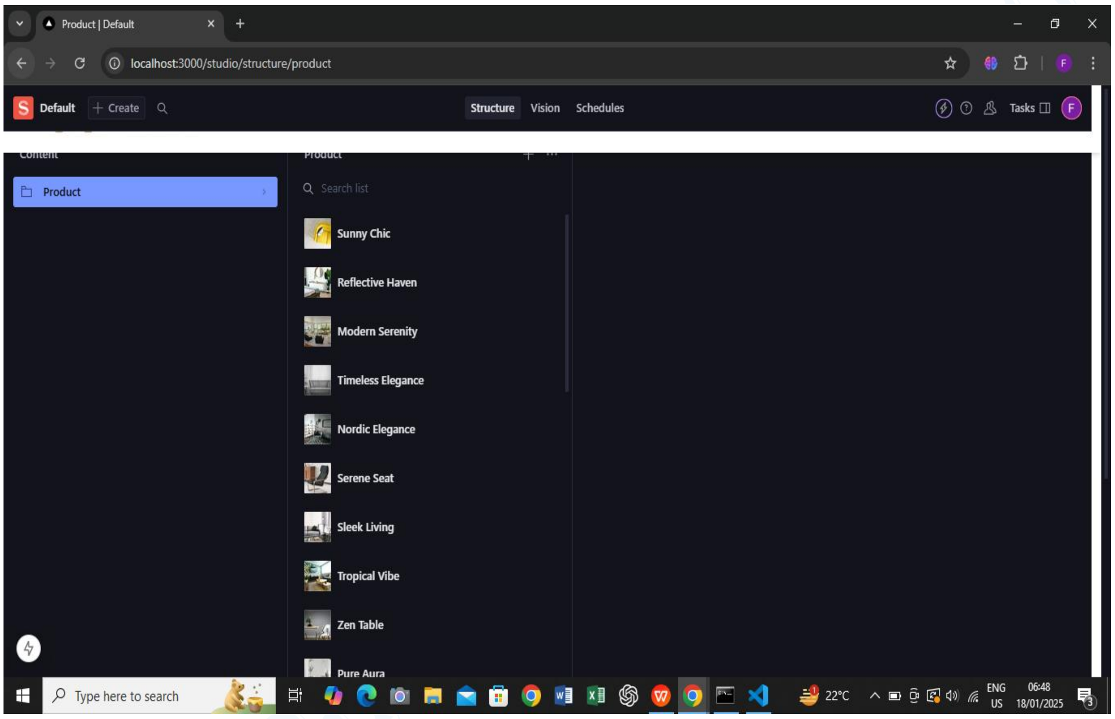
```
    'Key Features:\n' +
    '\n' +
    'Designed to enhance natural light and create an open, airy feel\n' +
    'Clean, minimalist lines that complement modern spaces\n' +
    'Light, neutral tones and reflective surfaces for a bright atmosphere\n' +
    'Crafted with quality materials for durability and lasting appeal\n' +
    'Perfect for creating a vibrant, positive space in your home or office\n' +
    'Transform your home into a bright, inviting haven with BrightSpace—where light, style, and functionality come together to create the pe
rfect environment.',
  dicountPercentage: 10,
  isNew: false,
  price: 180,
  productImage: {
    _type: 'image',
    asset: {
      _ref: 'image-d32066d0882c1632e027b053564115ed460e09eb-3393x5090-jpg'
    }
  },
  tags: [ 'bright ', 'space ', 'minimalistic ', 'modern ', 'decor' ],
  title: 'Bright Space'
}
Uploading image: https://cdn.sanity.io/images/7xt4qcah/production/e2cc69a6c9f9d1ff1fcfbe839844240e16f5f758-305x375.png
Image uploaded successfully: image-e2cc69a6c9f9d1ff1fcfbe839844240e16f5f758-305x375-png
Product The Dandy chair uploaded successfully: {
  _createdAt: '2025-01-18T13:32:37Z',
```

This command executed the **importData.mjs** script, which fetched the product data from Sanity CMS and populated the local database with the necessary information.

Migration Verification

Once the data was migrated, the following steps were taken to verify the success of the migration:

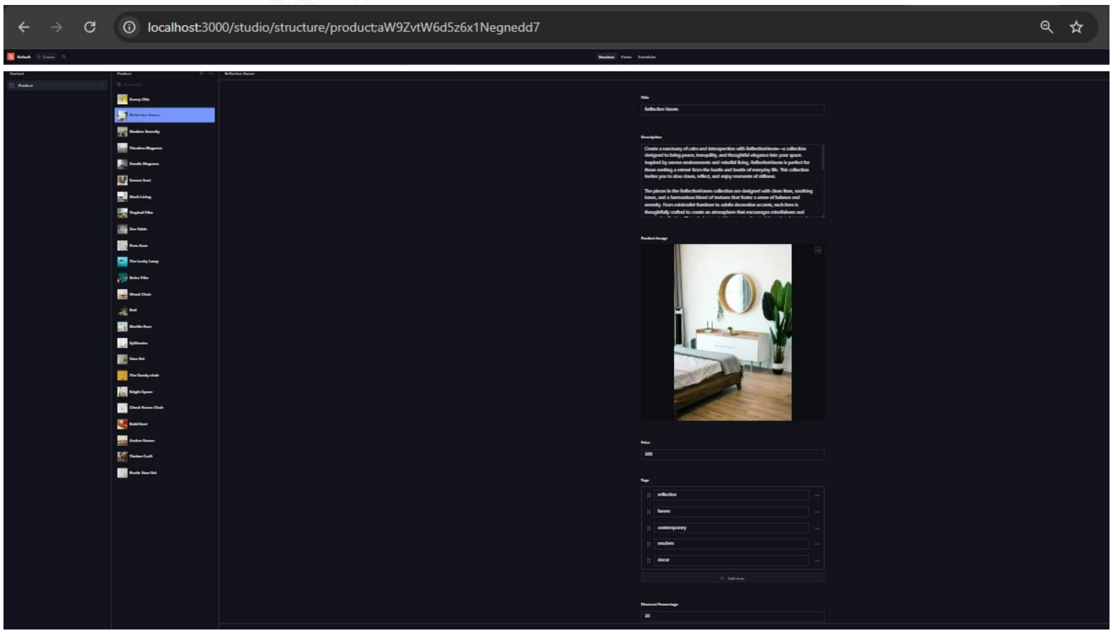
- The local project was run and the /studio/structure endpoint was accessed to confirm that the migrated data appeared in Sanity Studio.
- The same **Sanity login credentials** used for the CMS were employed to verify the accuracy of the data.



localhost:3000/studio/structure/product

Content
- Product

Product
- Sunny Chic
- Reflective Haven
- Modern Serenity
- Timeless Elegance
- Nordic Elegance
- Serene Seat
- Sleek Living
- Tropical Vibe
- Zen Table
- Pure Aura



localhost:3000/studio/structure/product;aW9ZvtW6d5z6x1Negnedd7

Title: Reflective Haven

Description: Create a sanctuary of calm and introspection with ReflectiveHaven—a collection designed to bring peace, tranquility, and thoughtful elegance into your space. Inspired by serene environments and mindful living, ReflectiveHaven is perfect for those seeking a retreat from the hustle and bustle of everyday life. This collection invites you to slow down, reflect, and enjoy moments of stillness.

The pieces in the ReflectiveHaven collection are designed with clean lines, soothing tones, and a harmonious blend of textures that foster a sense of balance and serenity. From minimalist furniture to subtle decorative accents, each item is thoughtfully crafted to create an atmosphere that encourages mindfulness and

Product Image

Price: 200

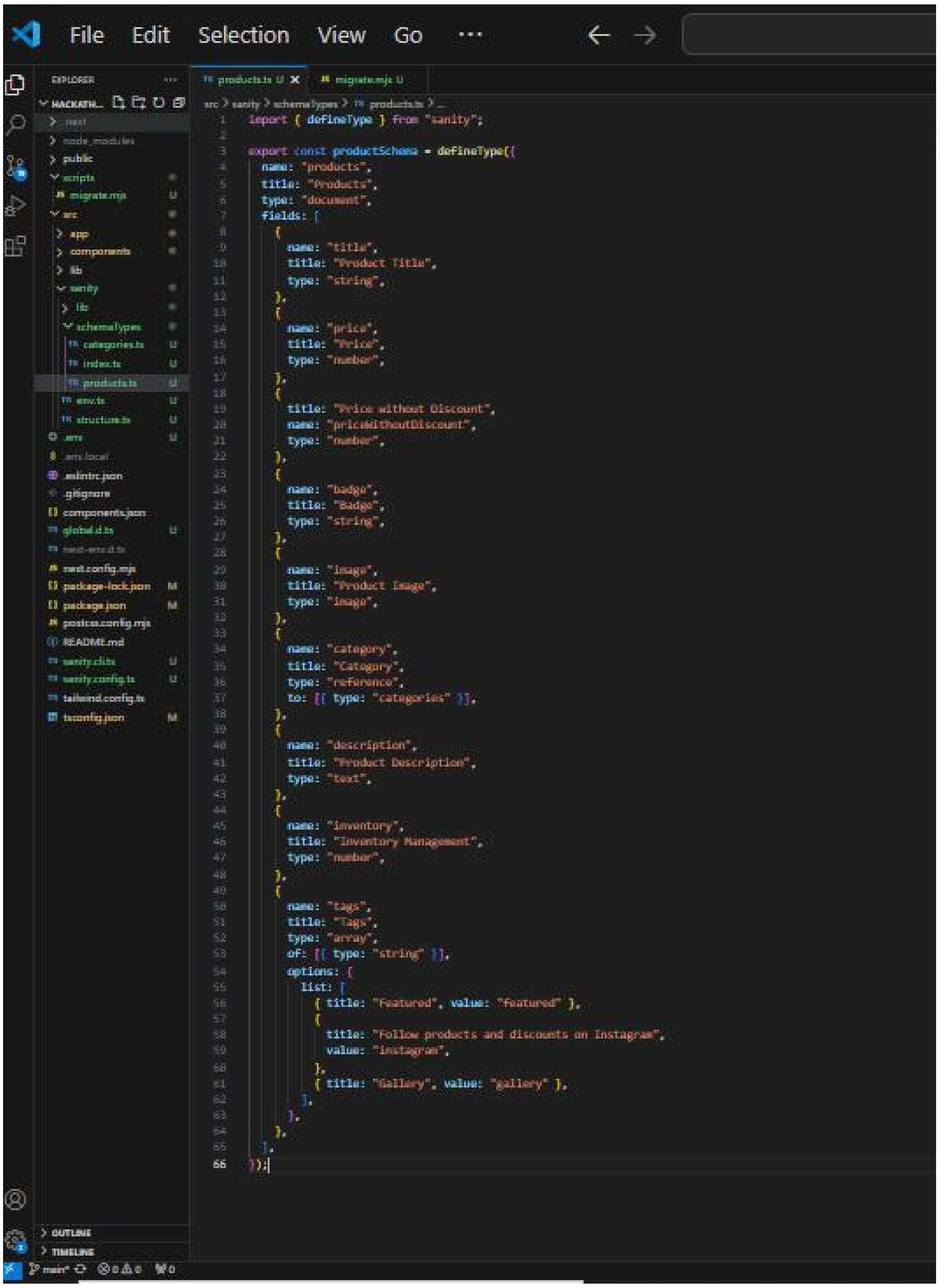
Tags: reflective, haven, contemporary, modern, decor

Discount Percentage: 20

4. Schema Adjustments

The product schema was created to ensure that the product data was properly structured and easy to work with in Sanity CMS.

Product Schema (product.ts)



```
import { defineType } from "sanity";

export const productSchema = defineType({
  name: "products",
  title: "Products",
  type: "document",
  fields: [
    {
      name: "title",
      title: "Product Title",
      type: "string",
    },
    {
      name: "price",
      title: "Price",
      type: "number",
    },
    {
      title: "Price without Discount",
      name: "priceWithoutDiscount",
      type: "number",
    },
    {
      name: "badge",
      title: "Badge",
      type: "string",
    },
    {
      name: "image",
      title: "Product Image",
      type: "image",
    },
    {
      name: "category",
      title: "Category",
      type: "reference",
      to: [{ type: "categories" }],
    },
    {
      name: "description",
      title: "Product Description",
      type: "text",
    },
    {
      name: "inventory",
      title: "Inventory Management",
      type: "number",
    },
    {
      name: "tags",
      title: "Tags",
      type: "array",
      of: [{ type: "string" }],
      options: {
        list: [
          { title: "Featured", value: "featured" },
          {
            title: "Follow products and discounts on Instagram",
            value: "instagram",
          },
          { title: "Gallery", value: "gallery" },
        ],
      },
    },
  ],
});
```

Key Fields:

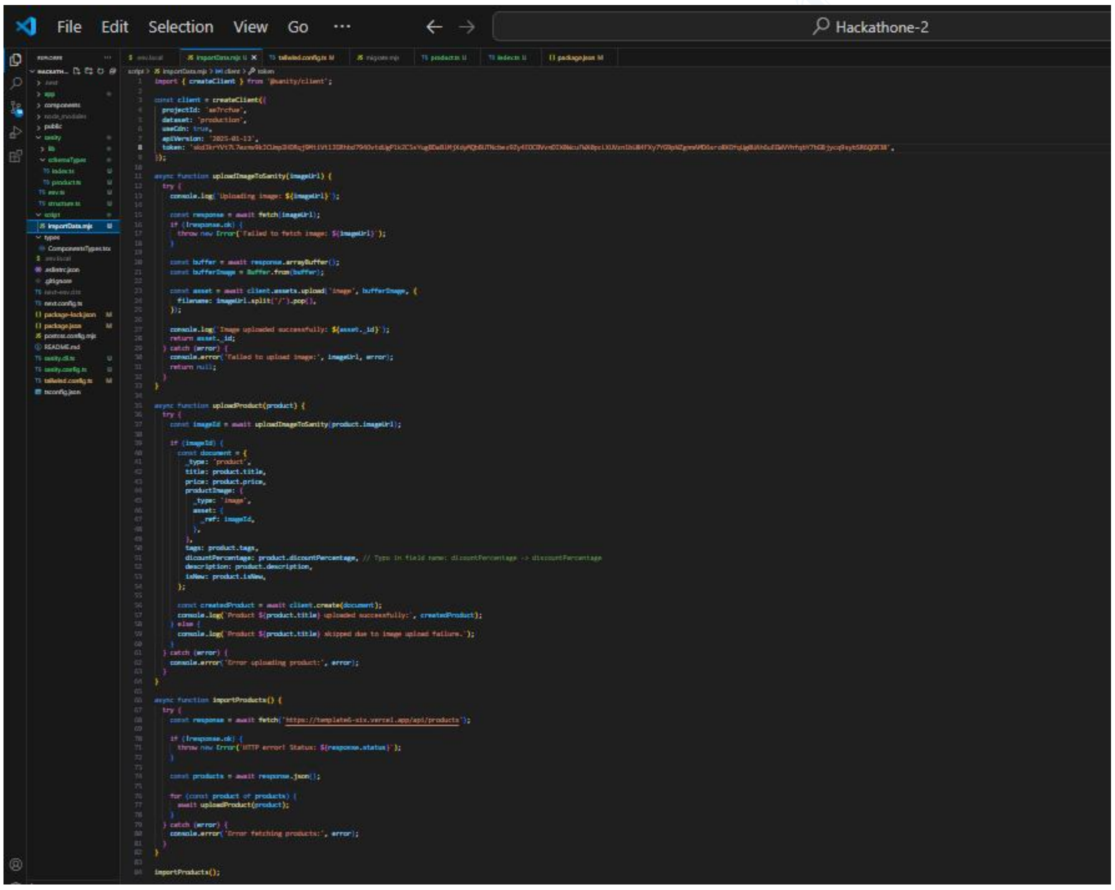
- Title: Name of the product.
- Description: Detailed description of the product.
- Price: The price at which the product is being sold.
- Image: Image(s) associated with the product.

As the focus was only on the product page, no categories schema was needed.

---

5. <u>Migration Script</u>

The migration was handled via the **importData.mjs** script. Below is the code used to automate the import of product data:



```
import { createClient } from '@sanity/client';

const client = createClient({
  projectId: 'ae7rcfue',
  dataset: 'production',
  useCdn: true,
  apiVersion: '2025-01-13',
  token: '[illegible]',
});

async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading image: ${imageUrl}`);

    const response = await fetch(imageUrl);
    if (!response.ok) {
      throw new Error(`Failed to fetch image: ${imageUrl}`);
    }

    const buffer = await response.arrayBuffer();
    const bufferImage = Buffer.from(buffer);

    const asset = await client.assets.upload('image', bufferImage, {
      filename: imageUrl.split('/').pop(),
    });

    console.log(`Image uploaded successfully: ${asset._id}`);
    return asset._id;
  } catch (error) {
    console.error('Failed to upload image:', imageUrl, error);
    return null;
  }
}

async function uploadProduct(product) {
  try {
    const imageId = await uploadImageToSanity(product.imageUrl);

    if (imageId) {
      const document = {
        _type: 'product',
        title: product.title,
        price: product.price,
        productImage: {
          _type: 'image',
          asset: {
            _ref: imageId,
          },
        },
        tags: product.tags,
        dicountPercentage: product.dicountPercentage, // Typo in field name: dicountPercentage -> discountPercentage
        description: product.description,
        isNew: product.isNew,
      };

      const createdProduct = await client.create(document);
      console.log(`Product ${product.title} uploaded successfully:`, createdProduct);
    } else {
      console.log(`Product ${product.title} skipped due to image upload failure.`);
    }
  } catch (error) {
    console.error('Error uploading product:', error);
  }
}

async function importProducts() {
  try {
    const response = await fetch('https://template6-six.vercel.app/api/products');

    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }

    const products = await response.json();

    for (const product of products) {
      await uploadProduct(product);
    }
  } catch (error) {
    console.error('Error fetching products:', error);
  }
}

importProducts();
```

This script fetches an array of product data and creates corresponding entries in Sanity CMS using the client.create() method

---

6. Conclusion

On Day 3 of the hackathon, the following accomplishments were achieved:

- The **data migration** was successfully completed, transferring product data from Sanity CMS into the local database.
- The **product schema** was implemented to structure the product data.
- **API integration** with the frontend was established, allowing the marketplace to fetch and display product data from the local database.

With this work completed, the next steps involve working on the frontend integration and ensuring the smooth display of the migrated product data.