Hackathon Day 3

Day3 API Integration and Data Migration

API Integration Report

1. Introduction

This report details the process of integrating product data from an external API into the backend system of **E-commerce**, an online clothing store. The integration utilizes **Sanity CMS** for content management and **Next.js** for frontend rendering.

Key Objectives of the Integration:

1. Efficiently store and manage product data in **Sanity CMS**.
2. Retrieve real-time data from an **external API**.
3. Dynamically render the fetched data on the **frontend** for a seamless shopping experience.

API Integration and Data Migration with Sanity and Schema

Step 1: Migrating Data to Sanity CMS

The retrieved data was structured and stored in **Sanity CMS** using its **JavaScript client library**.

1

```
import type { Rule as RuleType } from '@sanity/types';

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

```
      validation: (Rule: RuleType) => Rule.required().error('Name is required'),
    },
    {
      name: 'image',
      type: 'image',
      title: 'Image',
      options: {
        hotspot: true,
      },
      description: 'Upload an image of the product.',
    },
    {
      name: 'price',
      type: 'string',
      title: 'Price',
      validation: (Rule: RuleType) => Rule.required().error('Price is required'),
    },
    {
      name: 'description',
      type: 'text',
      title: 'Description',
      validation: (Rule: RuleType) =>
        Rule.max(150).warning('Keep the description under 150 characters.'),
    },
    {
      name: 'discountPercentage',
      type: 'number',
      title: 'Discount Percentage',
      validation: (Rule: RuleType) =>
        Rule.min(0).max(100).warning('Discount must be between 0 and 100.'),
    },
    {
      name: 'isFeaturedProduct',
      type: 'boolean',
      title: 'Is Featured Product',
    },
    {
      name: 'stockLevel',
      type: 'number',
      title: 'Stock Level',
      validation: (Rule: RuleType) => Rule.min(0).error('Stock level must be a
positive number.'),
    },
    {
      name: 'category',
```

```
      type: 'string',
      title: 'Category',
      options: {
        list: [
          { title: 'Chair', value: 'Chair' },

          { title: 'Sofa', value: 'Sofa' },
        ],
      },
      validation: (Rule: RuleType) => Rule.required().error('Category is
required'),
    },
  ],
};
```

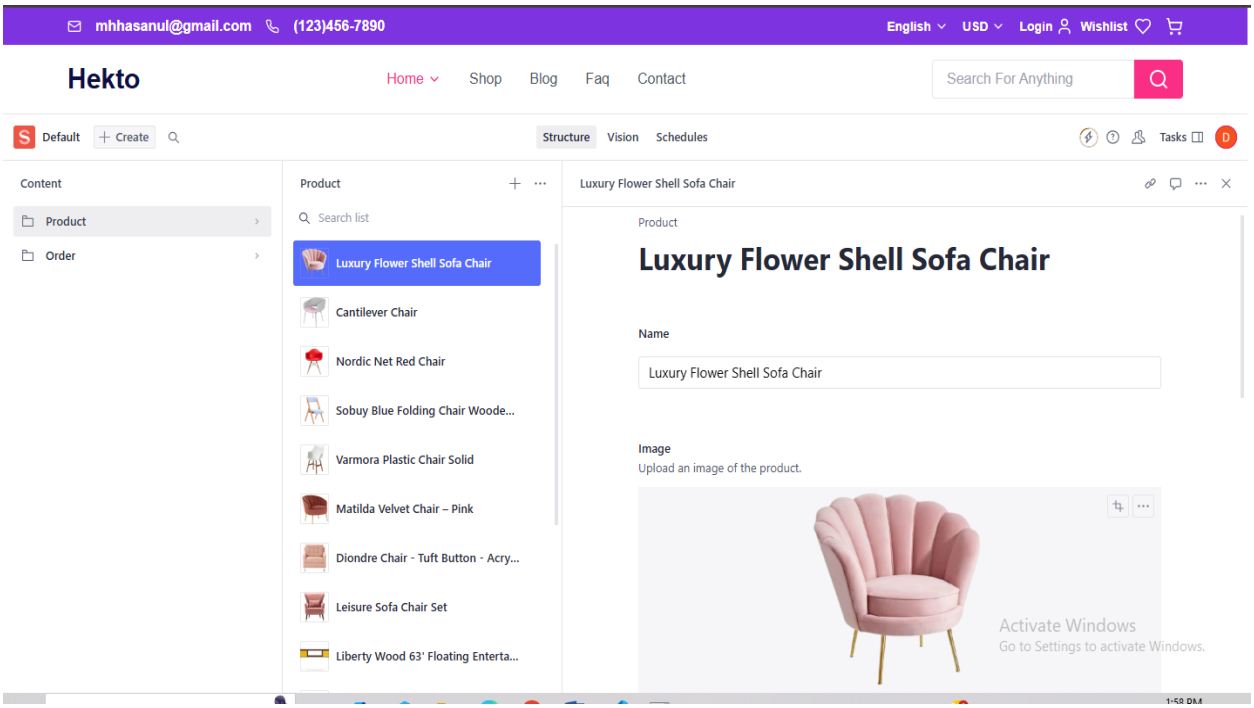
2

```
import { createClient } from '@sanity/client';
import axios from 'axios';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';
import path from 'path';

const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);
dotenv.config({ path: path.resolve(__dirname, '../../.env.local') });

const client = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
  token: process.env.SANITY_API_TOKEN,
  apiVersion: '2025-01-15',
  useCdn: false,
});

async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading Image : ${imageUrl}`);
    const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
    const buffer = Buffer.from(response.data);
    const asset = await client.assets.upload('image', buffer, {
      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 importData() {
  try {
    console.log('Fetching Product Data From API ...');

    const response = await axios.get("https://next-ecommerce-template-
4.vercel.app/api/product")
    const products = response.data.products;

    for (const item of products) {
      console.log(`Processing Item: ${item.name}`);

      let imageRef = null;
      if (item.imagePath) {
        imageRef = await uploadImageToSanity(item.imagePath);
      }

      const sanityItem = {
        _type: 'product',
        name: item.name,
        category: item.category || null,
        price: item.price,
        description: item.description || '',
        discountPercentage: item.discountPercentage || 0,
        stockLevel: item.stockLevel || 0,
        isFeaturedProduct: item.isFeaturedProduct,
        image: imageRef
          ? {
              _type: 'image',
              asset: {
                _type: 'reference',
                _ref: imageRef,
              },
            }
          : undefined,
      };
```

```
      console.log(`Uploading ${sanityItem.category} - ${sanityItem.name} to
Sanity !`);
      const result = await client.create(sanityItem);
      console.log(`Uploaded Successfully: ${result._id}`);
      console.log("-------------------------------------------------------")
      console.log("\n\n")
    }

    console.log('Data Import Completed Successfully !');
  } catch (error) {
    console.error('Error Importing Data : ', error);
  }
}

importData();
```

Sanity Dashboard Screenshot Showing Stored Products



Step 2: Retrieving Data from the API

The API offers specific endpoints for accessing data, including:

- **Products Endpoint:** This endpoint provides essential product details such as:
  - Product name
  - Pricing information
  - Descriptions
  - Categories
  - Stock availability
  - Product images

Base API URL:

https://next-ecommerce-template-4.vercel.app/api/product

```
async function importData() {
  try {
    const response = await axios.get("https://next-ecommerce-template-4.vercel.app/api/product")
```

4. Frontend Integration

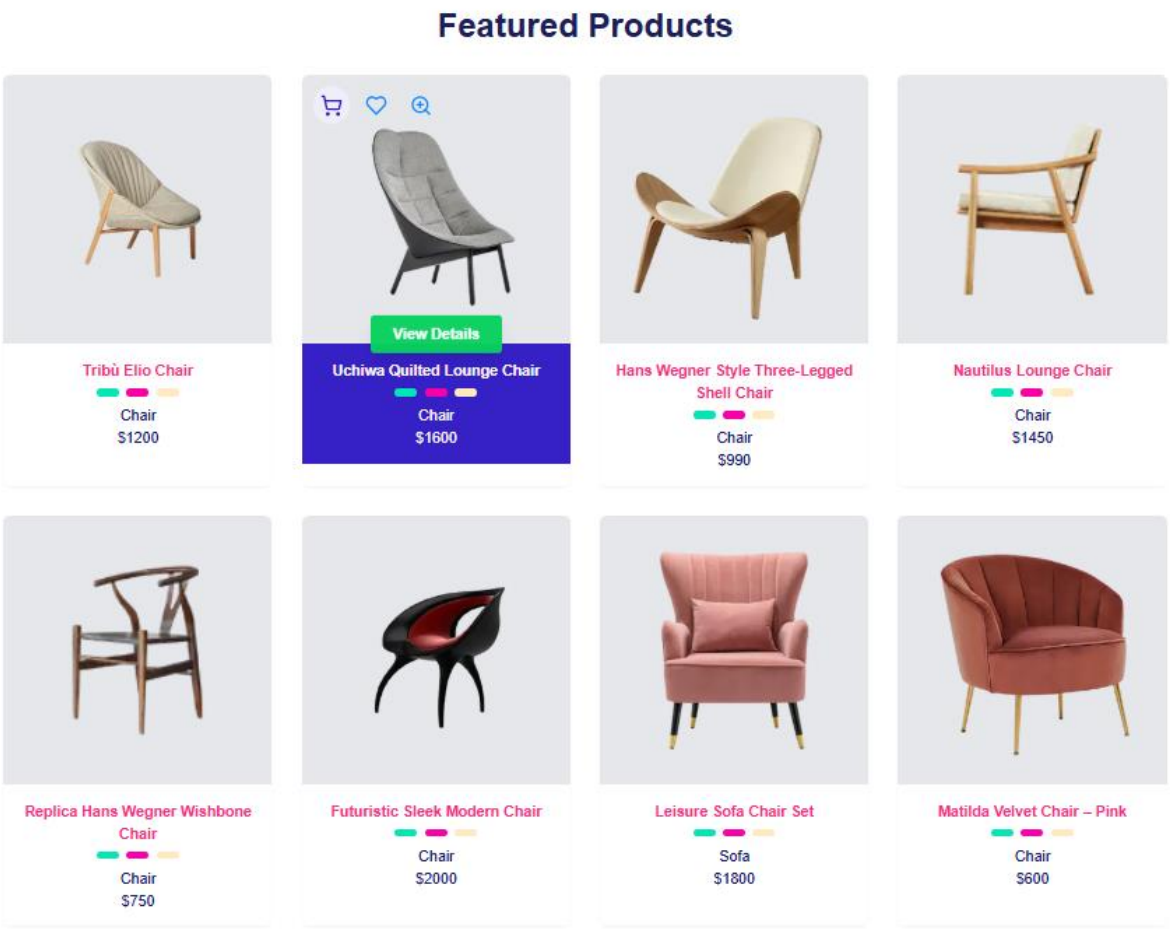
The stored product data was retrieved from **Sanity CMS** and showcased on the **E-Commerce** frontend. Utilizing **Next.js**, the data was dynamically rendered to populate the product listing interface, ensuring a seamless shopping experience.

code Example for Fetching Data in Frontend:

```
const Page = async () => {
  // Fetch products from Sanity
  const products = await client.fetch<Product[]>(
    `*[_type == "product"] {
      _id,
      name,
      "imageUrl": image.asset->url,
      price,
      description,
      discountPercentage,
      stockLevel,
      category
    }`
  );
```

Product Display:

Below is an image of the product listing page, where products are dynamically displayed on the frontend.



4. Conclusion

The API integration for E-Commerce was successfully implemented. The process involved retrieving data, storing it in Sanity CMS, and dynamically displaying it on the frontend. Attached screenshots confirm the successful execution at each stage. This integration optimizes product management, ensuring a smooth experience for both administrators and users.