**document Title: "Day 3 - API Integration Report –**

**[HMS E-COMMERCE]"**

**Step-by-Step API Integration Process**

**Step 1: API Selection & Configuration**

* We are using the **Sanity CMS API** for content management.
* The purpose of this API is to **store and retrieve data** related to products, categories, and orders.
* Authentication is handled using an **API\_KEY** and a **projectId**.
* Sensitive credentials are securely stored in the **.env file** to prevent security risks.

**Step 2: API Integration in Code**

* The API was integrated into a **React application** and a **Node.js backend**.
* We used **Axios** and **Fetch API** to make API requests and retrieve data.
* API request handling and **error handling mechanisms** were implemented in the backend to ensure smooth functionality.

import { createClient } from '@sanity/client'

import axios from 'axios'

import dotenv from 'dotenv'

import { fileURLToPath } from 'url'

import path from 'path'

// Load environment variables from .env.local

const \_\_filename = fileURLToPath(import.meta.url)

const \_\_dirname = path.dirname(\_\_filename)

dotenv.config({ path: path.resolve(\_\_dirname, '../.env.local') })

// Create Sanity client

const client = createClient({

projectId: process.env.NEXT\_PUBLIC\_SANITY\_PROJECT\_ID,

dataset: process.env.NEXT\_PUBLIC\_SANITY\_DATASET,

useCdn: false,

token: process.env.SANITY\_API\_TOKEN,

apiVersion: '2021-08-31'

})

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 products from API...')

const response = await axios.get('https://fakestoreapi.com/products')

const products = response.data

console.log(`Fetched ${products.length} products`)

for (const product of products) {

console.log(`Processing product: ${product.title}`)

let imageRef = null

if (product.image) {

imageRef = await uploadImageToSanity(product.image)

}

const sanityProduct = {

\_type: 'product',

name: product.title,

description: product.description,

price: product.price,

discountPercentage: 0,

priceWithoutDiscount: product.price,

rating: product.rating?.rate || 0,

ratingCount: product.rating?.count || 0,

tags: product.category ? [product.category] : [],

sizes: [],

image: imageRef ? {

\_type: 'image',

asset: {

\_type: 'reference',

\_ref: imageRef,

},

} : undefined,

}

console.log('Uploading product to Sanity:', sanityProduct.name)

const result = await client.create(sanityProduct)

console.log(`Product uploaded successfully: ${result.\_id}`)

}

console.log('Data import completed successfully!')

} catch (error) {

console.error('Error importing data:', error)

}

}

importData()

## **Adjustments Made to Schemas**

* **Added a new schema for products** in Sanity CMS to support product details, pricing, images, and categories.
* **Modified the existing schema** to include image references instead of direct URLs for better media management.

To store product-related data in \*\*Sanity CMS\*\*, we created a \*\*product schema\*\*. This schema defines the structure of product documents, including fields such as name, description, price, discount percentage, ratings, tags, sizes, and images.

### \*\*Updated Product Schema\*\*

Below is the schema used for defining product data in Sanity:

``

export default {

name: 'product',

type: 'document',

title: 'Product',

fields: [

{

name: 'name',

type: 'string',

title: 'Product Name',

},

{

name: 'description',

type: 'string',

title: 'Description'

},

{

name: 'price',

type: 'number',

title: 'Product Price',

},

{

name: 'discountPercentage',

type: 'number',

title: 'Discount Percentage',

},

{

name: 'priceWithoutDiscount',

type: 'number',

title: 'Price Without Discount',

description: 'Original price before discount'

},

{

name:'rating',

type:'number',

title:'Rating',

description:'Rating of the product'

},

{

name: 'ratingCount',

type: 'number',

title: 'Rating Count',

description: 'Number of ratings'

},

{

name: 'tags',

type: 'array',

title: 'Tags',

of: [{ type: 'string' }],

options: {

layout: 'tags'

},

description: 'Add tags like "new arrival", "bestseller", etc.'

},

{

name: 'sizes',

type: 'array',

title: 'Sizes',

of: [{ type: 'string' }],

options: {

layout: 'tags'

},

description: 'Add sizes like S , M , L , XL , XXL'

},

{

name: 'image',

type: 'image',

title: 'Product Image',

options: {

hotspot: true // Enables cropping and focal point selection

}

}

]

};

**Schema Explanation**

* **Product Name**: Stores the name of the product.
* **Description**: A brief description of the product.
* **Price & Discount**: Includes product price, discount percentage, and original price.
* **Ratings**: Stores the product rating and total number of ratings.
* **Tags & Sizes**: Allows adding multiple tags and size options.
* **Image**: Enables product image upload with cropping functionality.

This schema ensures all product-related data is stored and structured properly in **Sanity CMS**, making it easy to retrieve and display in the frontend.

## **Migration Steps and Tools Used**

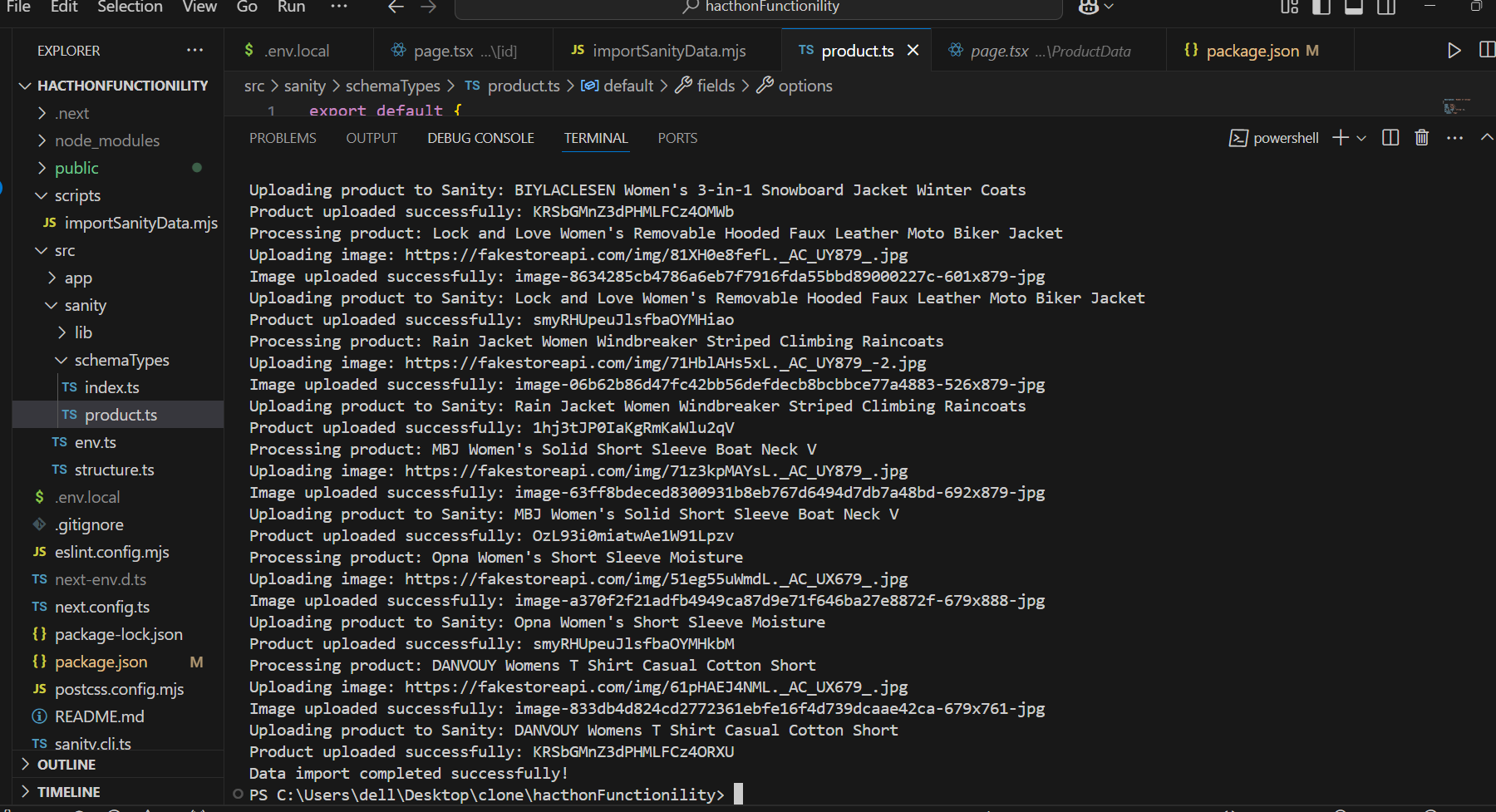
### ****Steps for Data Migration****

1. **Fetched product data** from an external API (FakeStore API).
2. **Processed and validated** the retrieved data to ensure consistency.
3. **Uploaded product images** to Sanity CMS for proper media handling.
4. **Mapped the data structure** to match the Sanity CMS schema.
5. **Inserted products** into the Sanity CMS database using the API.

### ****Tools Used****

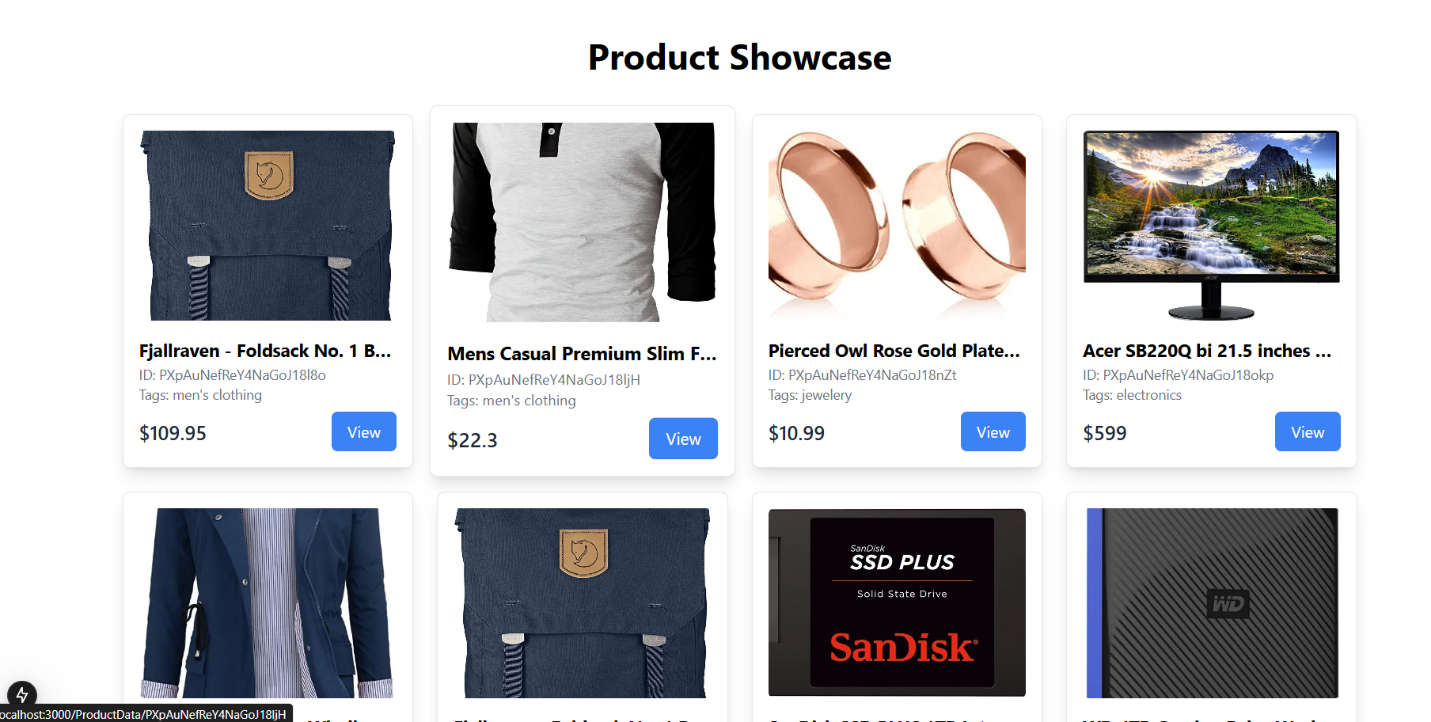
* **Sanity CMS API** (for content management)
* **Axios** (for making API requests)
* **dotenv** (for handling environment variables)
* **Node.js** (for script execution)

**API Calls Screenshots**

* 📷 Screenshot of the API response in the **browser console**.
* 

📷

Screenshot of the **frontend displaying product data**



**Code snippets for API integration and migration scripts.**

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

import axios from 'axios';

import dotenv from 'dotenv';

import { fileURLToPath } from 'url';

import path from 'path';

// Load environment variables from .env.local

const \_\_filename = fileURLToPath(import.meta.url);

const \_\_dirname = path.dirname(\_\_filename);

dotenv.config({ path: path.resolve(\_\_dirname, '../.env.local') });

// Create Sanity client

const client = createClient({

projectId: process.env.NEXT\_PUBLIC\_SANITY\_PROJECT\_ID,

dataset: process.env.NEXT\_PUBLIC\_SANITY\_DATASET,

useCdn: false,

token: process.env.SANITY\_API\_TOKEN,

apiVersion: '2021-08-31',

});

// Function to Upload Images to Sanity

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;

}

}

// Function to Import Data from FakeStore API to Sanity

async function importData() {

try {

console.log('Fetching products from API...');

const response = await axios.get('https://fakestoreapi.com/products');

const products = response.data;

console.log(`Fetched ${products.length} products`);

for (const product of products) {

console.log(`Processing product: ${product.title}`);

let imageRef = null;

if (product.image) {

imageRef = await uploadImageToSanity(product.image);

}

const sanityProduct = {

\_type: 'product',

name: product.title,

description: product.description,

price: product.price,

discountPercentage: 0,

priceWithoutDiscount: product.price,

rating: product.rating?.rate || 0,

ratingCount: product.rating?.count || 0,

tags: product.category ? [product.category] : [],

sizes: [],

image: imageRef ? {

\_type: 'image',

asset: {

\_type: 'reference',

\_ref: imageRef,

},

} : undefined,

};

console.log('Uploading product to Sanity:', sanityProduct.name);

const result = await client.create(sanityProduct);

console.log(`Product uploaded successfully: ${result.\_id}`);

}

console.log('Data import completed successfully!');

} catch (error) {

console.error('Error importing data:', error);

}

}

// Run the script

importData();

### ****Steps to Implement the Migration Script****

1. **Setup Environment Variables:** Ensure you have an .env.local file with the following entries:
   * NEXT\_PUBLIC\_SANITY\_PROJECT\_ID: Your Sanity Project ID.
   * NEXT\_PUBLIC\_SANITY\_DATASET: The dataset you are using in Sanity (e.g., "production").
   * SANITY\_API\_TOKEN: A token for your Sanity API (to be generated in the Sanity dashboard).
2. **Install Required Dependencies:** If not installed already, you'll need to install the necessary packages:

bash

CopyEdit

npm install axios dotenv @sanity/client

1. **Run the Script:** You can execute the script using Node.js to import the data into your Sanity CMS:

bash

CopyEdit

node scripts/importData.js

1. **Check Sanity CMS:** After running the script, check your **Sanity CMS** for the newly imported products.