API Integration Report - ShopCo

1. Overview of API Integration

🔍 API Selection

To integrate data, the given API was utilized for retrieving product details:

API URL: https://template1-neon-nu.vercel.app/api/products

Using the provided API ensured accurate data retrieval and compatibility with the project requirements.

🚀 Fetching Data

The Fetch was utilized to send GET requests to the given API. Once the data was fetched, it was processed and structured for seamless integration with Sanity CMS.

🛠️ Sanity Integration Process

The integration with Sanity CMS followed these main steps:

1.Sanity Client Configuration:

A custom client was set up with the necessary credentials (project ID, dataset, and token).

Image Uploads:

Images were retrieved from URLs, converted into suitable formats, and uploaded to Sanity as assets using a custom function.

Product Data Uploads:

Product details were mapped to a schema and uploaded to Sanity. References for assets and inventory were created to maintain proper linkage.

✨ Key Milestones

- Successfully retrieved product data from the given API.
- Uploaded images and assets into the CMS.
- Established relationships between products and their inventory.
- Integrated structured product data into Sanity.

2. Schema Enhancements

🛠 Initial Schema Design

Initially, schemas for products and inventory were set up with attributes like:

Name

Price

Description

Rating

Category

🔧 Adjustments and Improvements

During development, adjustments were made to:

Resolve Reference Issues:

Ensured all product-inventory relationships were correctly linked.

Add New Fields:

Included additional attributes like category, availability, and discount rate to enhance product details.

Validation Rules:

Implemented checks for empty product names and invalid prices to maintain data quality.

These modifications improved schema efficiency and aligned it with the project's goals.

Migration Steps and Tools Used

Migration Overview:

This migration process focused on transferring data from a given API to the Sanity CMS. The goal was to ensure the data was organized correctly, with all references properly linked for smooth functionality.

Steps Involved:

Setting Up the Sanity Project:

A new project was created in Sanity. Credentials like the project ID, dataset name, and token were obtained and configured to set up the system.

Fetching Data with Fetch:

The Fetch library was used to retrieve data from the given API. Custom error-handling functions were added to manage data fetching efficiently.

Creating and Adjusting Schemas:

Schemas for products, inventory, orders, and shipments were created in Sanity. Relationships between products, images, and inventory were defined to ensure data integrity.

Importing Data:

Data from the given API was processed, and for each product:

Images were uploaded to Sanity using a custom image upload feature.

Inventory references were either found or newly created.

Product details were linked to the schema and saved in Sanity.

Verification:

After completing the migration, all data in Sanity was reviewed to confirm accuracy and proper linking of references.

Tools and Technologies Used:

Fetch: To fetch data from the given API.

Sanity Client: To connect with Sanity CMS and manage data uploads.

Custom Migration Script: To upload product details, images, and inventory while ensuring all references were properly created or updated.

Node.js: The migration script was written and executed using Node.js.

Conclusion

The migration for ShopCo was completed successfully. Data from the given API was transferred to Sanity CMS, with custom functions ensuring smooth image uploads and accurate inventory referencing.

By linking product details, images, and inventory data effectively, the system is now ready for future use and scaling. This process ensures ShopCo's data is well-organized and easy to manage.