**Day 3 - API Integration Report - Reliable Furniture Hub**

**API Integration Process**

The API integration for **Reliable Furniture Hub** involved fetching product data from an external API and migrating it into **Sanity CMS**. The integration process included:

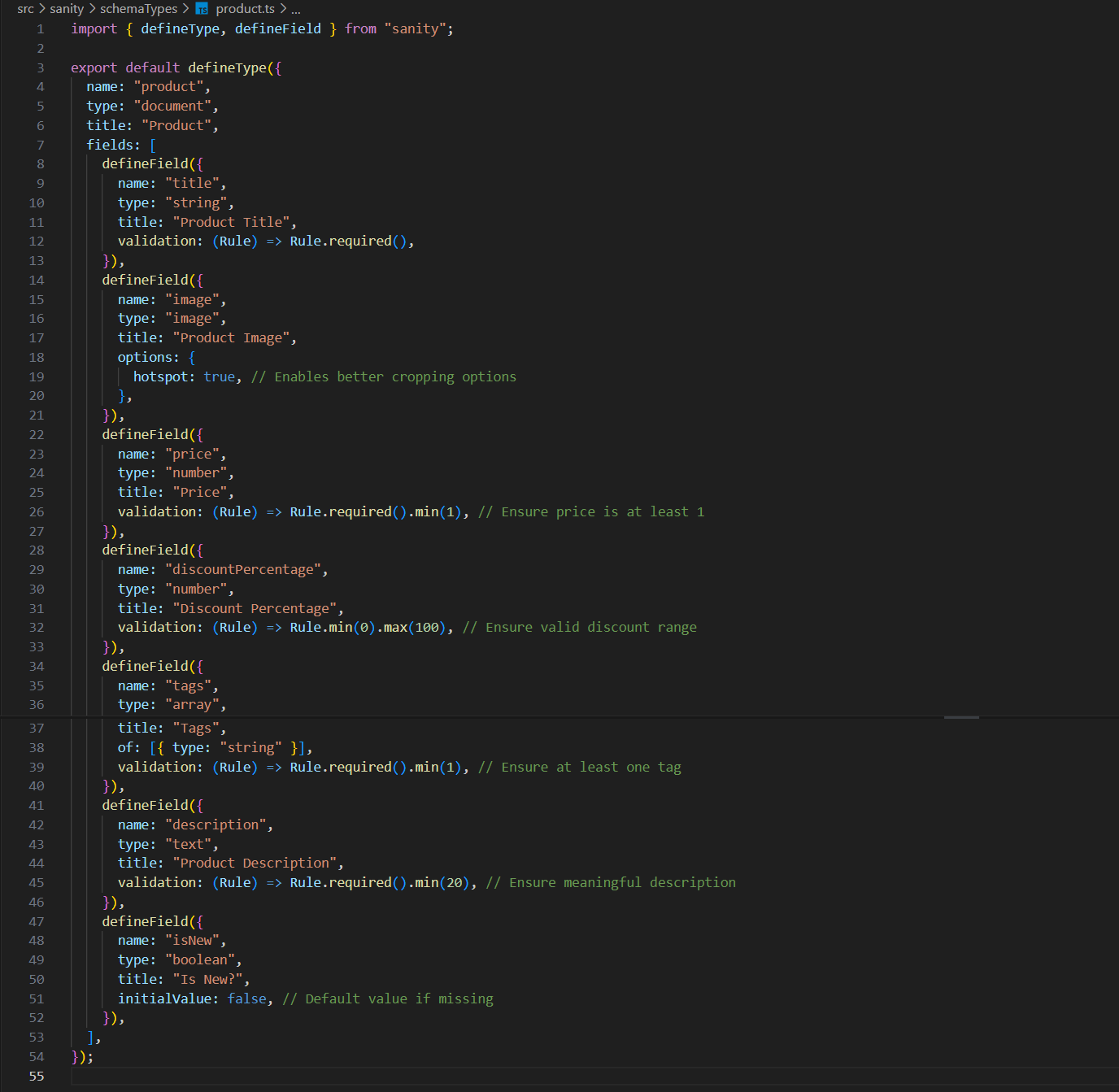
1. **Fetching API Data**: The product data was retrieved from https://template6-six.vercel.app/api/products.
2. **Image Uploads**: Product images were uploaded separately to **Sanity CMS** using the Sanity Assets API.
3. **Schema Adjustments**: The schema was modified to match the structure of the imported API data.
4. **Data Migration**: The fetched products were structured correctly and stored in Sanity CMS.
5. **Testing and Debugging**: The API responses and uploaded products were validated for correctness.

**Adjustments Made to Schemas**

The **product schema** was adjusted to ensure compatibility with the API data. The key changes included:

* **Title Field** (title): Used to store the product name.
* **Price Field** (price): Ensured numeric validation with a minimum value.
* **Tags Field** (tags): Converted into an array to support multiple product categories.
* **Discount Field** (discountPercentage): Added validation to restrict values between 0-100.
* **Description Field** (description): Required a minimum text length for better data consistency.
* **Image Upload** (image): Integrated with the Sanity asset system.
* **New Product Flag** (isNew): Included a boolean field to mark newly added items.

The final **TypeScript-based schema (product.ts)** was implemented as follows:

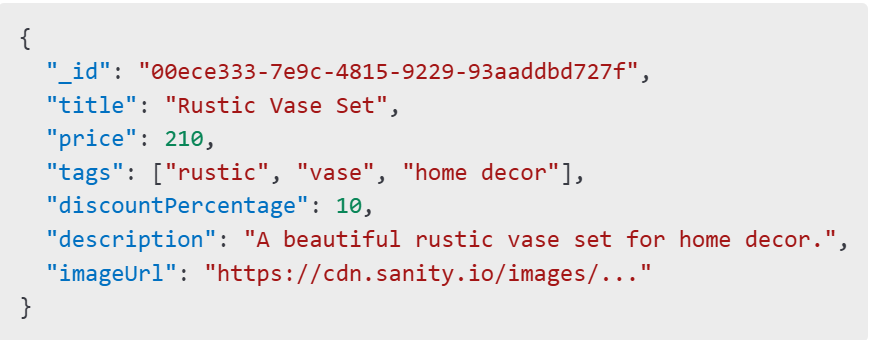


**Migration Steps and Tools Used**

The data migration was conducted using **Node.js, TypeScript, and Sanity Client APIs**. Below are the steps taken:

**1. Fetching API Data**

* The external API was accessed via Axios to retrieve product details.
* Example API response:



**2. Uploading Images to Sanity CMS**

* Each image was fetched and uploaded to **Sanity's asset system**.

**3. Storing Data in Sanity CMS**

* The cleaned product data was structured and inserted into the **Sanity database**.

**4. Error Handling & Debugging**

* Ensured proper handling of missing fields and invalid responses.
* Implemented logging for tracking uploaded products.

**5. Deployment to Sanity Studio**

* The schema was deployed using:

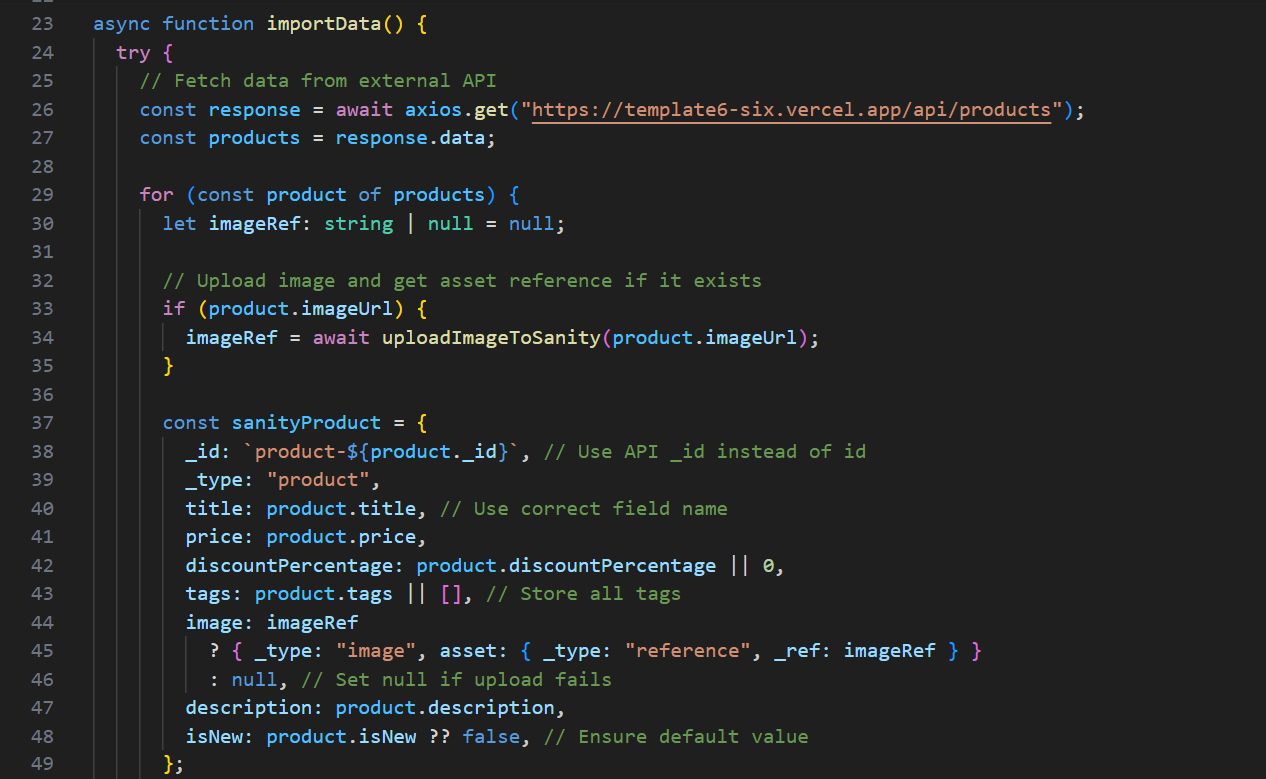
Sanity deploy

**Code Snippets for API Integration and Migration**

**API Data Fetching & Image Upload (importData.ts)**



**Data Import to Sanity CMS**





**Conclusion**

The API integration and data migration process was successfully completed. The data from an external API was fetched, images were uploaded, and the structured information was stored in **Sanity CMS**. With this implementation, the marketplace is now ready for dynamic product updates and further frontend integration.