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:

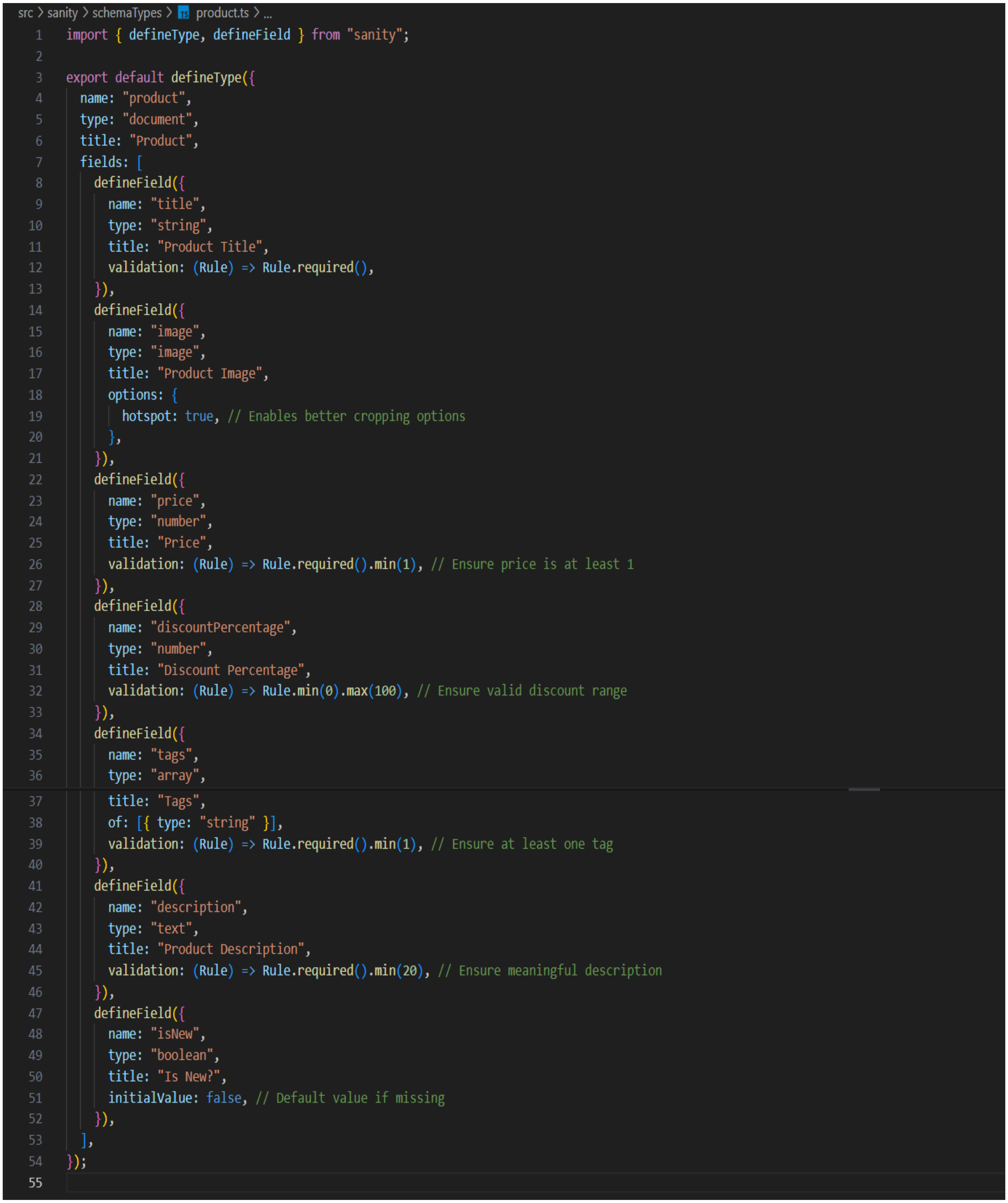
1. **Fetching API Data**: The product data was retrieved from https://template6-six.vercel.app/api/products.
2. **Image Uploads**: Product images were uploaded separately to **Sanity CMS** using the Sanity Assets API.
3. **Schema Adjustments**: The schema was modified to match the structure of the imported 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:


src > sanity > schemaTypes > product.ts > ...

```ts
import { defineType, defineField } from "sanity";

export default defineType({
  name: "product",
  type: "document",
  title: "Product",
  fields: [
    defineField({
      name: "title",
      type: "string",
      title: "Product Title",
      validation: (Rule) => Rule.required(),
    }),
    defineField({
      name: "image",
      type: "image",
      title: "Product Image",
      options: {
        hotspot: true, // Enables better cropping options
      },
    }),
    defineField({
      name: "price",
      type: "number",
      title: "Price",
      validation: (Rule) => Rule.required().min(1), // Ensure price is at least 1
    }),
    defineField({
      name: "discountPercentage",
      type: "number",
      title: "Discount Percentage",
      validation: (Rule) => Rule.min(0).max(100), // Ensure valid discount range
    }),
    defineField({
      name: "tags",
      type: "array",
      title: "Tags",
      of: [{ type: "string" }],
      validation: (Rule) => Rule.required().min(1), // Ensure at least one tag
    }),
    defineField({
      name: "description",
      type: "text",
      title: "Product Description",
      validation: (Rule) => Rule.required().min(20), // Ensure meaningful description
    }),
    defineField({
      name: "isNew",
      type: "boolean",
      title: "Is New?",
      initialValue: false, // Default value if missing
    }),
  ],
});

```

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:

```json
{
  "_id": "00ece333-7e9c-4815-9229-93aaddbd727f",
  "title": "Rustic Vase Set",
  "price": 210,
  "tags": ["rustic", "vase", "home decor"],
  "discountPercentage": 10,
  "description": "A beautiful rustic vase set for home decor.",
  "imageUrl": "https://cdn.sanity.io/images/..."
}
```

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.

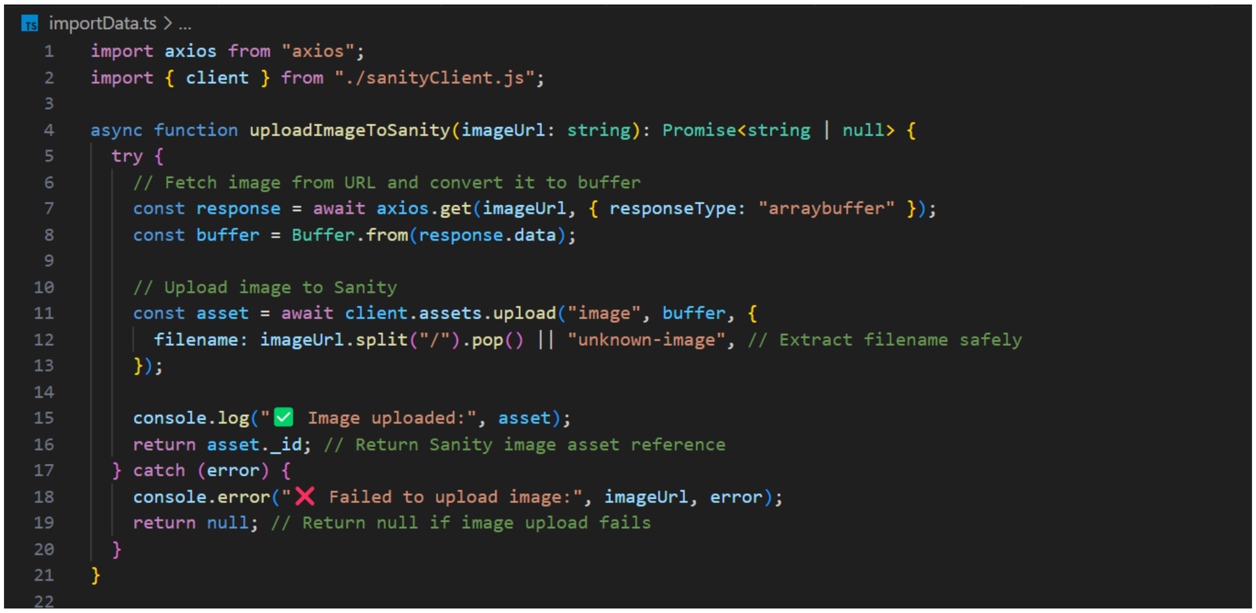
5. Deployment to Sanity Studio

- The schema was deployed using:

  Sanity deploy

Code Snippets for API Integration and Migration

API Data Fetching & Image Upload (importData.ts)



importData.ts > ...

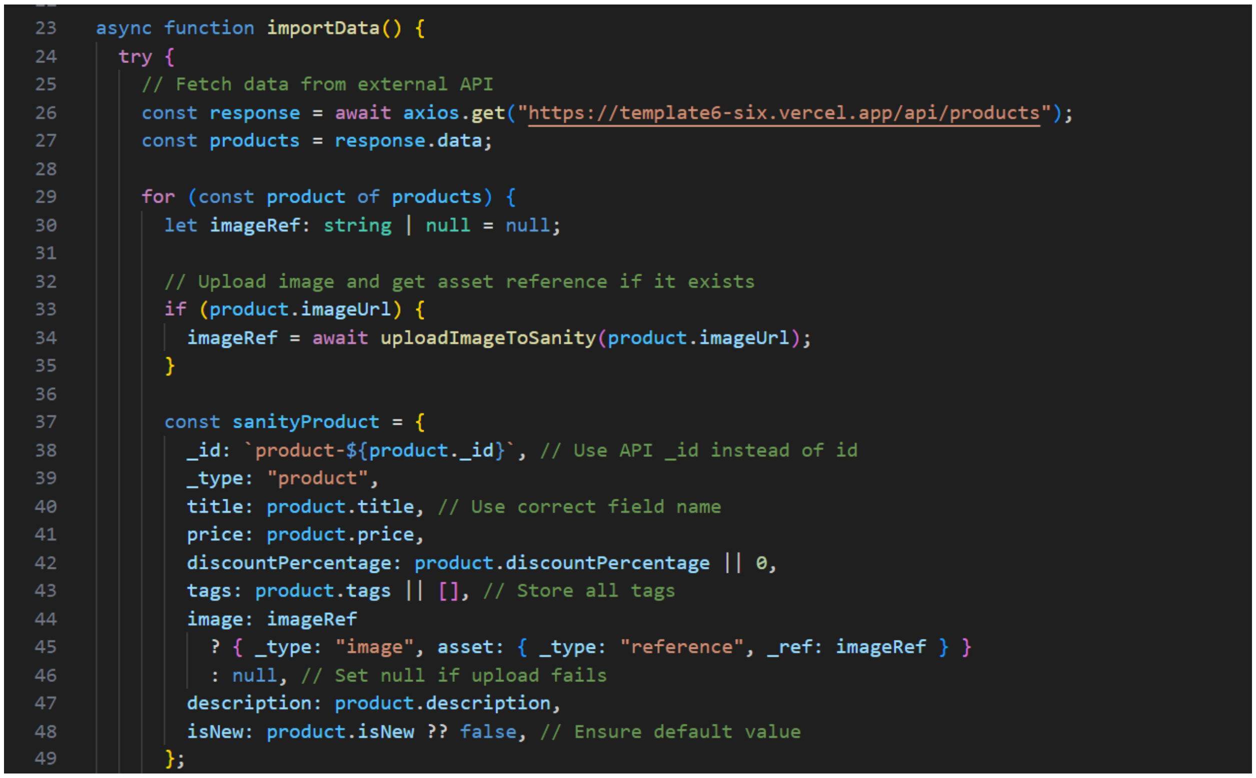
```ts
import axios from "axios";
import { client } from "./sanityClient.js";

async function uploadImageToSanity(imageUrl: string): Promise<string | null> {
  try {
    // Fetch image from URL and convert it to buffer
    const response = await axios.get(imageUrl, { responseType: "arraybuffer" });
    const buffer = Buffer.from(response.data);

    // Upload image to Sanity
    const asset = await client.assets.upload("image", buffer, {
      filename: imageUrl.split("/").pop() || "unknown-image", // Extract filename safely
    });

    console.log("✅ Image uploaded:", asset);
    return asset._id; // Return Sanity image asset reference
  } catch (error) {
    console.error("❌ Failed to upload image:", imageUrl, error);
    return null; // Return null if image upload fails
  }
}
```

Data Import to Sanity CMS

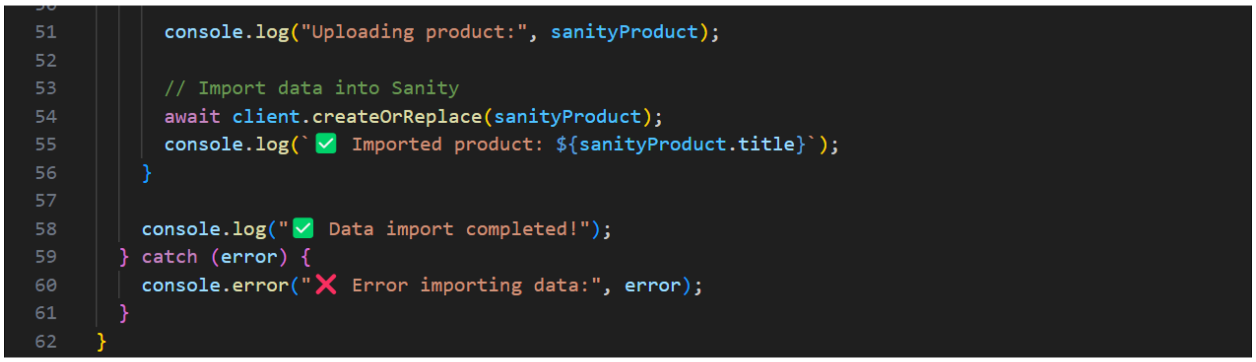


```ts
async function importData() {
  try {
    // Fetch data from external API
    const response = await axios.get("https://template6-six.vercel.app/api/products");
    const products = response.data;

    for (const product of products) {
      let imageRef: string | null = null;

      // Upload image and get asset reference if it exists
      if (product.imageUrl) {
        imageRef = await uploadImageToSanity(product.imageUrl);
      }

      const sanityProduct = {
        _id: `product-${product._id}`, // Use API _id instead of id
        _type: "product",
        title: product.title, // Use correct field name
        price: product.price,
        discountPercentage: product.discountPercentage || 0,
        tags: product.tags || [], // Store all tags
        image: imageRef
          ? { _type: "image", asset: { _type: "reference", _ref: imageRef } }
          : null, // Set null if upload fails
        description: product.description,
        isNew: product.isNew ?? false, // Ensure default value
      };
```



```
      console.log("Uploading product:", sanityProduct);

      // Import data into Sanity
      await client.createOrReplace(sanityProduct);
      console.log(`✅ Imported product: ${sanityProduct.title}`);
    }

    console.log("✅ Data import completed!");
  } catch (error) {
    console.error("❌ Error importing data:", error);
  }
}
```

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.