

Day 03: API Integration & Migration - Avion

Integration Workflow and Migration Summary

API Connection Steps:

1. Linking Product and Category Endpoints:

- **Endpoints Used:**
 - Product Data: `hackathon-apis.vercel.app/api/products`
 - Category Data: `hackathon-apis.vercel.app/api/products`
- Utilized fetch for data retrieval in the custom migration script.
- Validated API responses to ensure they matched the schema fields in Sanity CMS.

2. Schema Refinement in Sanity:

- Addressed discrepancies between manual schema design and API response format.
- Updated schemas to ensure proper structure:
 - **Product Schema Adjustments:** Added fields like categories and tags.
 - **Category Schema Modifications:** Introduced an image as a reference field for handling category visuals.

3. Building the Migration Utility:

- Created `migrate.mjs` for automated data migration.
- Configured the Sanity client to manage image uploads and save data efficiently.
- Explicitly defined `.env.local` to resolve environmental variable conflicts.

Migration Process Overview:

1. Sanity Platform Installation:

- Successfully installed and initialized Sanity.

2. Schema Development:

- Designed product.ts schema matching API data.

3. Environment Configuration:

- Created an .env file to securely store Sanity credentials.
- Defined the .env file path explicitly within the migration script.

4. Custom Migration Script:

- Developed migrate.mjs to fetch and upload data to Sanity CMS.
- Mapped API fields accurately to resolve inconsistencies.
- Added the following command in package.json for execution:

"migrate": "node migrate.mjs"

5. Executing the Migration:

- Verified successful integration of product datasets in Sanity CMS.

Implementation Evidence:

1. Sanity CMS Dashboard View:

- Displays populated product entities.

2. Application Frontend Display:

- Renders live data seamlessly on the user interface.

Obstacles Addressed:

- Resolved .env.local file conflicts by explicitly declaring its path.
 - Modified Sanity schemas to accommodate new data fields.
 - Ensured correct image uploads and category-to-product mapping.
-

Strategic Choices:

- Opted for .env configuration to ensure secure and consistent credential management.
- Updated schema designs to harmonize API data with Sanity requirements.

```

> sanity > schemaTypes > TS product.ts > [e] product
1  import { defineType, defineField } from "sanity"
2
3  export const product = defineType({
4    name: "product",
5    title: "Product",
6    type: "document",
7    fields: [
8      defineField({
9        name: "category",
10       title: "Category",
11       type: "reference",
12       to: [{
13         type: "category"
14       }]
15     },
16     ),
17     defineField({
18       name: "name",
19       title: "Title",
20       validation: (rule) => rule.required(),
21       type: "string"
22     },
23     ),
24     defineField({
25       name: "slug",
26       title: "Slug",
27       validation: (rule) => rule.required(),
28       type: "slug"
29     },
30     ),
31     defineField({
32       name: "image",
33       type: "image",
34       validation: (rule) => rule.required(),
35     },
36     ),
37   ],
38 })

```

Product Schema In Sanity:

HomeAboutProducts

Avion

Plant potsCeramicsTablesChairsCrockeryTablewareCutlery

SDefault+ CreateQ

StructureVisionSchedules

Tasks

Content

Product

Category

Product

Search list

The Dandy chair

Bright Space

Cloud Haven Chair

Bold Nest


Sunny Chic

Reflective Haven


Modern Serenity

Timeless Elegance


New Ceramics




The Dandy Chair£250



Rustic Vase Set£155



The Silky Vase£125



The Lucy Lamp£399

View Collection