

Day- 3 *API Integration and Data Migration*

API Integration Process

This report details the process of integrating a custom MockAPI and migrating data into Sanity CMS for SHOP.CO, ensuring seamless data handling and compatibility with the marketplace's frontend and backend systems.

1. APIs Used:

- MockAPI for Initial Data: Endpoint:

<https://677fa0d60476123f76a7500c.mockapi.io/product> Functionality: Provided initial product and category data for migration.

```
async function importData() {
  try {
    console.log('Fetching products data...')
    const response = await axios.get('https://677fa0d60476123f76a7500c.mockapi.io/product')
    const products = response.data

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

    for (const product of products) {
      console.log(`Processing product: ${product.title}`)

      // Create or get category
      const category = await createOrFetchCategory(product.category)

      // Create or get style
      const style = await createOrFetchStyle(product.style)
```

Steps Take

2. Data Creation in MockAPI:

- MockAPI was used to create a dataset for products and categories.
- Example products fields: name, description, price, original price, images, colors, sizes, categories, tags .

3. Manual Data Migration to Sanity:

- Data from MockAPI was manually exported and imported into Sanity CMS.
- Used Sanity Studio for field mapping and validation.

4. Testing API Integration:

- Verified data accuracy after migration using Sanity Studio and frontend rendering.

5. Frontend Integration:

- Rendered Sanity CMS data in components like product cards and category filters.

6. Schema Adjustments in Sanity CMS

The following schema adjustments were made to align with MockAPI data:

- Products Schema:
 - o Added fields: inventory, categories, images, tags etc..
 - o Adjusted field names to match API, e.g., title to name.
- Categories Schema:
 - o Added a description field for better categorization.



```
$.env.local
1 NEXT_PUBLIC_SANITY_PROJECT_ID="n9ho5zv8"
2 NEXT_PUBLIC_SANITY_DATASET="production"
3 SANITY_TOKEN="sk4J8tQ8bXJ7OdN3xnBUxmQBpH5f8RoyNcExXDb413EgzmPgZ04vNyKAIPpQ8bUv1YIjzjE0WPQYVa2wJGztsCFRfPanQVwoN6NwYDCiR"
4
5
```

Data Migration

Methodology

1. Manual Data Migration:

- o Data from MockAPI was exported as JSON.

- o Imported into Sanity CMS using the built-in Studio interface.

2. Data Validation:

- o Ensured all imported fields matched the schema requirements in Sanity CMS.

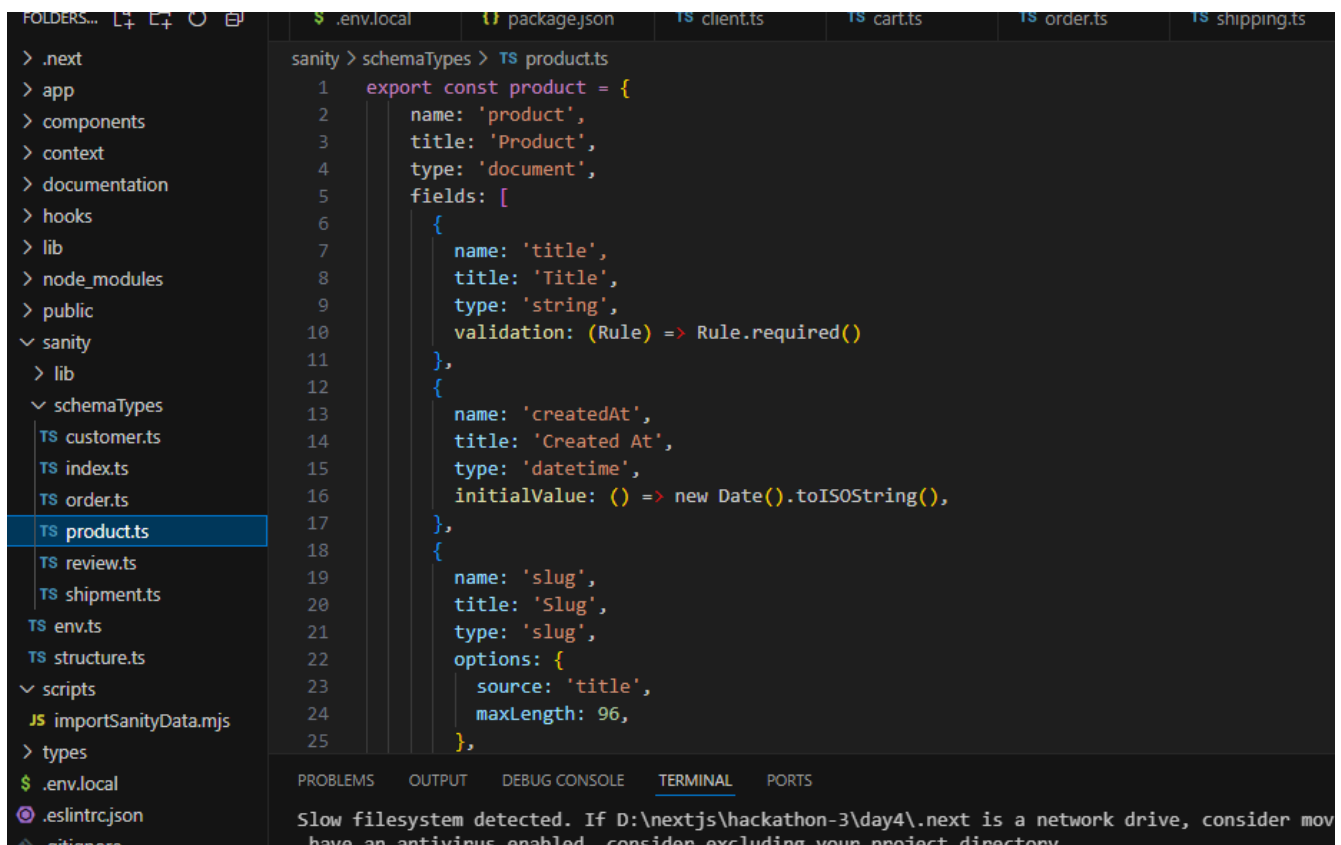
III. Challenges and Resolutions

- Field Mismatches:

Resolved by mapping MockAPI fields to Sanity schema fields.

- Duplicate Entries:

Validated unique fields (e.g., product_id) to prevent duplicates.



```
sanity > schemaTypes > TS products.ts
1  export const product = {
2    name: 'product',
3    title: 'Product',
4    type: 'document',
5    fields: [
6      {
7        name: 'title',
8        title: 'Title',
9        type: 'string',
10       validation: (Rule) => Rule.required()
11      },
12      {
13        name: 'createdAt',
14        title: 'Created At',
15        type: 'datetime',
16        initialValue: () => new Date().toISOString(),
17      },
18      {
19        name: 'slug',
20        title: 'Slug',
21        type: 'slug',
22        options: {
23          source: 'title',
24          maxLength: 96,
25        },
26      },
27    ],
28  }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Slow filesystem detected. If D:\nextjs\hackathon-3\day4\.next is a network drive, consider moving it to a local drive. Alternatively, you can have an antivirus enabled, consider excluding your project directory.

Frontend Display

IV. Components Updated

1. Product Listing Page:

- o Dynamically displayed products with stock status and category filters.

2. Category Filters:

- o Integrated category data into dropdown menus for filtering.

3. Product Details Page:

- o Showcased detailed product information including reviews and inventory.

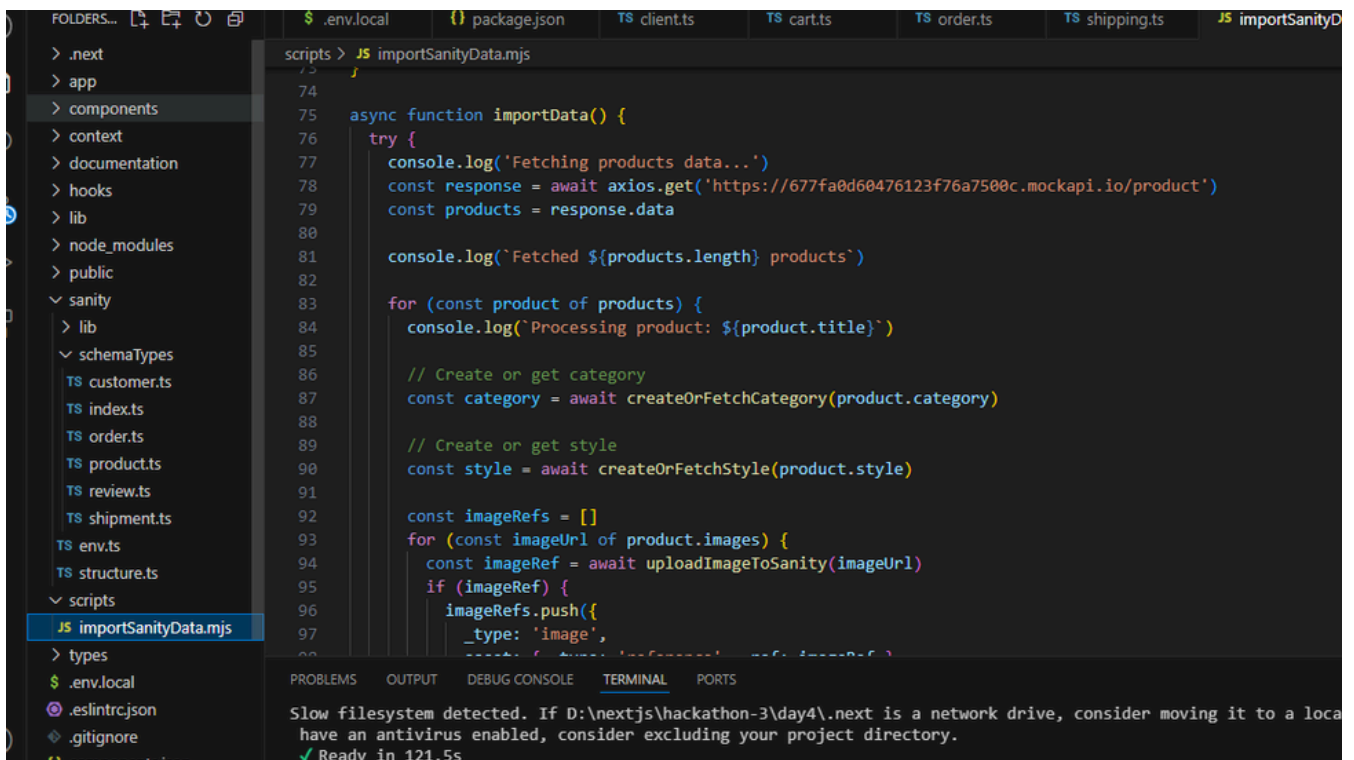
Error Handling

1. Fallback Mechanisms:

- o Displayed skeleton loaders and error messages in case of API failure:
if (error) return Failed to load products. Try again later.

2. Validation Checks:

- o Ensured all data from MockAPI met schema requirements before inserting into Sanity CMS.



The screenshot shows a VS Code editor with a file explorer on the left and a code editor in the center. The file explorer shows a project structure with folders like .next, app, components, context, documentation, hooks, lib, node_modules, public, sanity, schemaTypes, and scripts. The file 'importSanityData.mjs' is selected in the scripts folder. The code editor shows the following TypeScript code:

```
73 importSanityData.mjs
74
75 async function importData() {
76   try {
77     console.log('Fetching products data...')
78     const response = await axios.get('https://677fa0d60476123f76a7500c.mockapi.io/product')
79     const products = response.data
80
81     console.log(`Fetched ${products.length} products`)
82
83     for (const product of products) {
84       console.log('Processing product: ${product.title}')
85
86       // Create or get category
87       const category = await createOrFetchCategory(product.category)
88
89       // Create or get style
90       const style = await createOrFetchStyle(product.style)
91
92       const imageRefs = []
93       for (const imageUrl of product.images) {
94         const imageRef = await uploadImageToSanity(imageUrl)
95         if (imageRef) {
96           imageRefs.push({
97             _type: 'image',
98             url: imageRef.url,
99             alt: imageRef.alt
100           })
101         }
102       }
103       const productData = {
104         title: product.title,
105         category: category._id,
106         style: style._id,
107         images: imageRefs,
108       }
109       await createOrFetchProduct(productData)
110     }
111   } catch (error) {
112     console.error('Error importing data:', error)
113   }
114 }
```

The terminal window at the bottom shows the following output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Slow filesystem detected. If D:\nextjs\hackathon-3\day4\.next is a network drive, consider moving it to a local drive.
Have an antivirus enabled, consider excluding your project directory.
✓ Ready in 121.5s
```

Conclusion

Day 3's tasks were successfully completed using MockAPI for initial data creation and Sanity CMS for robust backend handling. The marketplace is now equipped with a functional backend and ready for advanced features.

Marketplace – SHOP.CO

By Alishba Meraj