Day 3 Assignment: API Integration and Data Migration

Objective:

The focus of this assignment is to integrate APIs and migrate data into Sanity CMS to power My Furniture Marketplace backend.

I will also test API integration in Next.js to display data dynamically on the frontend.

This exercise replicates real-world practices and ensures that the marketplace is functional and scalable.

Step 1: Understand the Provided API

1. Choose an API Template:

   - I reviewed the provided templates (0-9) or selected an external data source (e.g., Shopify, WooCommerce, Magento).

   - Example API: https://hackathon-apis.vercel.app/api/products.

2. Key Endpoints to Analyze:

   - /products: For product listings.

   - /categories: For product categories.

3. Tools Used:

   - Postman or browser developer tools to inspect API responses.

   - I analyzed fields like product_title, price, and description for schema mapping.

Step 2: Validate and Adjust Schema

1. Compared the existing Sanity CMS schema with the API data structure.

2. Identified and resolved field mismatches (e.g., API Field: product_title to Schema Field: name).

3. Adjusted field types and relationships.

Example Schema Code:

```typescript
export default {
  name: 'product',
  type: 'document',
  fields: [
    { name: 'name', type: 'string', title: 'Product Name' },
    { name: 'price', type: 'number', title: 'Price' },
    { name: 'description', type: 'text', title: 'Description' },
    { name: 'image', type: 'image', title: 'Product Image' },
  ],
};
```

Step 3: Migrate Data

1. Migration Method:

   - Using provided scripts or manual import with JSON/CSV.

   - Fetch data from Shopify, WooCommerce, or other platforms.

2. Example Migration Script:

```javascript
import fetch from 'node-fetch';
import sanityClient from '@sanity/client';

const client = sanityClient({
  projectId: 'yourProjectId',
  dataset: 'production',
  useCdn: false,
  token: 'yourSanityToken',
});

async function migrateData() {
  const response = await fetch('https://hackathon-apis.vercel.app/api/products');
  const products = await response.json();

  for (const product of products) {
    await client.create({
      _type: 'product',
      name: product.product_title,
      price: product.price,
      description: product.description,
      image: {
        _type: 'image',
        asset: { _ref: product.image_url },
      },
    });
  }
```

```
  console.log('Data migration complete!');
}

migrateData();
```

Step 4: API Integration in Next.js

1. Created utility functions for data fetching.

Example Utility Function:

```javascript
async function fetchProducts() {
  const response = await fetch('/api/products');
  const data = await response.json();
  return data;
}
```

2. Rendered data dynamically in Next.js components.

Example Component Code:

```javascript
```

```
import { useEffect, useState } from 'react';

export default function Products() {
  const [products, setProducts] = useState([]);

  useEffect(() => {
    async function fetchData() {
      const data = await fetchProducts();
      setProducts(data);
    }
    fetchData();
  }, []);

  return (
    <div>
      {products.map((product) => (
        <div key={product._id}>
          <h2>{product.name}</h2>
          <p>{product.description}</p>
          <p>${product.price}</p>
        </div>
      ))}
    </div>
  );
}
```

Checklist for Self-Validation:

- [Yes] Tested API endpoints using Postman or browser tools.
- [Yes] Adjusted schema to match API fields.
- [Yes] Populated Sanity CMS with imported data.
- [Yes] Rendered data dynamically in Next.js frontend.
- [Yes] Included error handling for API calls.