Day 3 - API Integration Report - [F.A Store]

1. API Integration Process

- **API Endpoint**: /products
- **Technology Stack**: Next.js, React, Sanity CMS, TypeScript.

Steps Taken:

- Integrated the /products endpoint with the frontend by fetching product data directly from **Sanity CMS** using the Sanity client.
- Used Sanity's **Groq** query language to fetch product data from the Sanity backend.
- The data is then displayed dynamically on the frontend by storing it in the state using **useState** and **useEffect** hooks.

Example code to fetch data:

```
import { createClient } from 'next-sanity';

import { useEffect, useState } from 'react';



const client = createClient({

  projectId: 'your-project-id',

  dataset: 'production',

  useCdn: true,

});



const fetchProducts = async () => {
```

```
  const products = await client.fetch(`*[_type == "product"]{title, price, description, images}`);

  setProducts(products);

};



useEffect(() => {

  fetchProducts();

}, []);
```

2. Adjustments Made to Schemas

- **Product Schema**:
  - Updated the product schema in **Sanity** to include necessary fields such as **title**, **price**, **description**, and **images**.
  - Ensured that the fields are correctly defined in the schema to allow for easy fetching and display of product data.

// Example of updated Sanity schema for products

```
export default {

  name: 'product',

  title: 'Product',

  type: 'document',

  fields: [

    { name: 'title', type: 'string' },

    { name: 'price', type: 'number' },

    { name: 'description', type: 'text' },

    { name: 'images', type: 'array', of: [{ type: 'image' }] },
```

```
 ]
};
```

3. Migration Steps and Tools Used

- **Migration Script**:
  - You may not need a migration script if you were adding data directly to **Sanity CMS**.
  - If there were any schema changes, you may have updated existing data manually through the **Sanity Studio** or via custom scripts to ensure the data was compatible with the updated schema.
- **Tools Used**:
  - **Sanity CLI** for schema updates.
  - **Sanity Studio** for managing product data manually.

4. Screenshots

1. **Populated Sanity CMS Fields**:
   Since you are using **Sanity** for managing the data, you will capture screenshots of the **Sanity Studio** interface, showing the populated product fields (such as title, price, description, and images).

   Example:

   This screenshot shows the populated product fields in the Sanity CMS for a product like "Red Shirt".

2. **Frontend Display**:
   Show a screenshot of the frontend where the product data is rendered. It will display the title, price, description, and images that were populated in **Sanity**.

   Example:

   This screenshot shows how the product data appears on the product listing page.

5. Code Snippets for API Integration and Migration Scripts

- **API Integration Code**:

```
// Fetching products from Sanity CMS

const fetchProducts = async () => {

  try {

    const products = await client.fetch(`*[_type == "product"]{title, price, description, images}`);

    setProducts(products);  // Update state with fetched products

  } catch (error) {

    console.error("Error fetching products:", error);

  }

};
```

Sanity Data Entry (Manual):

- Data was manually entered via **Sanity Studio**. The product fields like title, price, and description were populated through the CMS interface.

6. Best Practices Followed

1. **Sensitive Data**:
   Stored API keys and other sensitive data securely in .env files.
2. **Clean Code**:
   - Descriptive variable names were used.
   - Modularized functions for reusability.
   - Added comments for clarity.
3. **Data Validation**:
   - Ensured that the product data adhered to schema constraints within **Sanity**.
4. **Documentation**:

    - All steps, from fetching product data from Sanity to displaying it on the frontend, were thoroughly documented with appropriate code snippets.
5. **Version Control**:
    - Code changes were committed regularly to Git with meaningful messages.
6. **Testing**:
    - Product data was manually tested in **Sanity Studio** and displayed correctly on the frontend.