REPORT
HACKATHON Day 3 -MARKETPLACE BUILDER NIKEE

Overview:

On Day 3 task,I focused on integerating APIs and migrating data in to Sanity (CMS) to build a backend of functional marketplace,

- Connecting APIs to my next.js project.
- Moved data APIs into Sanity.
- Use provided APIs of Template 3.

1: API INTERGRATION:

The integration of APIs was essential for fetching product data from an external source (in this case, the provided APIs) and populating it into our Sanity CMS for the marketplace. Below is the step-by-step process I followed for API integration in the Next.js project:

- **API Choosing:** I chose the API from Template 3 provided in the documentation.I used the provided API **https://template-03-api.vercel.app/api/products** was used to fetch product data. This endpoint provided essential details, including product titles, descriptions, prices, and category IDs.

- **API Documentation Review:** I reviewed the API documentation thoroughly to understand the endpoints.The documentation helped identify the necessary fields (e.g., product_title, price, category_id).

- **Setting Sanity API Call:** Here is the snip of Sanity/lib/sanity.js

- **Schema Revisions:** I updated the existing schema to ensure compatibility with the product data fetched from the API.

2. Adjustments Made to Schemas:

In order to store the product data in Sanity CMS, I had to adjust the existing schema to ensure compatibility with the data fetched from the API.

3.MIGRATION STEPS AND TOOLS :

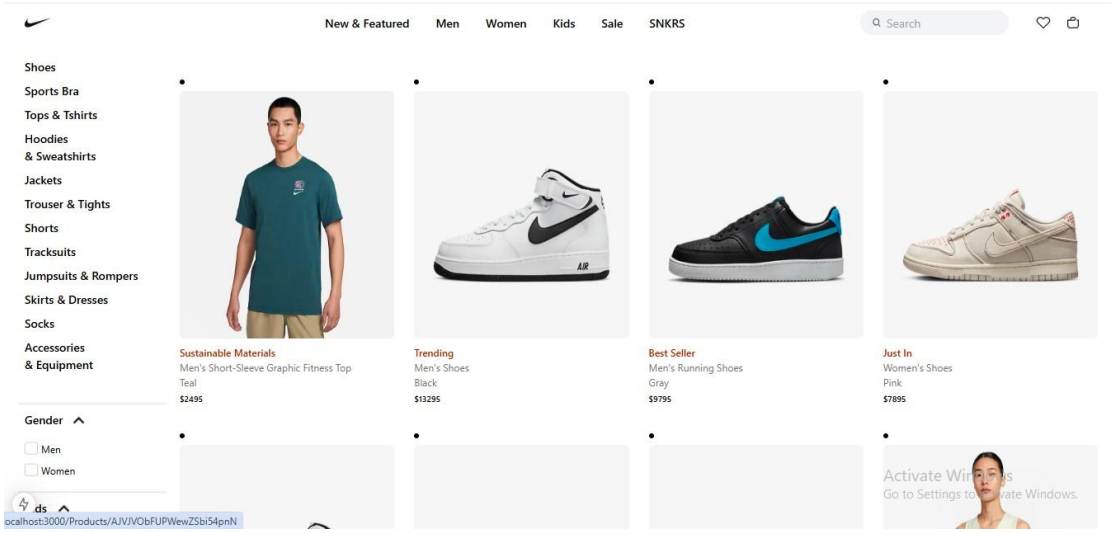
- **Data Migration:**

Migration Process: Using the provide migrating script to transferred the data of APIs into Sanity(CMS).The script fetched product data from the API, transformed it to match the Sanity schema, and then imported the data into the Sanity CMS.

- **Tools Used:** Sanity/Client.

FRONTENT

After migrating the into Sanity ,I created a dynamic responsive frontend in next.js project to display the data.



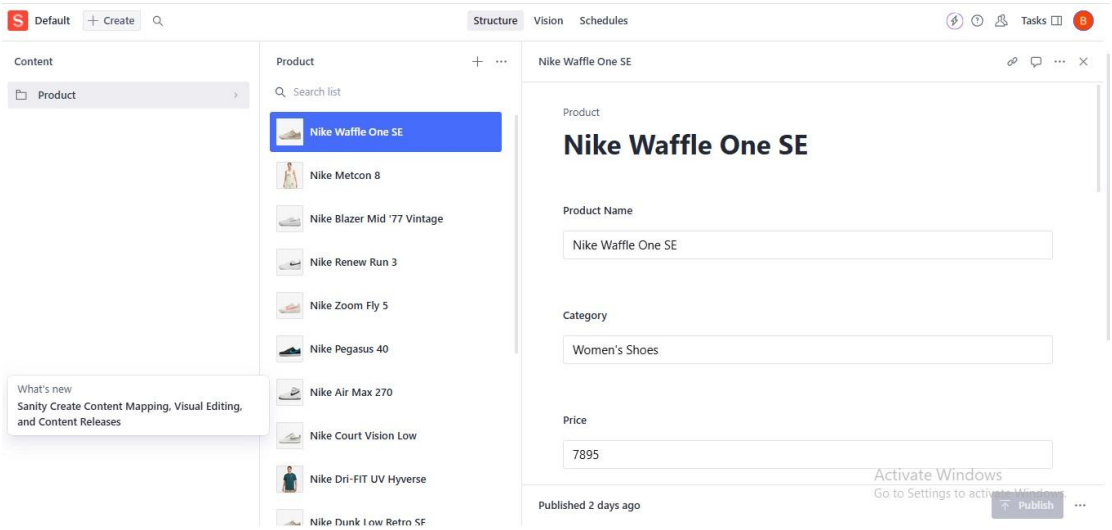
FETCHING PRODUCTS DATA:

I used sanity GROQ queries to fetch data directly from sanity(CMS) to my next.js project.

```tsx
import { fetchProducts } from '../../sanity/lib/sanity';
import imageUrlBuilder from '@sanity/image-url';
import { client } from '../../sanity/lib/sanity';
import Link from 'next/link';
import SideBar from '../Components/Side';


const builder = imageUrlBuilder(client);


const urlFor = (source: any) => builder.image(source).url();

interface Product {
  _id: string;
  name: string;
  price: number;
  description?: string;
  category: string;
  colors: string[];
  inventory: number;
  status: string;
  image: any;
}

const ProductsPage = async () => {
  const products: Product[] = await fetchProducts(); // Fetch the products

  return (
    <div className='flex'>
      <SideBar />
      <div className="px-4 py-8 ">
        <div className="grid grid-cols-2 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-6">
```

SANITY STUDIO INTERFACE:

After migrating the data script,the data appears in Sanity studio.



API CALLS:

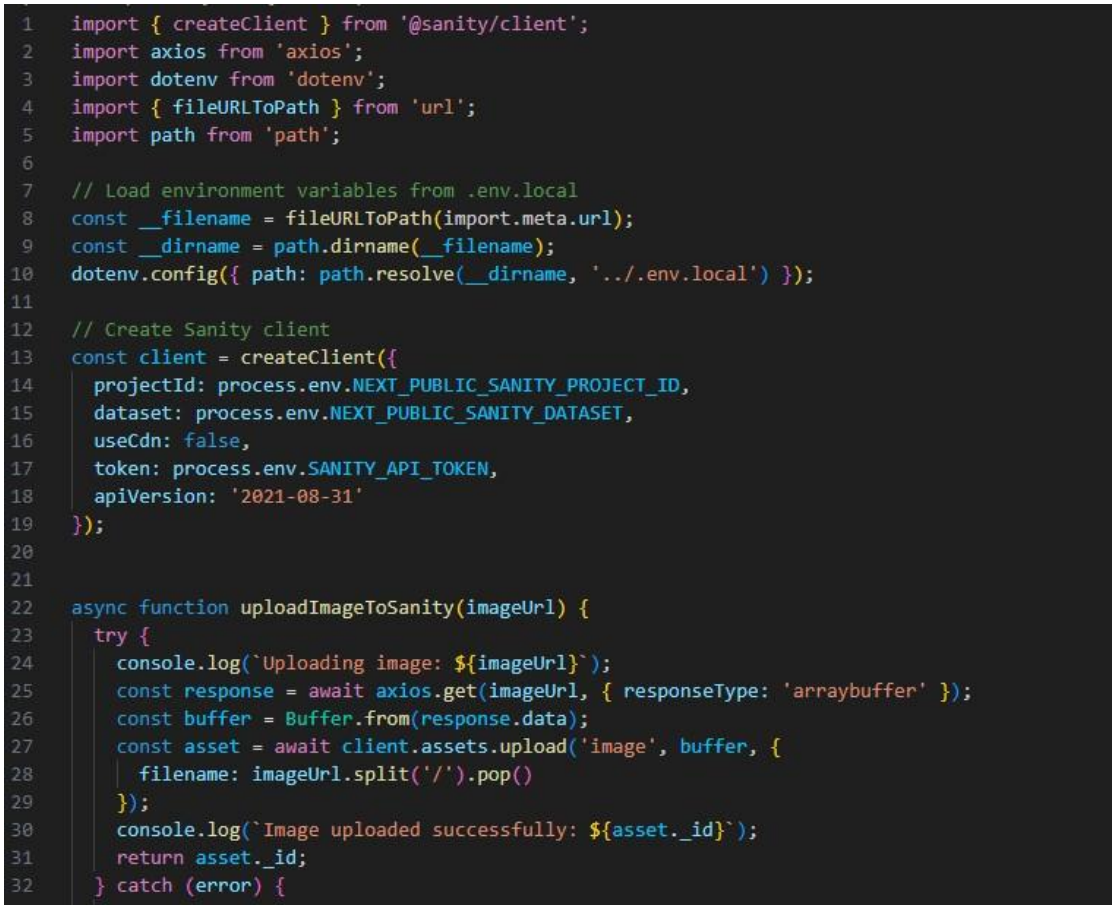
src > sanity > lib > JS sanity.js > ...

```
import sanityClient from '@sanity/client';


export const client = sanityClient({
    projectId: 'fxeltr1j',
    dataset: 'production',
    useCdn: process.env.NODE_ENV === 'production',
  });


export async function fetchProducts() {
  const query = '*[_type == "product"]';
  const products = await client.fetch(query);
  return products;
}


```

src > sanity > lib > JS sanity.js > ...

```
import sanityClient from '@sanity/client';


export const client = sanityClient({
    projectId: 'fxeltr1j',
    dataset: 'production',
    useCdn: process.env.NODE_ENV === 'production',
  });


export async function fetchProducts() {
  const query = '*[_type == "product"]';
  const products = await client.fetch(query);
  return products;
}


```

MIGRATING SCRIPT:



```
import { createClient } from '@sanity/client';
import axios from 'axios';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';
import path from 'path';

// Load environment variables from .env.local
const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);
dotenv.config({ path: path.resolve(__dirname, '../.env.local') });

// Create Sanity client
const client = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
  useCdn: false,
  token: process.env.SANITY_API_TOKEN,
  apiVersion: '2021-08-31'
});


async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading image: ${imageUrl}`);
    const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
    const buffer = Buffer.from(response.data);
    const asset = await client.assets.upload('image', buffer, {
      filename: imageUrl.split('/').pop()
    });
    console.log(`Image uploaded successfully: ${asset._id}`);
    return asset._id;
  } catch (error) {
```

Conclusion:

In this report, I have documented the process of integrating an external API into a Next.js project, adjusting the Sanity CMS schema, and migrating product data into the CMS. The project successfully fetched product data from the API, displayed it on the frontend, and populated the Sanity CMS with the data.