

NAME: AQSA ARSHAD RASHEED

ROLL NO : 00496448

Day 3 - API Integration Report - [Comforty Marketplace]

Introduction:

API Integration Process:

On Day 3, I focused on integrating APIs and migrating data into the Sanity CMS to build a functional backend for the Comforty Marketplace. I used the provided API (Template 8) to populate my Sanity CMS with product data, followed by integrating the data into the frontend.

Steps Followed:

1) Review API Documentation:

I thoroughly reviewed the API documentation for Template 6, which provided a URL to fetch product data:

Template 8 API Documentation.

2) Validate and Adjust Sanity

Schema: I compared the API data structure with my existing Sanity schema and adjusted the field names to ensure they matched. I used the following schema:

- The script was successfully executed to transfer the product data into Sanity.

Data Migration:

- **Migration Method Used:** I used the migration script provided in Template 8 to fetch the data from the API and populate the Sanity CMS.

[Migration Script](#)

- The script was successfully executed to transfer the product data into Sanity.

```

1 // Import environment variables from .env.local
2 import "dotenv/config";
3
4 // Import the Sanity client to interact with the Sanity backend
5 import { createClient } from "@sanity/client";
6
7 // Load required environment variables
8 const {
9   NEXT_PUBLIC_SANITY_PROJECT_ID, // Sanity project ID
10  NEXT_PUBLIC_SANITY_DATASET, // Sanity dataset (e.g., "production")
11  NEXT_PUBLIC_SANITY_AUTH_TOKEN, // Sanity API token
12  BASE_URL = "https://gialc-hackathon-template-08.vercel.app", // API base URL for products and categories
13 } = process.env;
14
15 // Check if the required environment variables are provided
16 if (!NEXT_PUBLIC_SANITY_PROJECT_ID || !NEXT_PUBLIC_SANITY_AUTH_TOKEN) {
17   console.error(
18     "Missing required environment variables. Please check your .env.local file."
19   );
20   process.exit(1); // Stop execution if variables are missing
21 }
22
23 // Create a Sanity client instance to interact with the target Sanity dataset
24 const targetClient = createClient({
25   projectId: NEXT_PUBLIC_SANITY_PROJECT_ID, // Your Sanity project ID
26   dataset: NEXT_PUBLIC_SANITY_DATASET || "production", // Default to "production" if not set
27   useCdn: false, // Disable CDN for real-time updates
28   apiVersion: "2023-01-01", // Sanity API version
29   token: NEXT_PUBLIC_SANITY_AUTH_TOKEN, // API token for authentication
30 });
31
32 // Function to upload an image to Sanity
33 async function uploadImageToSanity(imageUrl) {
34   try {
35     // Fetch the image from the provided URL
36     const response = await fetch(imageUrl);
37     if (!response.ok) throw new Error("Failed to fetch image: ${imageUrl}");
38
39     // Convert the image to a buffer (binary format)
40     const buffer = await response.arrayBuffer();
41
42     // Upload the image to Sanity and get its asset ID
43     const uploadedAsset = await targetClient.assets.upload(
44       "image",
45       Buffer.from(buffer),

```

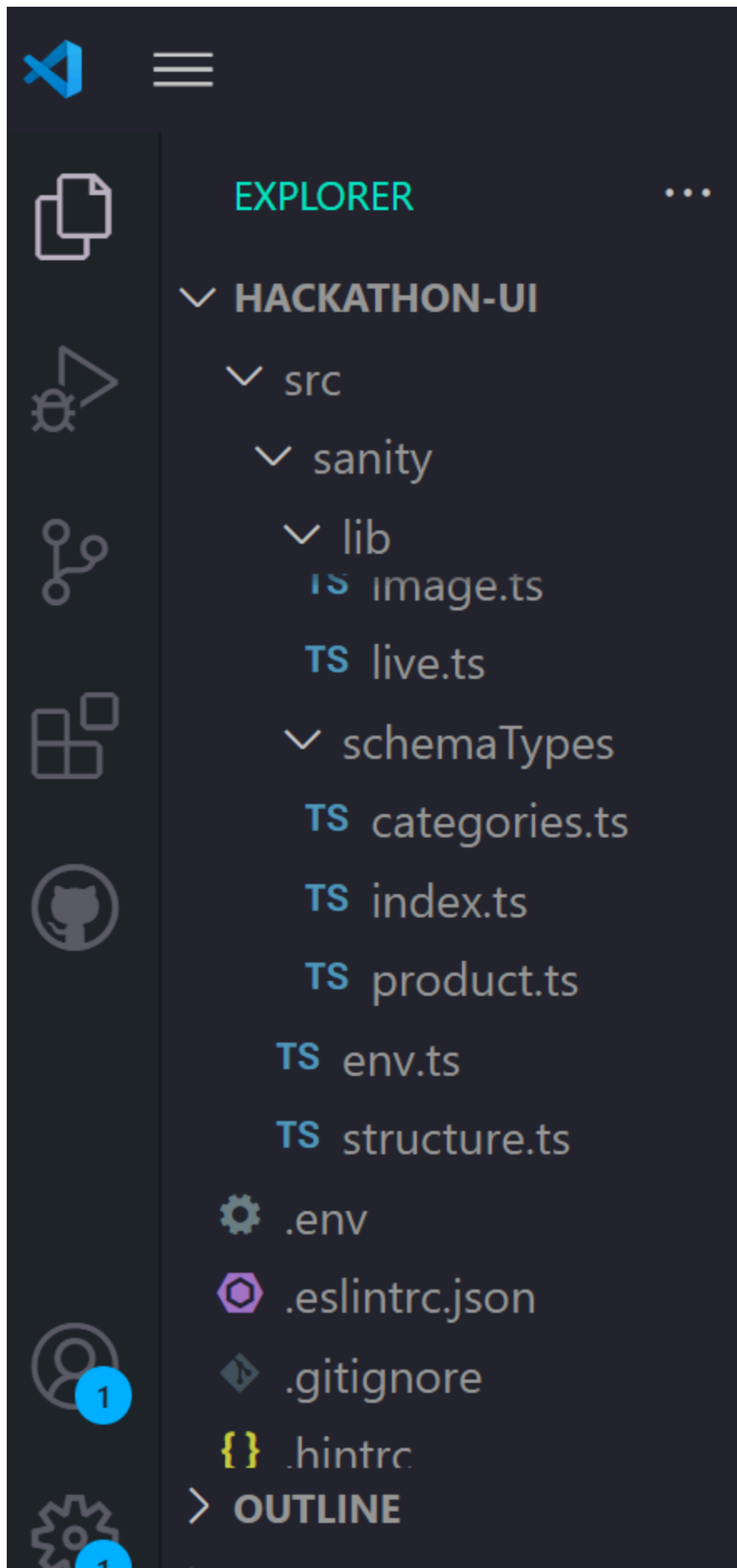
4) API Integration in Next.js:

- I created utility functions in my Next.js project to fetch data from the API and display product listings on the frontend.
- Data was successfully rendered in the product section of the homepage.

• Adjustments Made to Schema:

During the migration, I adjusted my Sanity schema to accommodate the product data. The field names were mapped from the API to the schema, such as:

- API field `product_title` was mapped to Sanity's `name`.
- The `product_price` field was mapped to `price` in Sanity.



Screenshots:

Sanity CMS Data Population:

