

Day 3 – API Integration
Report MarketplaCe

Comforty

Shabnam Wahid

API Integration Process

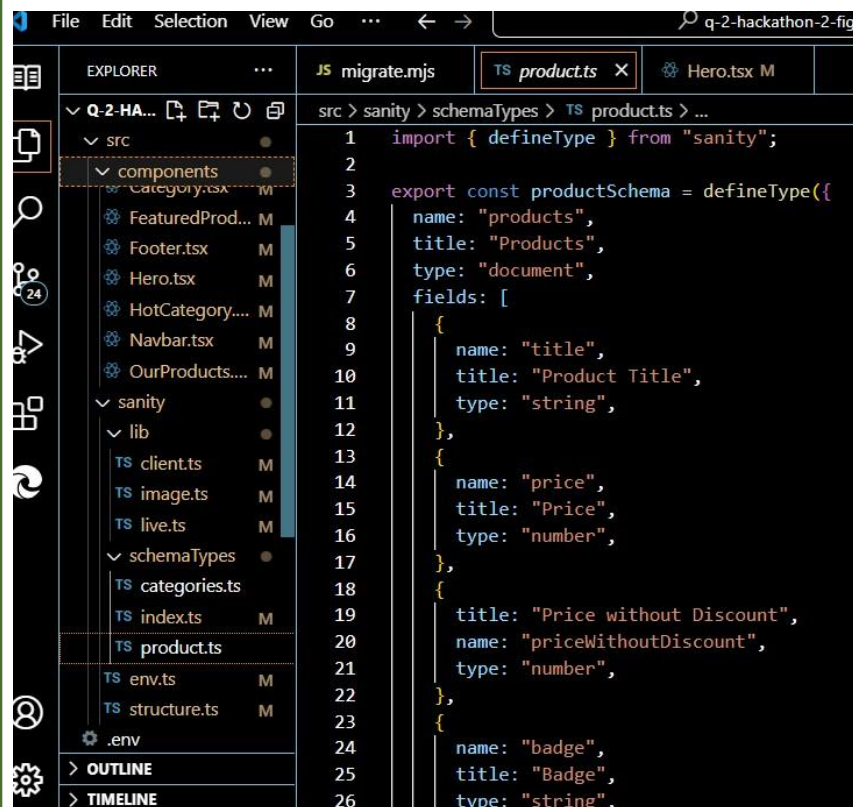
Objective:

Integrate Sanity APIs for dynamic data handling and create a seamless backend connection to display product and category data in the frontend.

Steps Followed:

Sanity Backend Setup:

- Created and configured the Sanity Studio using `sanity init`.
- Defined schemas for Products and Categories in `schemaTypes` folder



```
1 import { defineType } from "sanity";
2
3 export const productSchema = defineType({
4   name: "products",
5   title: "Products",
6   type: "document",
7   fields: [
8     {
9       name: "title",
10      title: "Product Title",
11      type: "string",
12    },
13    {
14      name: "price",
15      title: "Price",
16      type: "number",
17    },
18    {
19      title: "Price without Discount",
20      name: "priceWithoutDiscount",
21      type: "number",
22    },
23    {
24      name: "badge",
25      title: "Badge",
26      type: "string",
```

Data Fetching in Next.js

Implemented a Sanity client using `@sanity/client` for API communication:

```
JS migrate.mjs TS fetchProducts.ts U page.tsx M TS client.ts M X JS next.config.mj
src > sanity > lib > TS client.ts > ...
1 import { createClient } from 'next-sanity'
2
3 import { apiVersion, dataset, projectId } from '../env'
4
5 export const client = createClient({
6   projectId,
7   dataset,
8   apiVersion,
9   useCdn: true, // Set to false if statically generating pages, using ISR or
10 })
```

Called fetchProducts on the product page to display data dynamically.

Adjustments Made to Schemas

Added categories.ts schema for product categorization:

```
File Edit Selection View Go ... q-2-hackathon-2-fig
EXPLORER TS migrate.mjs TS products.ts X Hero.tsx M
Q-2-HA... src > sanity > schemaTypes > TS product.ts > ...
  components
    Category.tsx M
    FeaturedProd... M
    Footer.tsx M
    Hero.tsx M
    HotCategory... M
    Navbar.tsx M
    OurProducts... M
  sanity
    lib
      TS client.ts M
      TS image.ts M
      TS live.ts M
    schemaTypes
      TS categories.ts
      TS index.ts M
      TS product.ts
    TS env.ts M
    TS structure.ts M
    .env
  > OUTLINE
  > TIMELINE
1 import { defineType } from "sanity";
2
3 export const productSchema = defineType({
4   name: "products",
5   title: "Products",
6   type: "document",
7   fields: [
8     {
9       name: "title",
10      title: "Product Title",
11      type: "string",
12    },
13    {
14      name: "price",
15      title: "Price",
16      type: "number",
17    },
18    {
19      title: "Price without Discount",
20      name: "priceWithoutDiscount",
21      type: "number",
22    },
23    {
24      name: "badge",
25      title: "Badge",
26      type: "string",
```

Migration Steps and Tools Used

Created a migration script to transfer REST API data to Sanity:

- Added `dotenv` for managing environment variables.
- Wrote `migrate.mjs`:

```

1 // Import environment variables from .env.local
2 import dotenv from 'dotenv';
3
4 // Import the Sanity client to interact with the Sanity backend
5 import { createClient } from "@sanity/client";
6 dotenv.config();
7
8 // Load required environment variables
9 const {
10   NEXT_PUBLIC_SANITY_PROJECT_ID="usep95cz",
11   NEXT_PUBLIC_SANITY_DATASET="production",
12   NEXT_PUBLIC_SANITY_AUTH_TOKEN= "sanity.token",
13   BASE_URL = "https://giaic-hackathon-template-08.vercel.app", // API base URL
14 } = process.env;
15
16 // Check if the required environment variables are provided
17 if (!NEXT_PUBLIC_SANITY_PROJECT_ID || !NEXT_PUBLIC_SANITY_AUTH_TOKEN) {
18   console.error("Missing required environment variables. Please check your .env.");
19   process.exit(1); // Stop execution if variables are missing
20 }
21
22 // Create a Sanity client instance to interact with the target Sanity dataset
23 const targetClient = createClient({
24   projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID, // Environment variable
25   dataset: process.env.NEXT_PUBLIC_SANITY_DATASET || "production", // Default
26   useCdn: false, // Disable CDN for real-time updates
27 });

```

Frontend Data Display:

