Market Place Technical Foundation- Comforty(Template 8)

Name: Abdullah Shaikh Day-3 API Integration and showing Sanity Data in Frontend.

Sanity Installation:

I install sanity to keep data of products and categories in it.

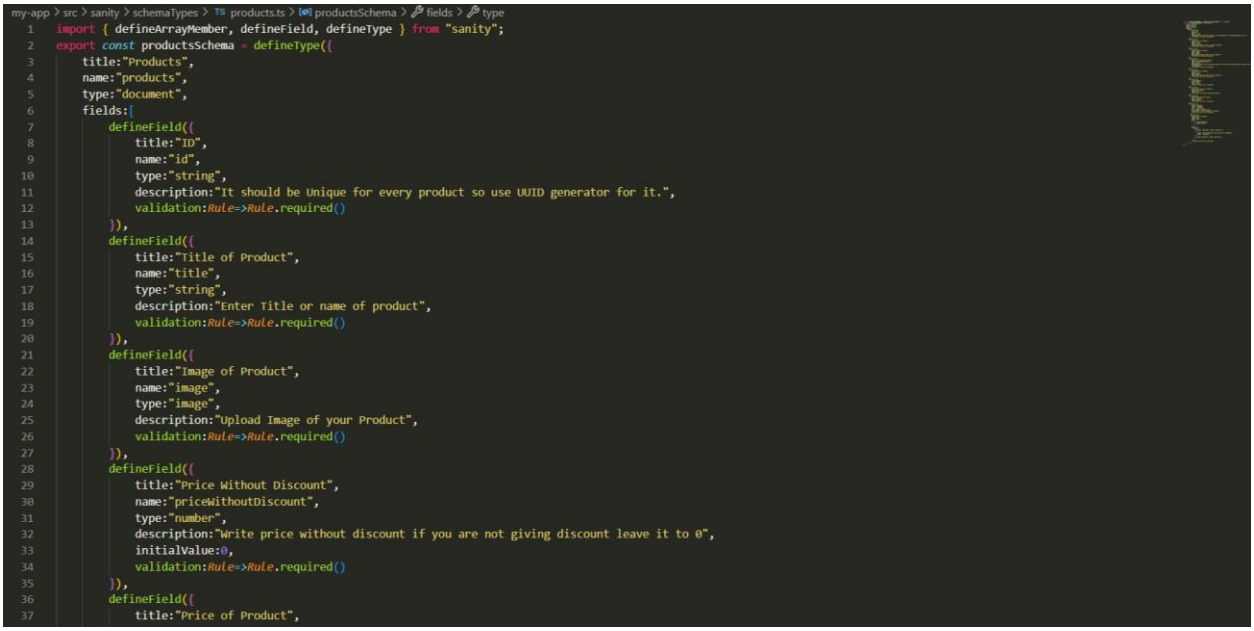
```
C:\Users\Dell\Documents\Market_Place_Builder_Hackathon_2025_GIAIC\my-app>npm create sanity@latest
Need to install the following packages:
create-sanity@3.70.0
Ok to proceed? (y) y
✓ You are logged in as abdullahkamran6601@gmail.com using Google
✓ Fetching existing projects

? Create a new project or select an existing one Market_Place_Builder_Hackathon_2025_GIAIC (59wvk5e2)
? Select dataset to use production
? Would you like to add configuration files for a Sanity project in this Next.js folder? Yes
⚠
⚠
⚠  It looks like you are using Next.js 15 and React 19
⚠  Please read our compatibility guide.
⚠  https://www.sanity.io/help/react-19
⚠
⚠
? Do you want to use TypeScript? Yes
? Would you like an embedded Sanity Studio? Yes
? What route do you want to use for the Studio? /studio
? Select project template to use Clean project with no predefined schema types
? Would you like to add the project ID and dataset to your .env.local file? Yes
Added http://localhost:3000 to CORS origins
Running 'npm install --legacy-peer-deps --save @sanity/vision@3 sanity@3 @sanity/image-url@1 styled-components@6'
npm WARN deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and tested way to coalesce async reques
ts by a key value, which is much more comprehensive and powerful.
npm WARN deprecated @humanwhocodes/config-array@0.13.0: Use @eslint/config-array instead
npm WARN deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported
npm WARN deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm WARN deprecated @humanwhocodes/object-schema@2.0.3: Use @eslint/object-schema instead
npm WARN deprecated @sanity/block-tools@3.70.0: Renamed - use `@portabletext/block-tools` instead. `@sanity/block-tools` will no longer receive updates.
npm WARN deprecated eslint@8.57.1: This version is no longer supported. Please see https://eslint.org/version-support for other options.

added 1298 packages, and audited 1299 packages in 4m

244 packages are looking for funding
  run `npm fund` for details

1 moderate severity vulnerability
```

```
To address all issues, run:
  npm audit fix --force

Run `npm audit` for details.

Success! Your Sanity configuration files has been added to this project
```

Making Schemas:

Then, I make schemas for products and categories:

Products:



```
        defineField({
            title:"Price of Product",
            name:"price",
            type:"number",
            description:"Write actual price of product",
            validation:Rule=>Rule.required()
        }),
        defineField({
            title:"Badge?",
            name:"badge",
            type:"string",
            validation:Rule=>Rule.required()
        }),
        defineField({
            title:"Description of product",
            name:"description",
            type:"text",
            validation:Rule=>Rule.required().max(600)
        }),
        defineField({
            title:"Inventory Of Product",
            name:"inventory",
            type:"number",
            validation:Rule=>Rule.required()
        }),
        defineField({
            name: "category",
            title: "Category",
            type: "reference",
            to: [{ type: "category" }],
            description:"Choose Category of product",
            validation:Rule=>Rule.required()
         }),
         defineField({
```

```
         defineField({
           title:"Tags of Product",
           name:"tags",
           type:"array",
           of:[
               defineArrayMember({
                   type:"string"
               })
           ],
           options: {
               list: [
                 { title: "Featured", value: "featured" },
                 {
                   title: "Follow products and discounts on Instagram",
                   value: "instagram",
                 },
                 { title: "Gallery", value: "gallery" },
               ],
             },
           validation:Rule=>Rule.required()
       }),
   ]
})
```

Categories:

```
my-app > src > sanity > schemaTypes > TS categories.ts > [e] categoriesSchema
import { defineField, defineType } from "sanity";

export const categoriesSchema = defineType({
    title:"Categories",
    name:"categories",
    type:"document",
    fields:[
        defineField({
            title:"ID",
            name:"id",
            type:"string",
            description:"It should be Unique for every category so use UUID generator for it.",
            validation:Rule=>Rule.required()
        }),
        defineField({
            title:"Title of Category",
            name:"title",
            type:"string",
            description:"Enter Title or name of category",
            validation:Rule=>Rule.required()
        }),
        defineField({
            title:"Image of Product",
            name:"image",
            type:"image",
            description:"Upload Feature Image of your category",
            validation:Rule=>Rule.required()
        }),
    ]
})
```

ENV:

I make env file and make environment variables in it

```
1. NEXT_PUBLIC_SANITY_PROJECT_ID
2. NEXT_PUBLIC_SANITY_DATASET
3. NEXT_PUBLIC_SANITY_AUTH_TOKEN
4. baseUrl
```

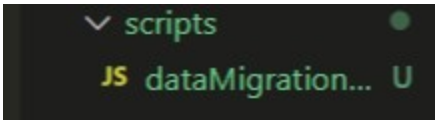
Data Migration:

For mock data migration I use these api's for it.

https://giaic-hackathon-template-08.vercel.app/api/products.

https://giaic-hackathon-template-08.vercel.app/api/categories

To migrate it I make migration file in scripts folder:



and then write this code in it

```
// Import environment variables from .env.local import
"dotenv/config";


// Import the Sanity client to interact with the Sanity backend
import { createClient } from "@sanity/client";


// Load required environment variables const
{
  NEXT_PUBLIC_SANITY_PROJECT_ID, // Sanity project ID
  NEXT_PUBLIC_SANITY_DATASET, // Sanity dataset (e.g., "production")
  NEXT_PUBLIC_SANITY_AUTH_TOKEN, // Sanity API token
```

```
  BASE_URL = "https://giaic-hackathon-template-08.vercel.app", // API base URL
for products and categories
} = process.env;


// Check if the required environment variables are provided if
(!NEXT_PUBLIC_SANITY_PROJECT_ID || !NEXT_PUBLIC_SANITY_AUTH_TOKEN) {
console.error("Missing required environment variables. Please check your
.env.local file.");   process.exit(1); // Stop execution if
variables are missing }


// Create a Sanity client instance to interact with the target Sanity dataset
const targetClient = createClient({   projectId: NEXT_PUBLIC_SANITY_PROJECT_ID,
// Your Sanity project ID   dataset: NEXT_PUBLIC_SANITY_DATASET || "production",
// Default to "production" if not set   useCdn: false, // Disable CDN for real-
time updates   apiVersion: "2023-01-01", // Sanity API version   token:
NEXT_PUBLIC_SANITY_AUTH_TOKEN, // API token for authentication });


// Function to upload an image to Sanity
async function uploadImageToSanity(imageUrl)
{   try {
    // Fetch the image from the provided URL     const response = await
fetch(imageUrl);     if (!response.ok) throw new Error(`Failed to fetch
image: ${imageUrl}`);
    // Convert the image to a buffer (binary format)
const buffer = await response.arrayBuffer();


    // Upload the image to Sanity and get its asset ID
    const uploadedAsset = await targetClient.assets.upload("image",
Buffer.from(buffer), {
      filename: imageUrl.split("/").pop(), // Use the file name from the URL
});
    return uploadedAsset._id; // Return the asset ID
  } catch (error) {     console.error("Error uploading
image:", error.message);     return null; // Return null
if the upload fails
  }
}


// Main function to migrate data from REST API to Sanity
```

```
async function migrateData() {
console.log("Starting data migration...");
   try
{
    // Fetch categories from the REST API     const categoriesResponse = await
fetch(`${BASE_URL}/api/categories`);     if (!categoriesResponse.ok) throw new
Error("Failed to fetch categories.");     const categoriesData = await
categoriesResponse.json(); // Parse response to JSON


    // Fetch products from the REST API     const productsResponse = await
fetch(`${BASE_URL}/api/products`);     if (!productsResponse.ok) throw new
Error("Failed to fetch products.");     const productsData = await
productsResponse.json(); // Parse response to JSON
     const categoryIdMap = {}; // Map to store migrated category IDs
    // Migrate categories     for (const category of categoriesData) {
console.log(`Migrating category: ${category.title}`);       const imageId =
await uploadImageToSanity(category.imageUrl); // Upload category image


      // Prepare the new category object       const newCategory =
{         id: category._id, // Use the same ID for reference
mapping
        _type: "category",         title: category.title,
image: imageId ? { _type: "image", asset: { _ref: imageId } } :
undefined, // Add image if uploaded
      };


      // Save the category to Sanity
      // const result = await targetClient.create(newCategory);
      // categoryIdMap[category._id] = result.id; // Store the new category ID
      // console.log(`Migrated category: ${category.title} (ID: ${result.id})`);
    }


    // Migrate products     for (const
product of productsData) {
      console.log(`Migrating product: ${product.title}`);
      const  imageId  =  await  uploadImageToSanity(product.imageUrl);  //  Upload
product image


      // Prepare the new product object
```

```
      const newProduct = {         _type: "product",
id:product._id,         title: product.title,         price:
product.price,         priceWithoutDiscount:
product.priceWithoutDiscount,         badge: product.badge,
image: imageId ? { _type: "image", asset: { _ref: imageId } } :
undefined, // Add image if uploaded
category: {
          _type: "reference",
          _ref: categoryIdMap[product.category._id], // Use the migrated category
ID
        },         description:
product.description,         inventory:
product.inventory,         tags:
product.tags,
      };


      // Save the product to Sanity       const result = await
targetClient.create(newProduct);       console.log(`Migrated product:
${product.title} (ID: ${result._id})`);     }      console.log("Data
migration completed successfully!");
  } catch (error) {     console.error("Error during
migration:", error.message);     process.exit(1); // Stop
execution if an error occurs
  }
}


// Start the migration process
migrateData();
```

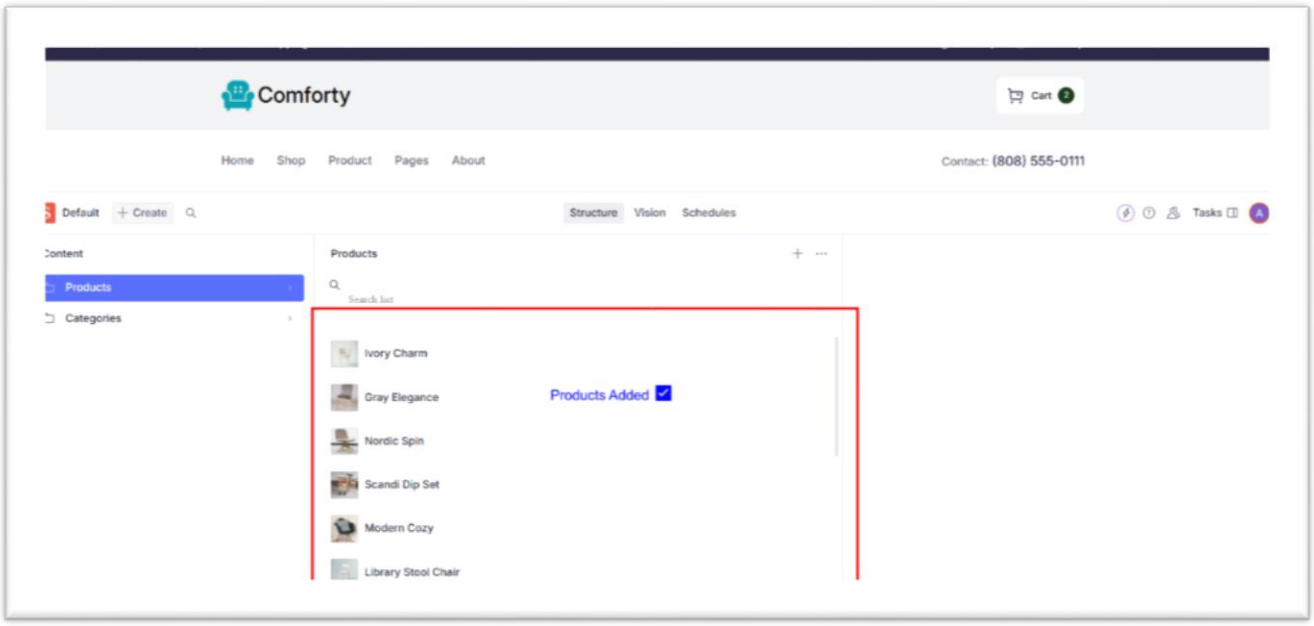
and then add this in

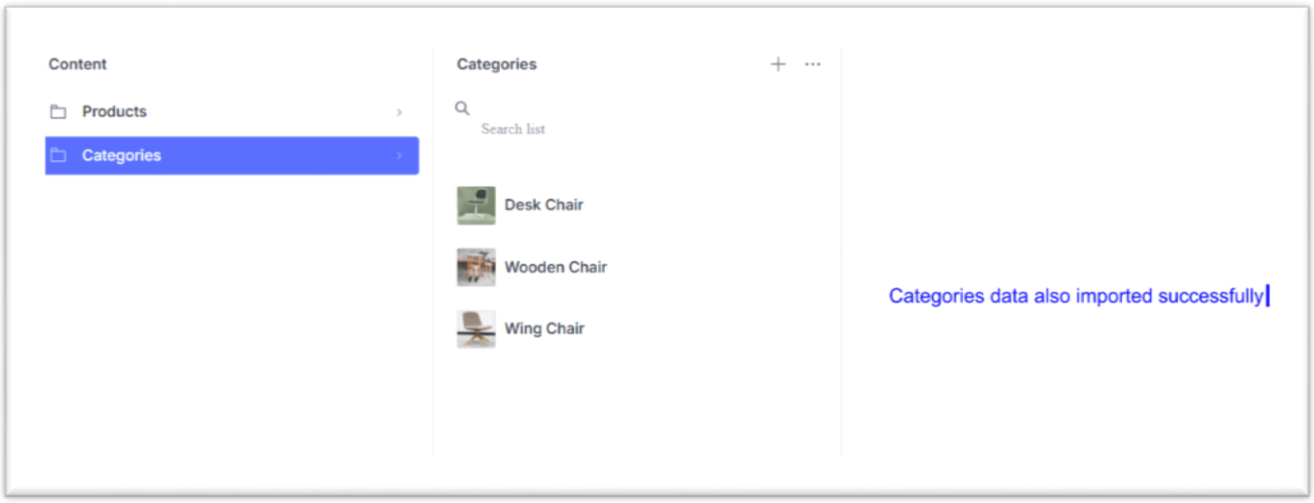
package.json:

```
"scripts": {
  "dev": "next dev --turbopack",
  "build": "next build",
  "start": "next start",
  "lint": "next lint",
  "migrateData": "node scripts/dataMigration.mjs"
},
```

And then run npm run migrateData to migrate Data and data migrated completely.

```
C:\Users\Dell\Documents\Market_Place_Builder_Hackathon_2025_GIAIC\my-app>npm run migrateData

> my-app@0.1.0 migrateData
> node scripts/dataMigration.mjs

Starting data migration...
Migrating category: Wing Chair
Migrated category: Wing Chair (ID: 26fd7176-3c4d-40fc-a73a-3b85a9b5e15f)
Migrating category: Wooden Chair
Migrated category: Wooden Chair (ID: 407a8583-6203-4f61-becf-8e8b4c5461b6)
Migrating category: Desk Chair
Migrated category: Desk Chair (ID: b5710116-09af-4d0e-aa9a-dcd02fe919a9)
Migrating product: SleekSpin
Migrated product: SleekSpin (ID: eW4V35CliPqhGTiVkCR4Zp)
Migrating product: Citrus Edge
Migrated product: Citrus Edge (ID: tiTonAuk6WGSOb8ZgZeAJG)
Migrating product: Rose Luxe Armchair
Migrated product: Rose Luxe Armchair (ID: tiTonAuk6WGSOb8ZgZeANo)
Migrating product: Library Stool Chair
Migrated product: Library Stool Chair (ID: eW4V35CliPqhGTiVkCR4jN)
Migrating product: Modern Cozy
Migrated product: Modern Cozy (ID: tiTonAuk6WGSOb8ZgZeASM)
Migrating product: Scandi Dip Set
Migrated product: Scandi Dip Set (ID: EJwKv1AMm1jkttyWGN7Dkz)
Migrating product: Citrus Edge
Migrated product: Citrus Edge (ID: tiTonAuk6WGSOb8ZgZeAZB)
Migrating product: Rose Luxe Armchair
Migrated product: Rose Luxe Armchair (ID: tiTonAuk6WGSOb8ZgZeAcy)
Migrating product: SleekSpin
Migrated product: SleekSpin (ID: tiTonAuk6WGSOb8ZgZeAfF)
Migrating product: Library Stool Chair
Migrated product: Library Stool Chair (ID: tiTonAuk6WGSOb8ZgZeAhW)
Migrating product: Modern Cozy
Migrated product: Modern Cozy (ID: eW4V35CliPqhGTiVkCR5Sj)
Migrating product: Scandi Dip Set
Migrated product: Scandi Dip Set (ID: tiTonAuk6WGSOb8ZgZeAlJ)
Migrating product: Nordic Spin
Migrated product: Nordic Spin (ID: eW4V35CliPqhGTiVkCR5xl)
Migrating product: Gray Elegance
Migrated product: Gray Elegance (ID: eW4V35CliPqhGTiVkCR6JF)
Migrating product: Ivory Charm
Migrated product: Ivory Charm (ID: EJwKv1AMm1jkttyWGN7HBF)
Data migration completed successfully!
```





API'S:

And then I make API's for data fetching from Sanity:

S.NO	PATH	PURPOSE	METHOD	CODE

1	/api/fetch /allProducts	To Fetch All products from Sanity	GET	import { NextResponse } from "next/server";
				import { client } from "@/sanity/lib/client";
				
				export async function GET() {
				try{
				const query = `*[_type=="product"]{badge,image,title,priceWithoutDiscount,id,price,id}`;
				const res = await client.fetch(query);
				return NextResponse.json(res)
				}
				catch (error) {
				console.log(error)
				return new Response(
				JSON.stringify({ error: "Failed to fetch all Products" }),
				{
				status: 500,
				headers: { "Content-Type": "application/json" },
				}
				);
				}
				}

2	/api/ fetch/ featured Products	To fetch featured products from sanity	GET	import { NextResponse } from "next/server"; import { client } from "@/sanity/lib/client"; export async function GET() { try{ const query = `*[_type=="product" && "featured" in tags]{ badge,image,title,priceWithoutDiscount,id,price }`; const res = await client.fetch(query); return NextResponse.json(res) } catch (error) { console.log(error) return new Response(JSON.stringify({ error: "Failed to fetch Featured Products" }), { status: 500, headers: { "Content-Type": "application/json" }, }); } }

3	/api/fetch /ourProducts	Fetch Products for ourProducts Section	GET	import { NextResponse } from "next/server"; import { client } from "@/sanity/lib/client"; export async function GET() { try{ const query = `*[_type=="product"]{badge,image,title,priceWithoutDiscount,id,price,id} [0..7]`; const res = await client.fetch(query); return NextResponse.json(res) } catch (error) { console.log(error) return new Response(JSON.stringify({ error: "Failed to fetch" }), { status: 500, headers: { "Content-Type": "application/json" }, }); } }

And I also show data to front-end but that is part of day 4 so, we look at on that day.