Marketplace E-commerce

By MINAL SALEEM Date

This documentation will guide you through the setup and implementation of a Sanity schema, a migration script for transferring data to another Sanity account, and overall project setup.

1:Install Nextjs Project:

Command: npx create-next-app@14.2.20 .

2: Sanity:

Create a new project of sanity in sanity dashboard
Then run the project installation command in your next js project.

3: Sanity Schema:

Create a new project of sanity in sanity dashboard
1: After the Installation navigate to schema folder:
`/src/sanity/schemaTypes`.

2: Create two new files and add the following code:

`products.ts`

```
import { defineType } from "sanity";



export const productSchema = defineType({

  name: "products",

  title: "Products",

  type: "document",

  fields: [

    {

      name: "title",

      title: "Product Title",

      type: "string",

    },

    {

      name: "price",

      title: "Price",

      type: "number",

    },

    {

      title: "Price without Discount",

      name: "priceWithoutDiscount",

      type: "number",

    },

    {

      name: "badge",
```

```
    title: "Badge",
    type: "string",
  },
  {
    name: "image",
    title: "Product Image",
    type: "image",
  },
  {
    name: "category",
    title: "Category",
    type: "reference",
    to: [{ type: "categories" }],
  },
  {
    name: "description",
    title: "Product Description",
    type: "text",
  },
  {
    name: "inventory",
    title: "Inventory Management",
    type: "number",
  },
  {
    name: "tags",
```

```
      title: "Tags",
      type: "array",
      of: [{ type: "string" }],
      options: {
        list: [
          { title: "Featured", value: "featured" },
          {
            title: "Follow products and discounts on Instagram",
            value: "instagram",
          },
          { title: "Gallery", value: "gallery" },
        ],
      },
    },
  ],
});
```

- categories.ts

```
import { defineType } from "sanity";


export const categorySchema = defineType({
    name: 'categories',
    title: 'Categories',
    type: 'document',
```

```
    fields: [
        {
            name: 'title',
            title: 'Category Title',
            type: 'string',
        },
        {
            name: 'image',
            title: 'Category Image',
            type: 'image',
        },
        {
            title: 'Number of Products',
            name: 'products',
            type: 'number',
        }
    ],
});
```

2. **Importing Schemas in index.ts**
    - src/sanity/schemaTypes/index.ts

```
import { type SchemaTypeDefinition } from 'sanity'
import { productSchema } from "./products";
import { categorySchema } from "./categories";

export const schema: { types: SchemaTypeDefinition[] } = {
  types: [
    productSchema,
```

```
    categorySchema
  ],
}
```

4: Environment Variables:

Create .env file and add the following variables:

```
NEXT_PUBLIC_SANITY_PROJECT_ID="<Project ID>"

NEXT_PUBLIC_SANITY_DATASET="production"

NEXT_PUBLIC_SANITY_AUTH_TOKEN="<Auth Token>"
```

5: Data Migration Script:

.When you pull these changes, ensuring consistency between your Sanity dataset and the API's data structure.data from an external API, the structure of the API data may change over time, or you may need to adapt your local schema to accommodate new fields, relationships, or types. Migration files allow you to systematically manage

1: Create a folder scripts/migrate.mjs :

```
// 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: "categories",
        title: category.title,
        image: imageId ? { _type: "image", asset: { _ref: imageId } } :
undefined, // Add image if uploaded
      };

      // Save the category to Sanity
      const result = await targetClient.createOrReplace(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: "products",
        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();
```

2: Open`package.json` file and add the following code inside of scripts:

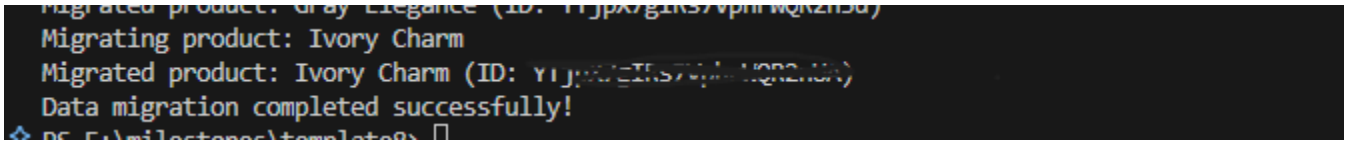
```
"name": "template8",
  "version": "0.1.0",
  "private": true,
  "scripts": {
    "dev": "next dev",
    "build": "next build",
    "start": "next start",
    "lint": "next lint",
    "migrate": "node scripts/migrate.mjs"
  },
```

3: Install the following package before running the script

npm install dotenv

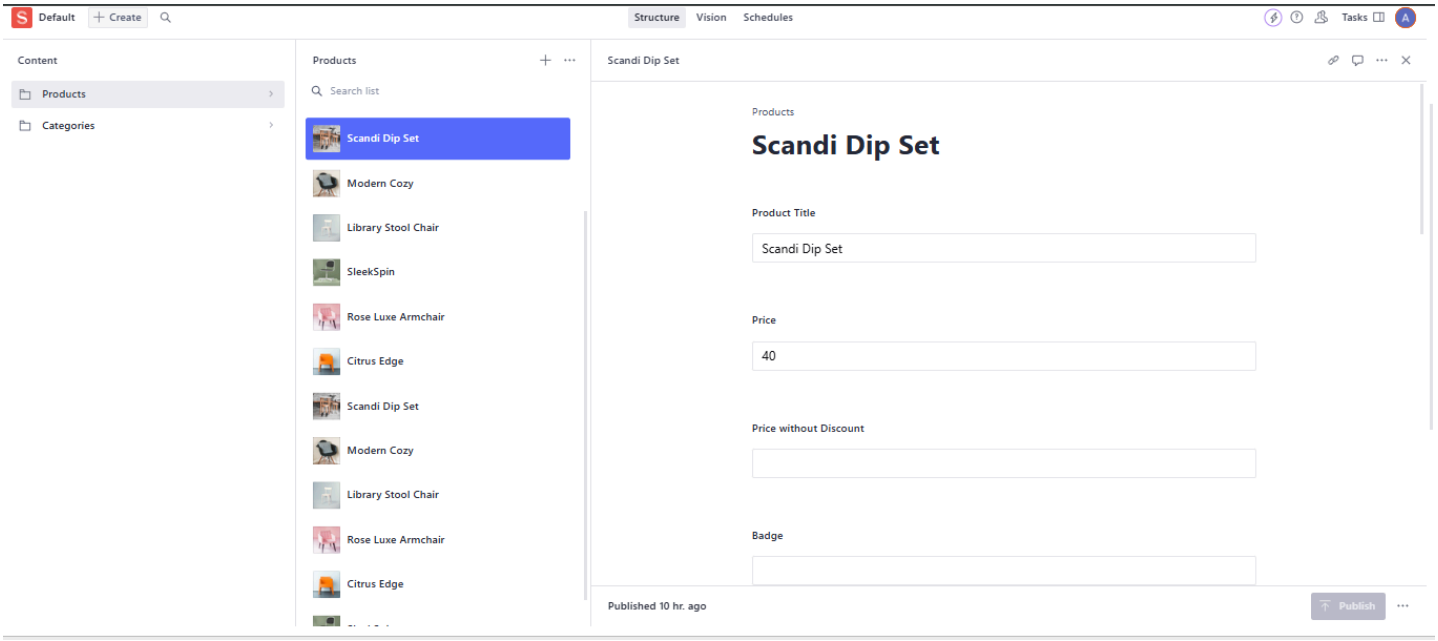
4: **Now run the command npm run migrate**

In terminal success message shows at end:



6: Sanity Studio:

the data shows in sanity studio from the rest api.



Write Query for fetch data from sanity to ui

Now UI looks like this: