API Integration Process

1. **Understanding API Endpoints**:
   - **Products API**: https://giaic-hackathon-template-08.vercel.app/api/products.
   - **Categories API**: https://giaic-hackathon-template-08.vercel.app/api/categories.
2. **Fetching Data**:
   - Used the provided REST API endpoints to retrieve product and category data.
   - Axios or Fetch API was utilized to integrate these APIs in the migration script.
3. **Environment Configuration**:
   - Added the following environment variables in the .env file: env

     ```
     NEXT_PUBLIC_SANITY_PROJECT_ID="<Project ID>"
     NEXT_PUBLIC_SANITY_DATASET="production"
     NEXT_PUBLIC_SANITY_AUTH_TOKEN="<Auth Token>"
     ```

4. **Integration with Sanity**:
   - The migration script connected to Sanity CMS via Sanity's API using the NEXT_PUBLIC_SANITY_AUTH_TOKEN.
   - Imported data was formatted to match the structure defined in the schemas.

Adjustments Made to Schemas

1. **Products Schema (products.ts)**:
   - Added fields for:
     - title: String.
     - price: Number.
     - description: String.
     - categories: Reference to categories.
2. **Categories Schema (categories.ts)**:
   - Added fields for:
     - name: String.
     - slug: String (for URL-friendly category names).
3. **Updated Index Schema (index.ts)**:
   - Imported and registered the products and categories schemas:

     ```
     import products from './products';
     import categories from './categories';

     export const schemaTypes = [products, categories];
     ```

Migration Steps and Tools Used

1. **Tools Used**:
   - **Sanity CLI**: To manage datasets and configuration.
   - **Node.js**: For executing the migration script.
   - **dotenv**: For loading environment variables.
2. **Steps**:
   1. **Script Setup**:
      - Created migrate.mjs in the /scripts folder.
      - Added logic to fetch data from the REST APIs and upload it to Sanity Studio.
   2. **Environment Preparation**:
      - Set environment variables in the .env file.
      - Installed necessary dependencies with npm install dotenv.
   3. **Execution**:
      - Ran the migration script using:npm run migrate
      - Verified that data was successfully migrated into the Sanity dataset.

- Document Title: "Day 3 - API Integration Report - [Your Marketplace Name]"

- What to Submit:

o A report documenting:

♣ API integration process.

♣ Adjustments made to schemas.

♣ Migration steps and tools used.

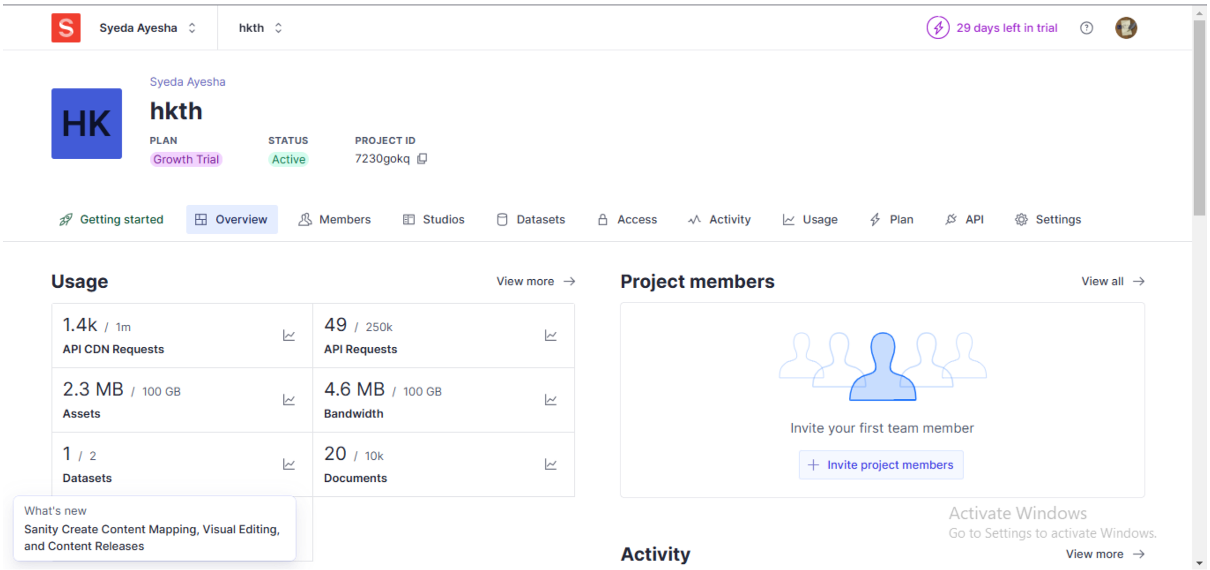
o Screenshots of:

♣ API calls.

♣ Data successfully displayed in the frontend.

♣ Populated Sanity CMS fields.

Screenshots



Syeda Ayesha

hkth

PLAN: Growth Trial

STATUS: Active

PROJECT ID: 7230gokq

Usage

1.4k / 1m API CDN Requests	49 / 250k API Requests
2.3 MB / 100 GB Assets	4.6 MB / 100 GB Bandwidth
1 / 2 Datasets	20 / 10k Documents

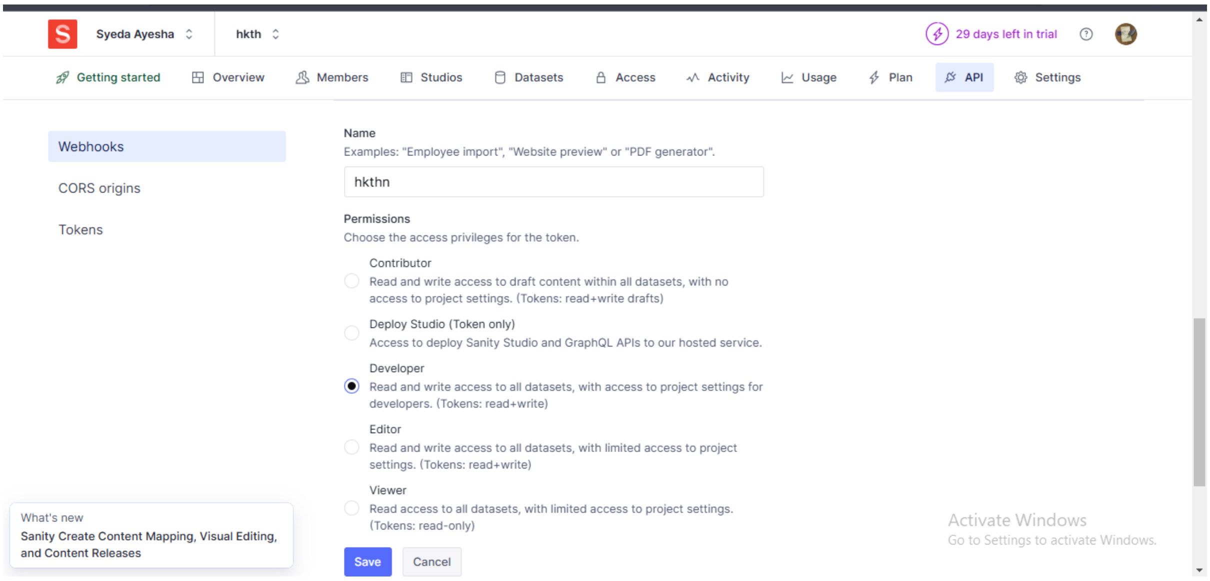
What's new

Sanity Create Content Mapping, Visual Editing, and Content Releases

Project members

Invite your first team member

Activity



Name

Examples: "Employee import", "Website preview" or "PDF generator".

hkthn

Permissions

Choose the access privileges for the token.

○ Contributor
Read and write access to draft content within all datasets, with no access to project settings. (Tokens: read+write drafts)

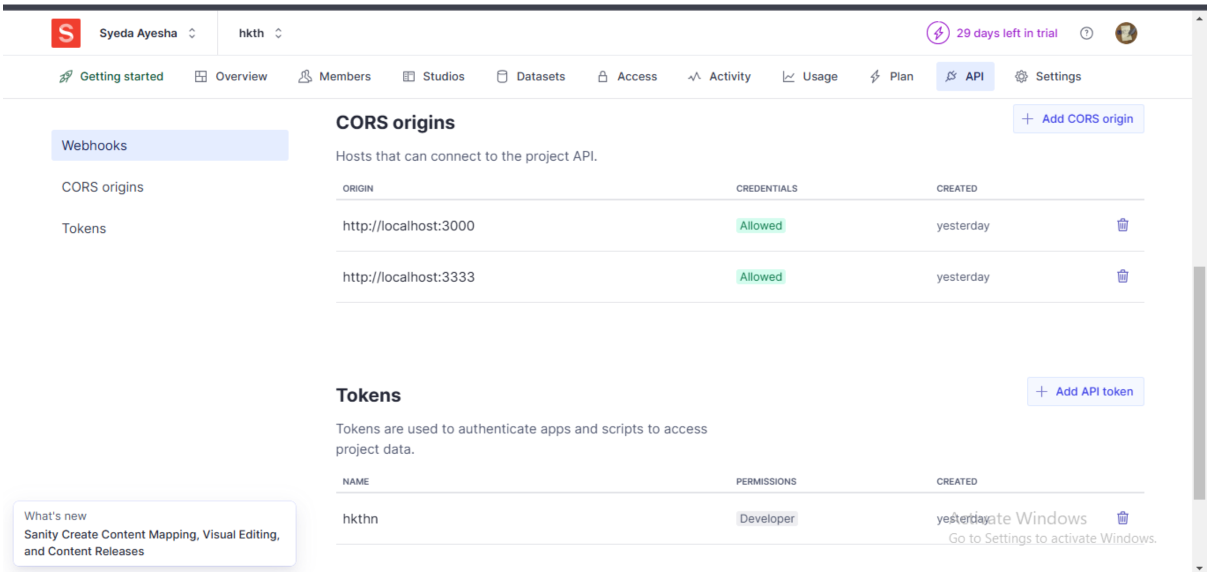
○ Deploy Studio (Token only)
Access to deploy Sanity Studio and GraphQL APIs to our hosted service.

● Developer
Read and write access to all datasets, with access to project settings for developers. (Tokens: read+write)

○ Editor
Read and write access to all datasets, with limited access to project settings. (Tokens: read+write)

○ Viewer
Read access to all datasets, with limited access to project settings. (Tokens: read-only)

Save
Cancel
What's new
Sanity Create Content Mapping, Visual Editing, and Content Releases
Activate Windows
Go to Settings to activate Windows.



CORS origins

Hosts that can connect to the project API.

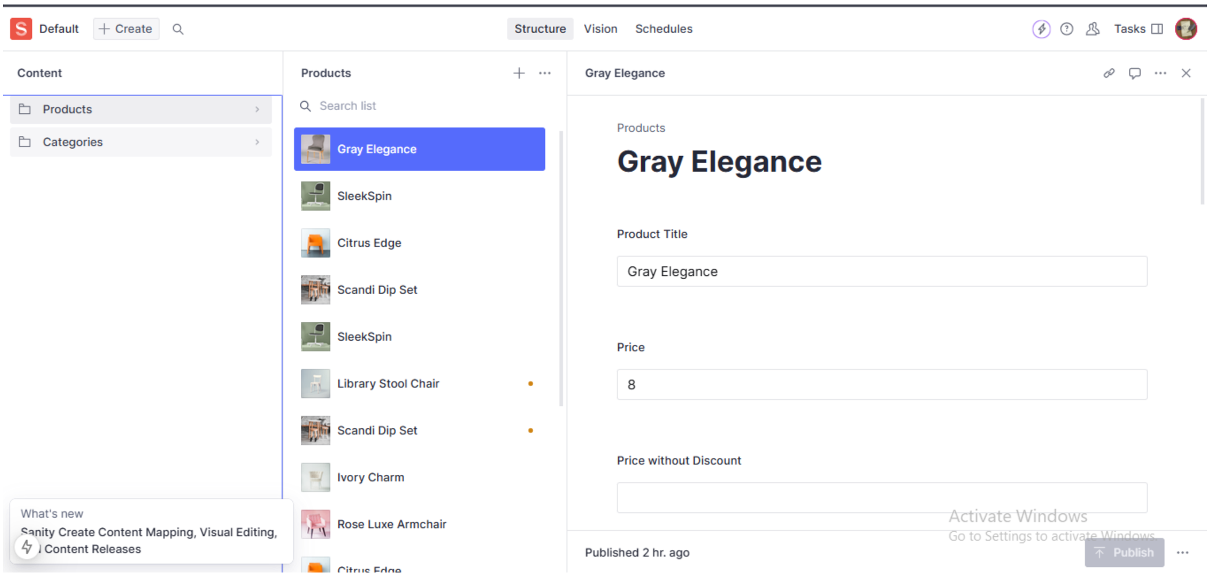
ORIGIN	CREDENTIALS	CREATED
http://localhost:3000	Allowed	yesterday
http://localhost:3333	Allowed	yesterday

Tokens

Tokens are used to authenticate apps and scripts to access project data.

NAME	PERMISSIONS	CREATED
hkthn	Developer	yesterday

What's new
Sanity Create Content Mapping, Visual Editing, and Content Releases
Activate Windows
Go to Settings to activate Windows.



Products

Gray Elegance

Product Title

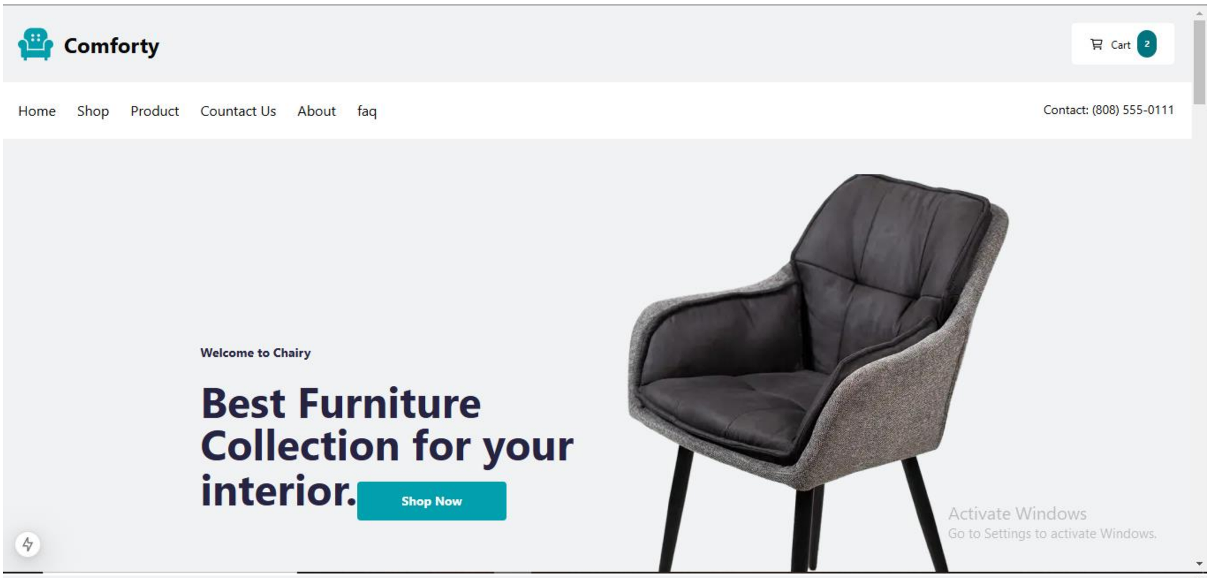
Gray Elegance

Price

8

Price without Discount

Published 2 hr. ago



Comforty

Cart 2

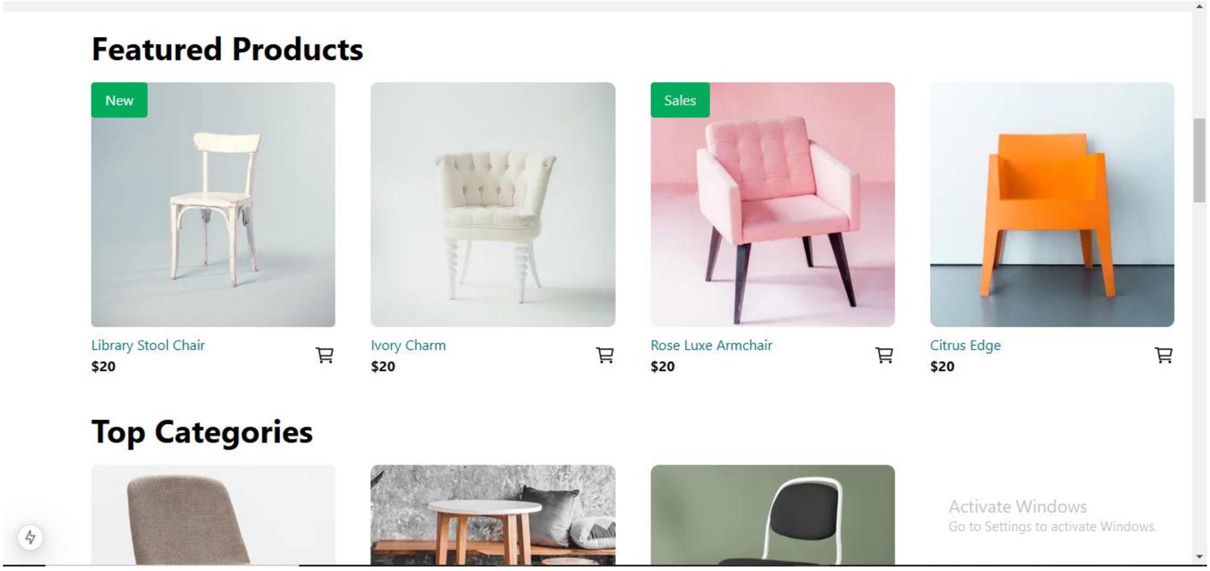
Home Shop Product Countact Us About faq

Contact: (808) 555-0111

Welcome to Chairy

Best Furniture Collection for your interior.

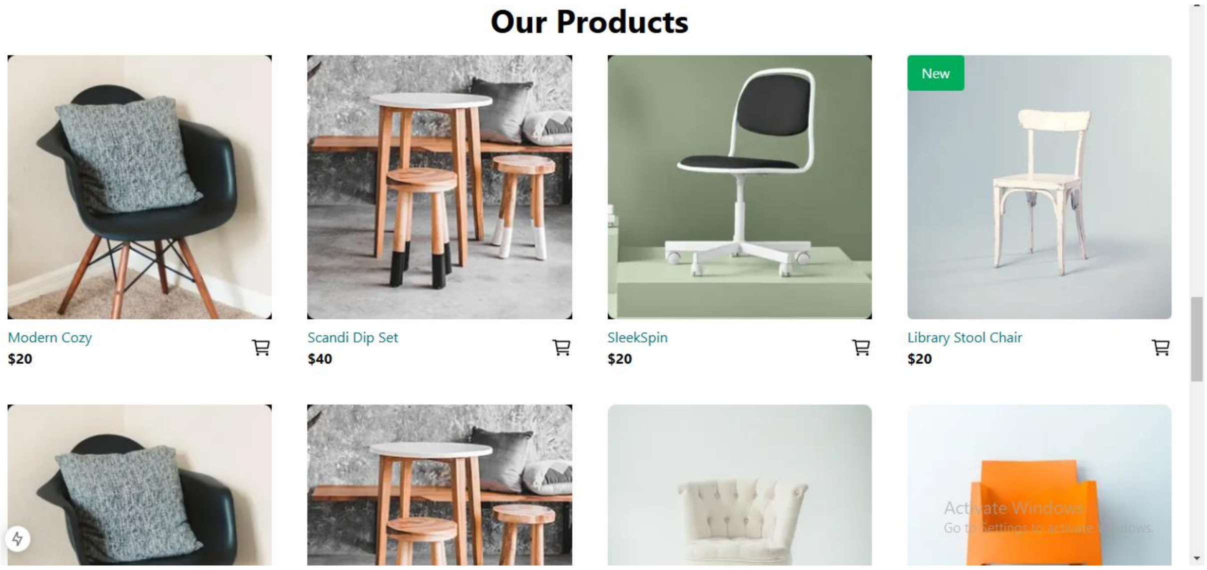
Shop Now



Featured Products

New Library Stool Chair $20	Ivory Charm $20	Sales Rose Luxe Armchair $20	Citrus Edge $20

Top Categories



Our Products

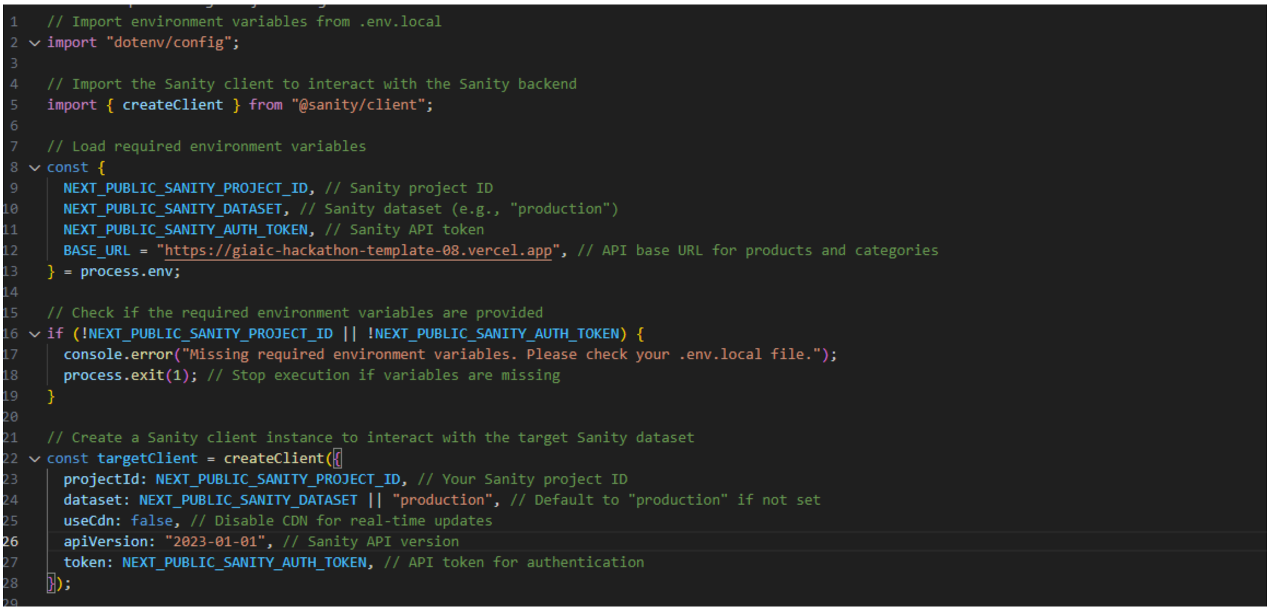
New

Modern Cozy
$20

Scandi Dip Set
$40

SleekSpin
$20

Library Stool Chair
$20



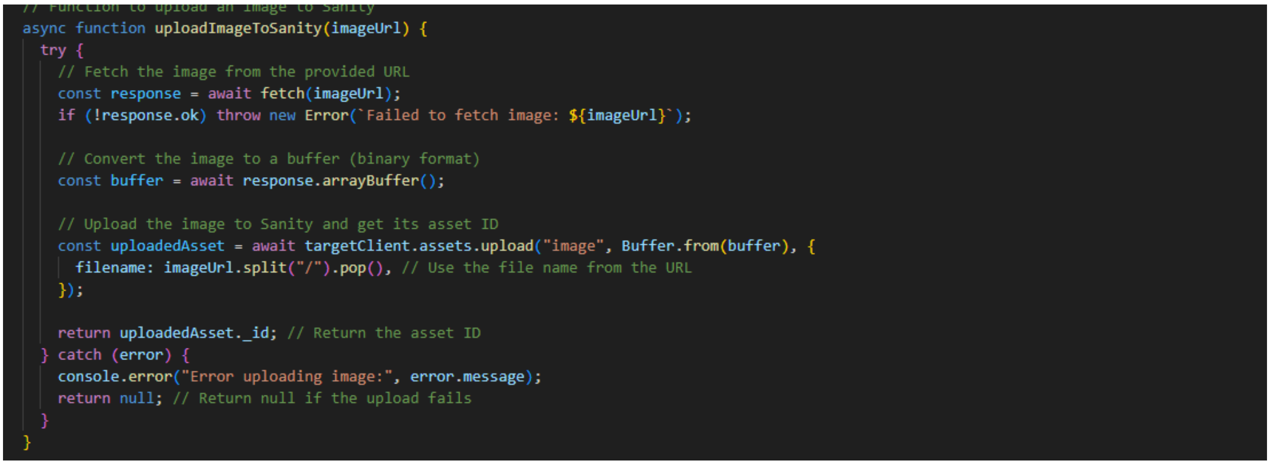
```
// Import environment variables from .env.local
import "dotenv/config";

// 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
});
```

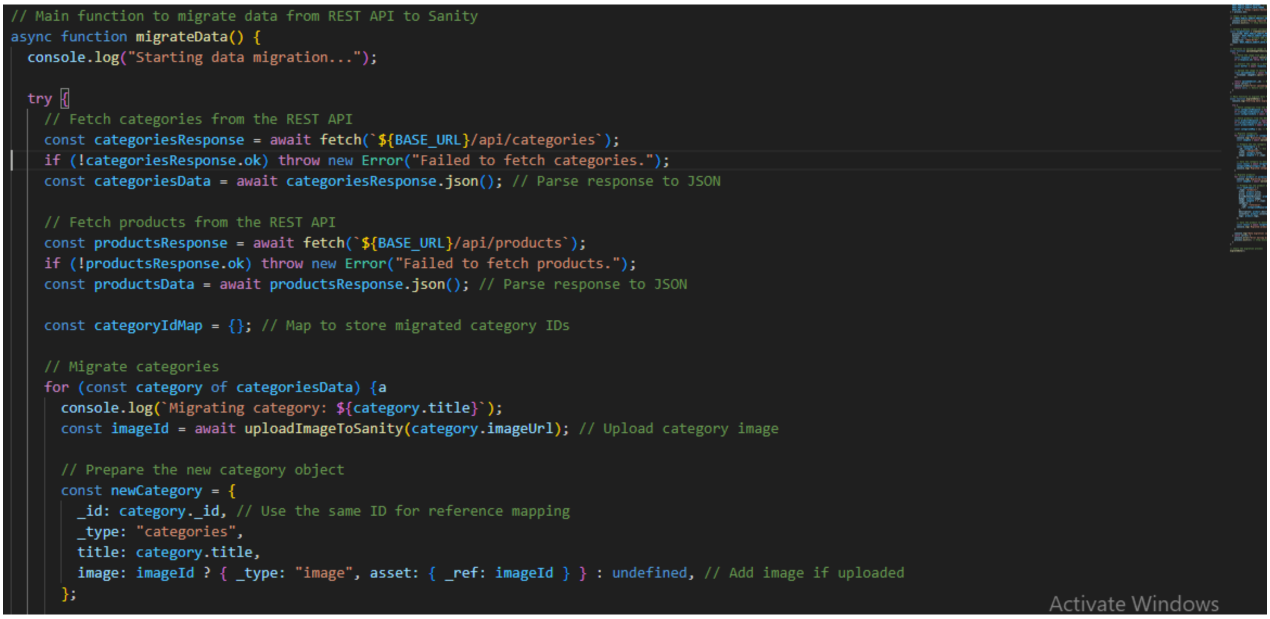


```
// Function to upload an image to Sanity
async function uploadImageToSanity(imageUrl) {
  try {
    // Fetch the image from the provided URL
    const response = await fetch(imageUrl);
    if (!response.ok) throw new Error(`Failed to fetch image: ${imageUrl}`);

    // 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) {a
      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
      };
```