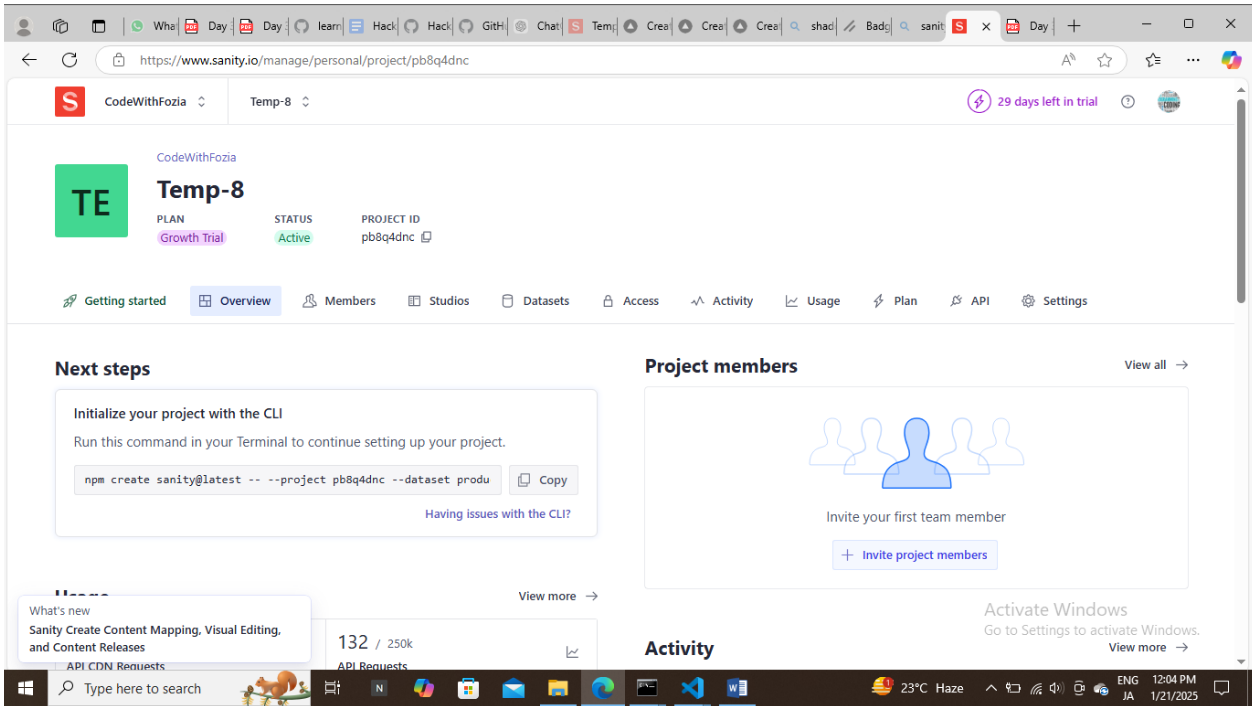
DAY 3 – API INTEGRATION REPORT – [CHAIRY]

Project Created On Sanity



https://www.sanity.io/manage/personal/project/pb8q4dnc

CodeWithFozia · Temp-8 · 29 days left in trial

CodeWithFozia
Temp-8

PLAN	STATUS	PROJECT ID
Growth Trial	Active	pb8q4dnc

Getting started · Overview · Members · Studios · Datasets · Access · Activity · Usage · Plan · API · Settings

Next steps

Initialize your project with the CLI

Run this command in your Terminal to continue setting up your project.

```
npm create sanity@latest -- --project pb8q4dnc --dataset produ
```

Having issues with the CLI?

Project members

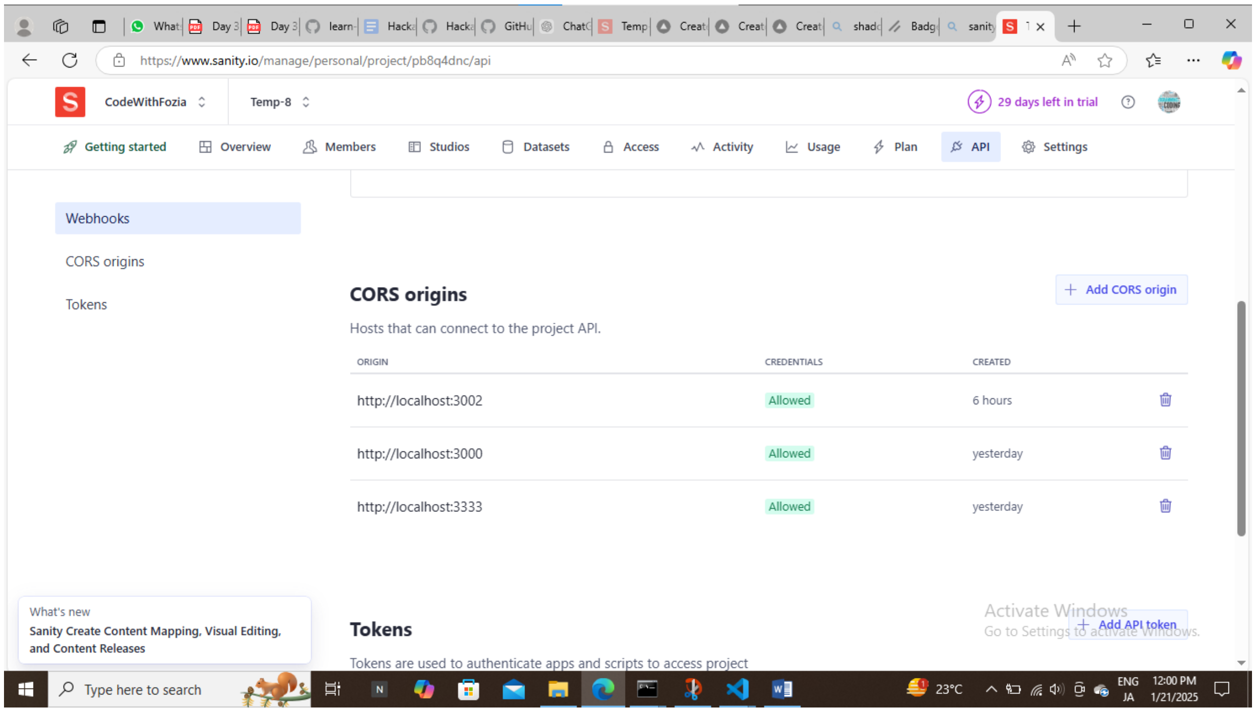
Invite your first team member

Invite project members

132 / 250k
API CDN Requests

What's new
Sanity Create Content Mapping, Visual Editing, and Content Releases

Activity



https://www.sanity.io/manage/personal/project/pb8q4dnc/api

Webhooks · CORS origins · Tokens

CORS origins

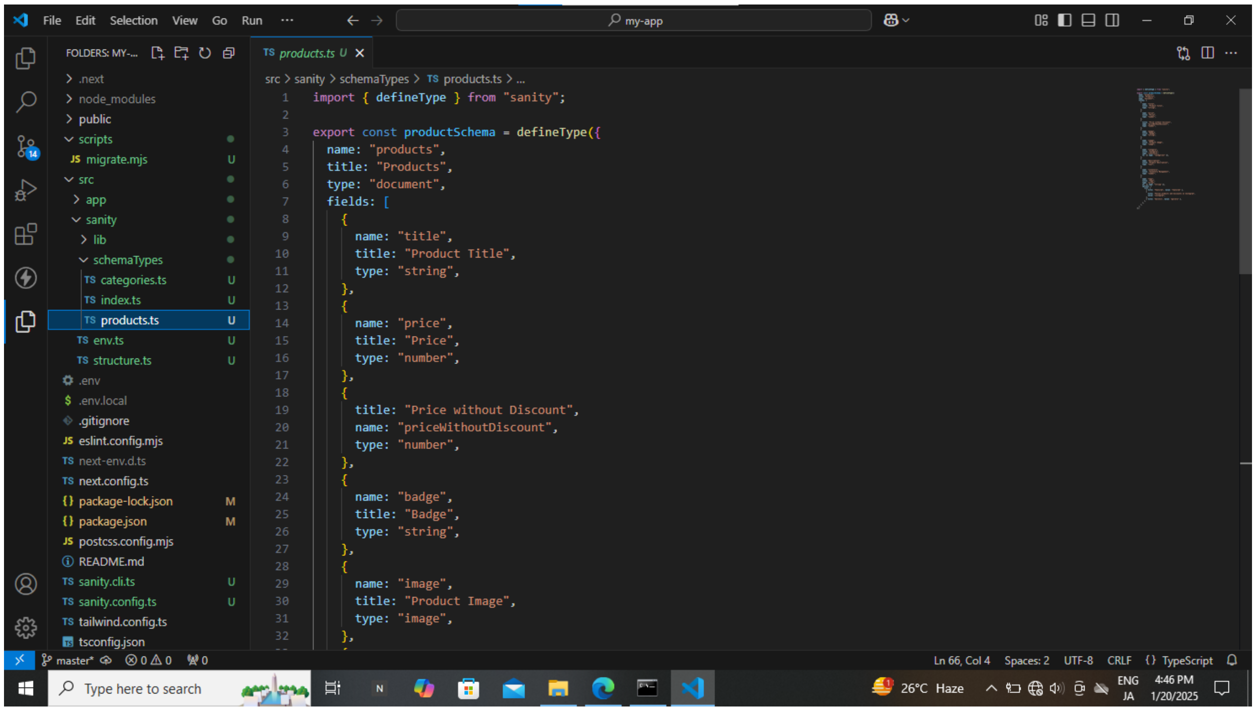
Hosts that can connect to the project API.

ORIGIN	CREDENTIALS	CREATED
http://localhost:3002	Allowed	6 hours
http://localhost:3000	Allowed	yesterday
http://localhost:3333	Allowed	yesterday

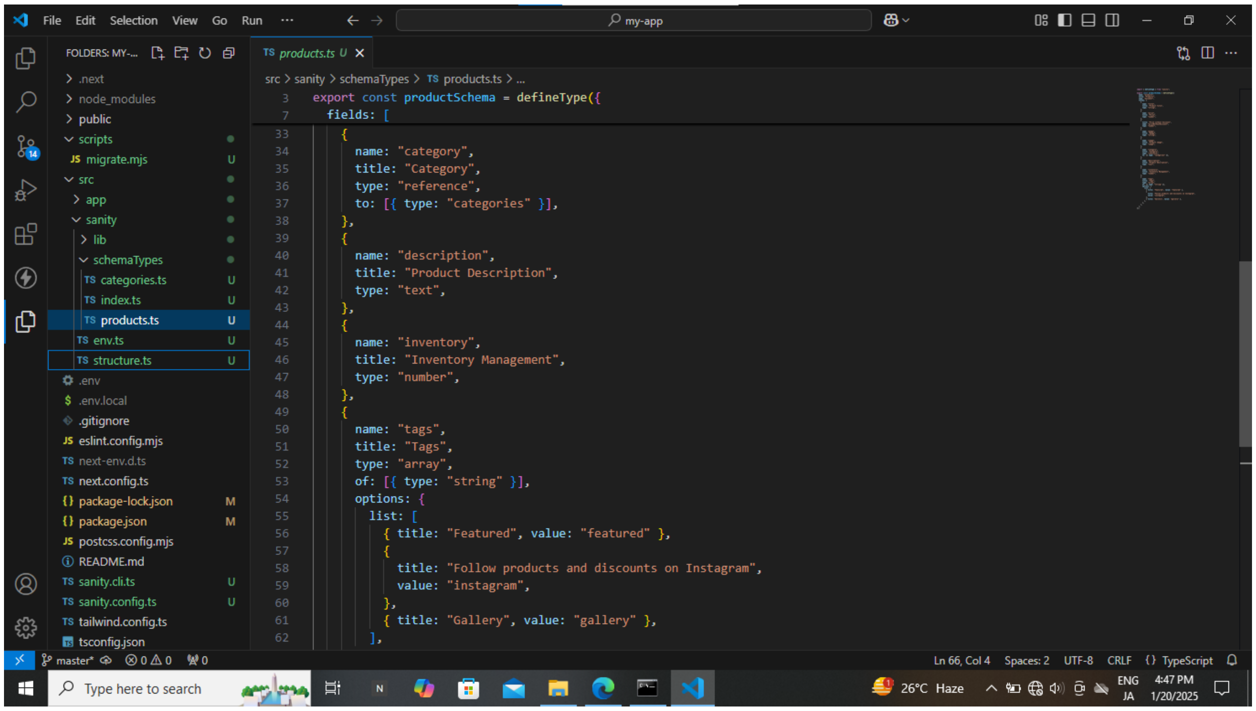
Tokens

Tokens are used to authenticate apps and scripts to access project

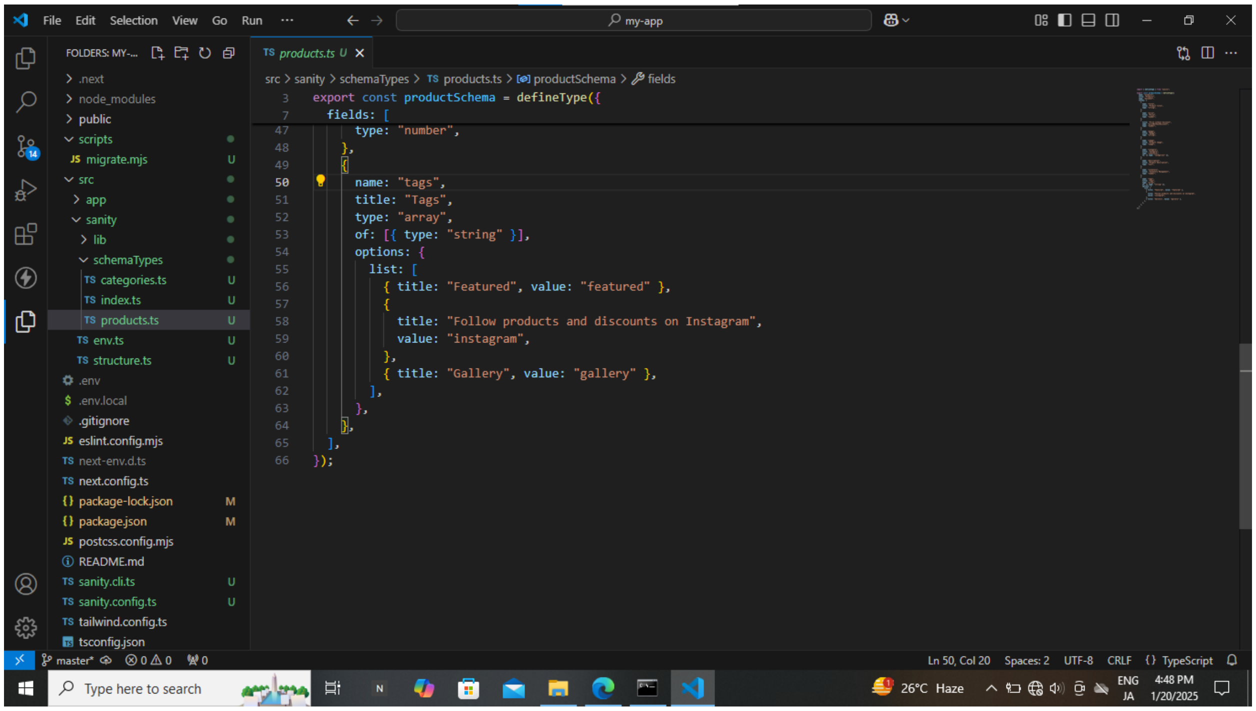
Product.ts File



```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",
    },
```

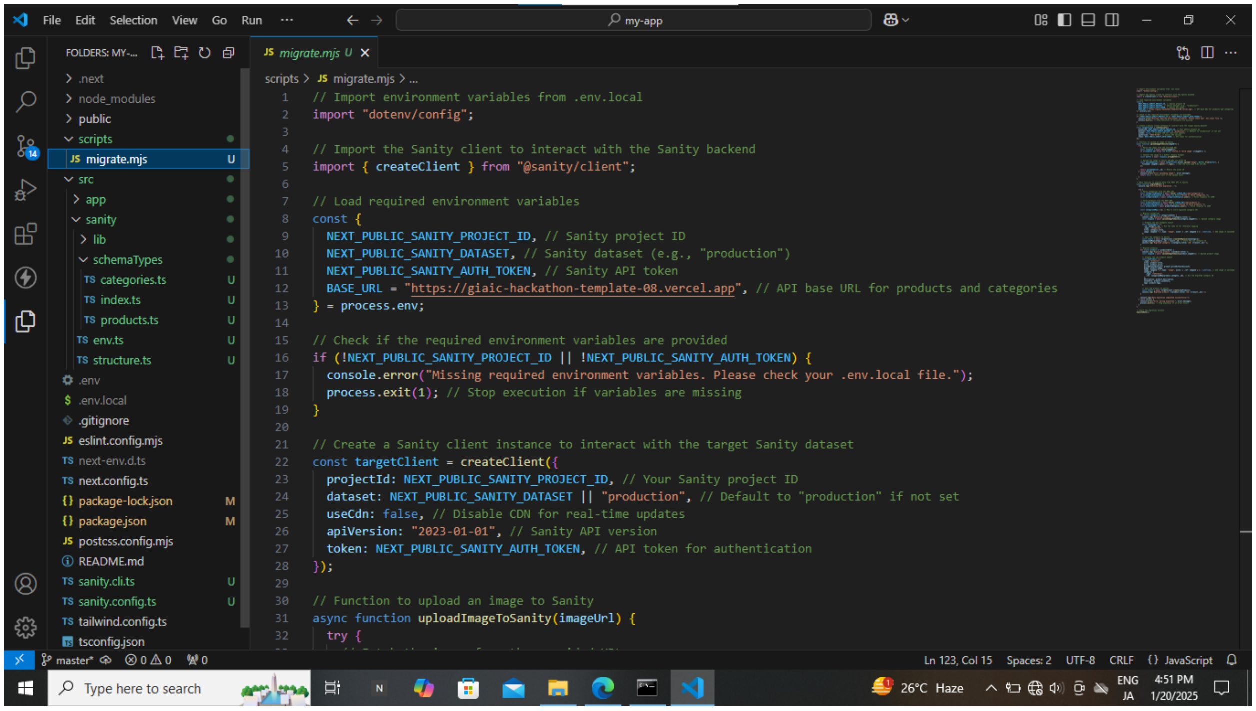


```ts
export const productSchema = defineType({
  fields: [
    {
      name: "category",
      title: "Category",
      type: "reference",
      to: [{ type: "categories" }],
    },
    {
      name: "description",
      title: "Product Description",
      type: "text",
    },
    {
      name: "inventory",
      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" },
        ],
```

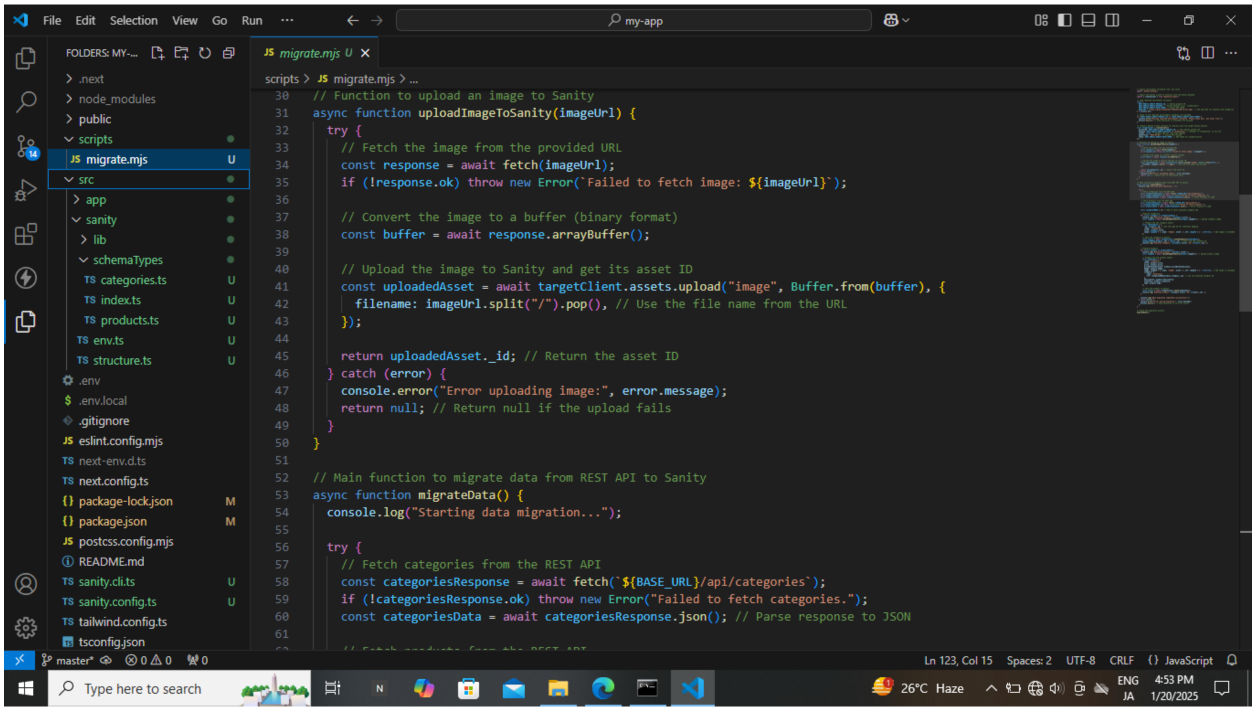


```
export const productSchema = defineType({
  fields: [
        type: "number",
      },
      {
        name: "tags",
        title: "Tags",
        type: "array",
        of: [{ type: "string" }],
        options: {
          list: [
            { title: "Featured", value: "featured" },
            {
              title: "Follow products and discounts on Instagram",
              value: "instagram",
            },
            { title: "Gallery", value: "gallery" },
          ],
        },
      },
  ],
});
```

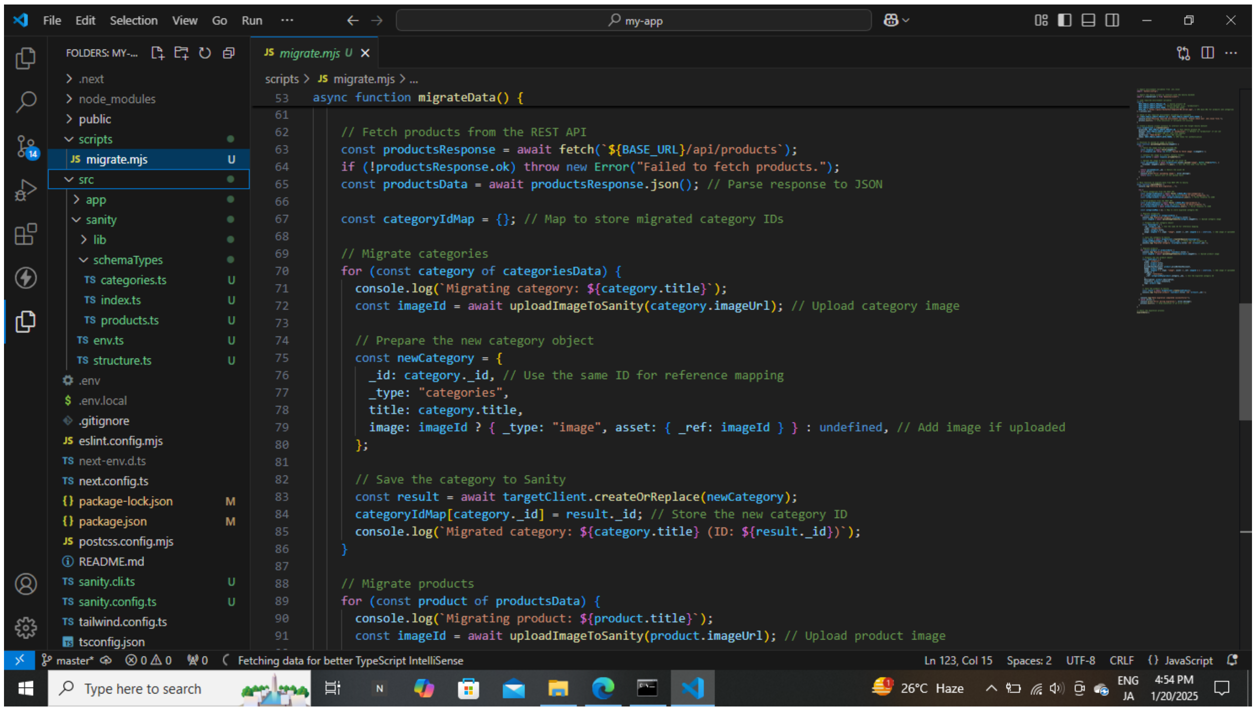
Migrate.mjs File



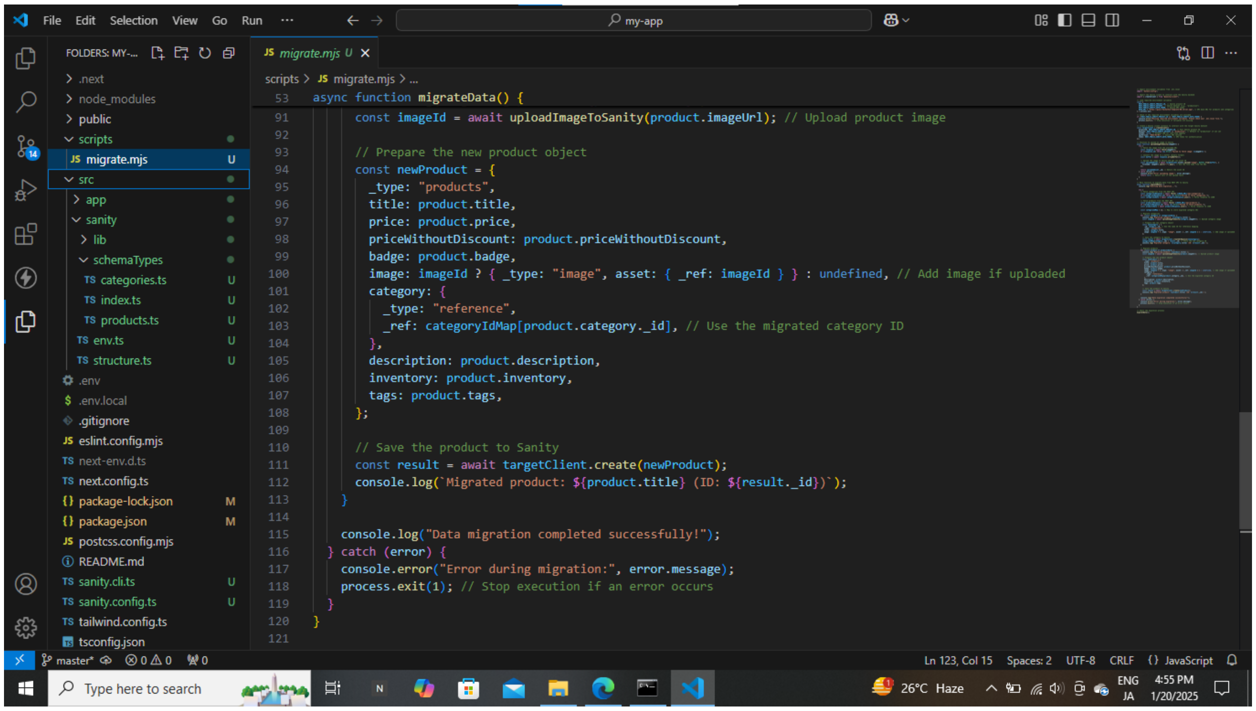
```
// Import environment variables from .env.local
import "dotenv/config";

// 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 {
```



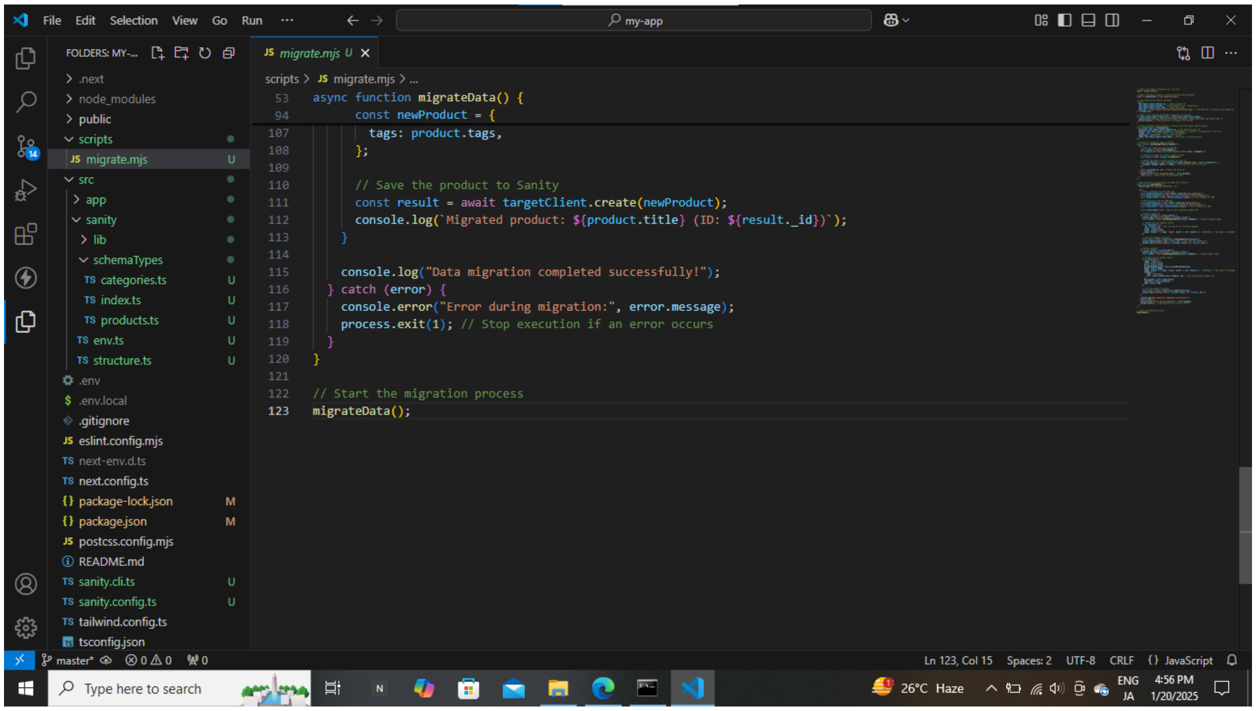
```
// Function to upload an image to Sanity
async function uploadImageToSanity(imageUrl) {
  try {
    // Fetch the image from the provided URL
    const response = await fetch(imageUrl);
    if (!response.ok) throw new Error(`Failed to fetch image: ${imageUrl}`);

    // 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

```



```
async function migrateData() {

    // 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
```

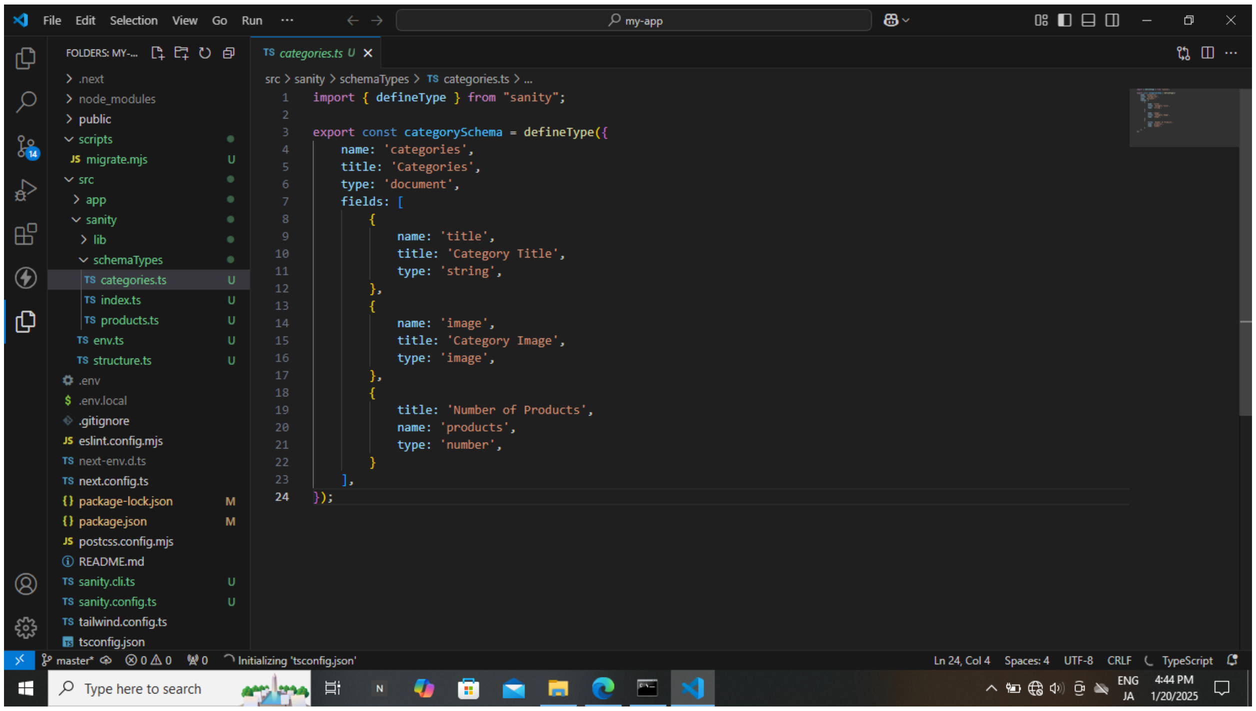


```
async function migrateData() {
      const imageId = await uploadImageToSanity(product.imageUrl); // Upload product image

      // 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
  }
}
```



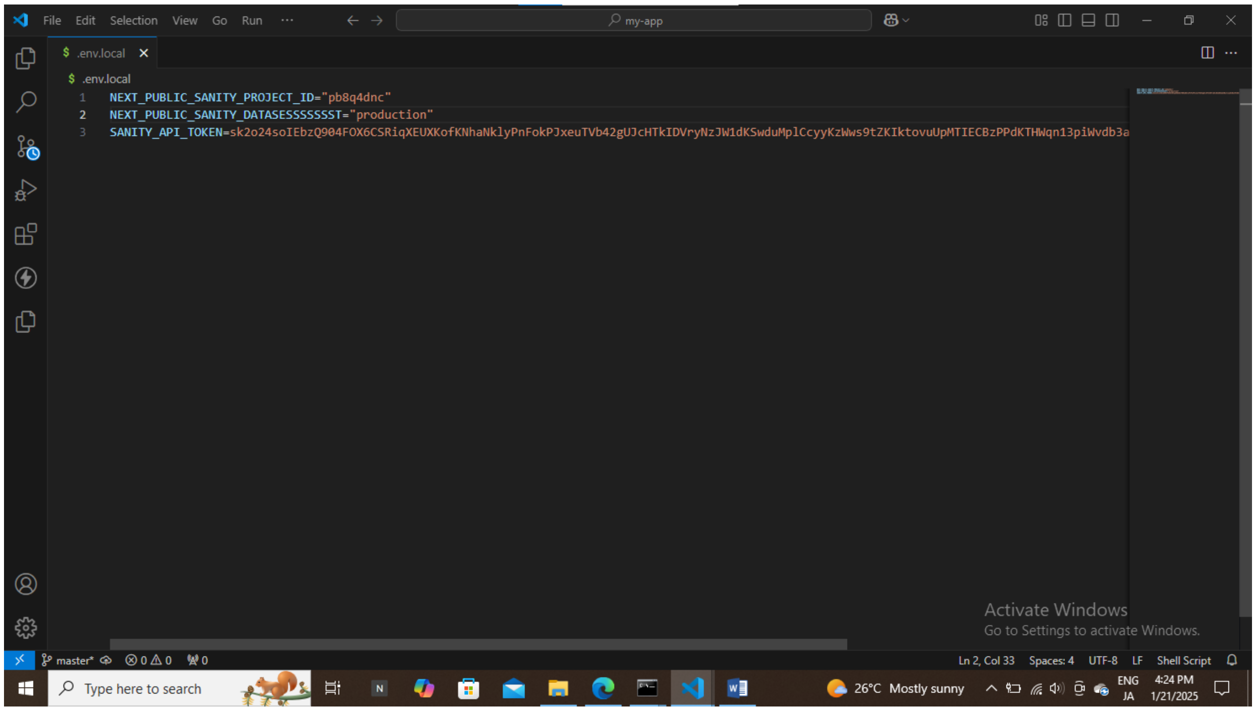
```
async function migrateData() {
      const newProduct = {
        tags: product.tags,
      };

      // 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();
```

category.ts File



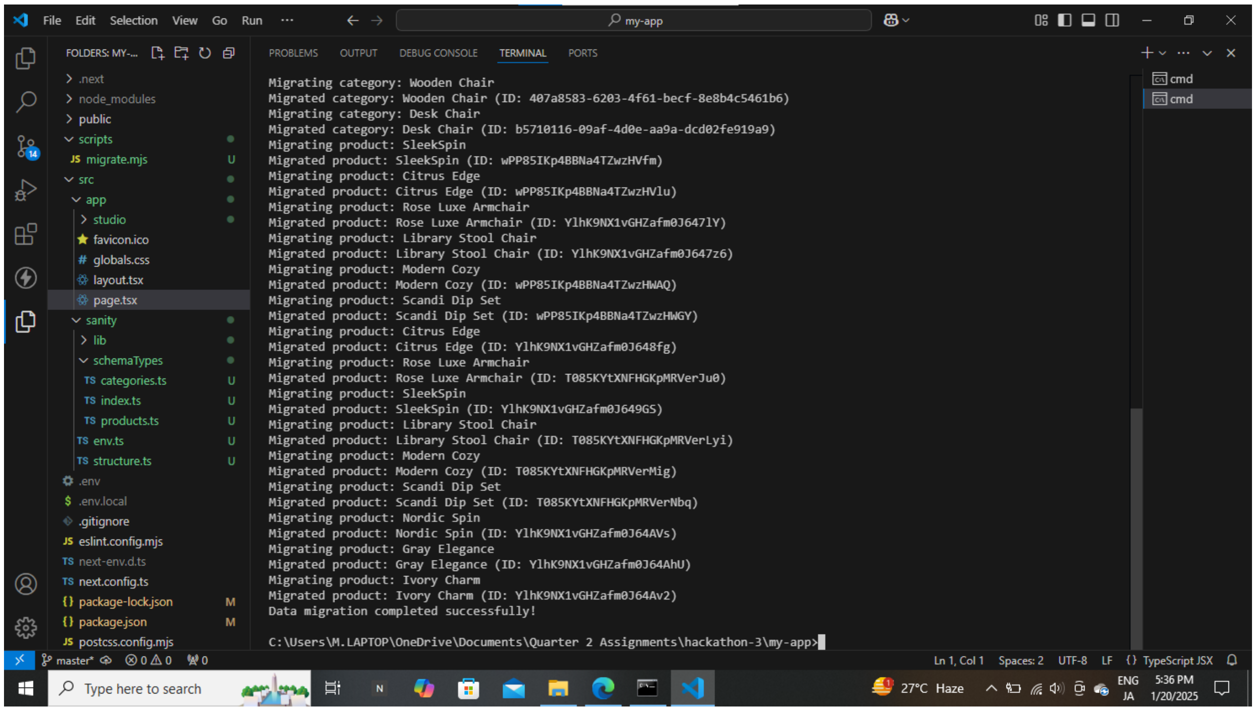
```ts
import { defineType } from "sanity";

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',
        }
    ],
});
```

.env File



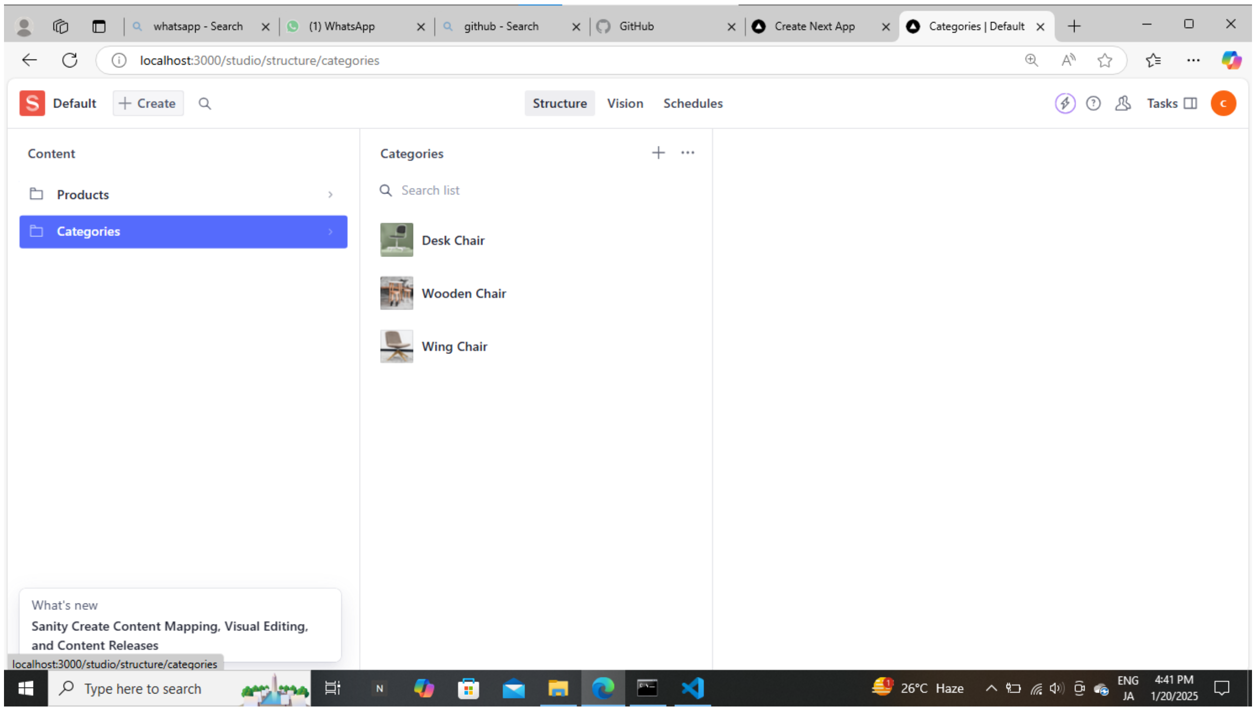
```
NEXT_PUBLIC_SANITY_PROJECT_ID="pb8q4dnc"
NEXT_PUBLIC_SANITY_DATASESSSSSSST="production"
SANITY_API_TOKEN=sk2o24soIEbzQ904FOX6CSRiqXEUXKofKNhaNklyPnFokPJxeuTVb42gUJcHTkIDVryNzJW1dKSwduMplCcyyKzWws9tZKIktovuUpMTIECBzPPdKTHWqn13piWvdb3a
```

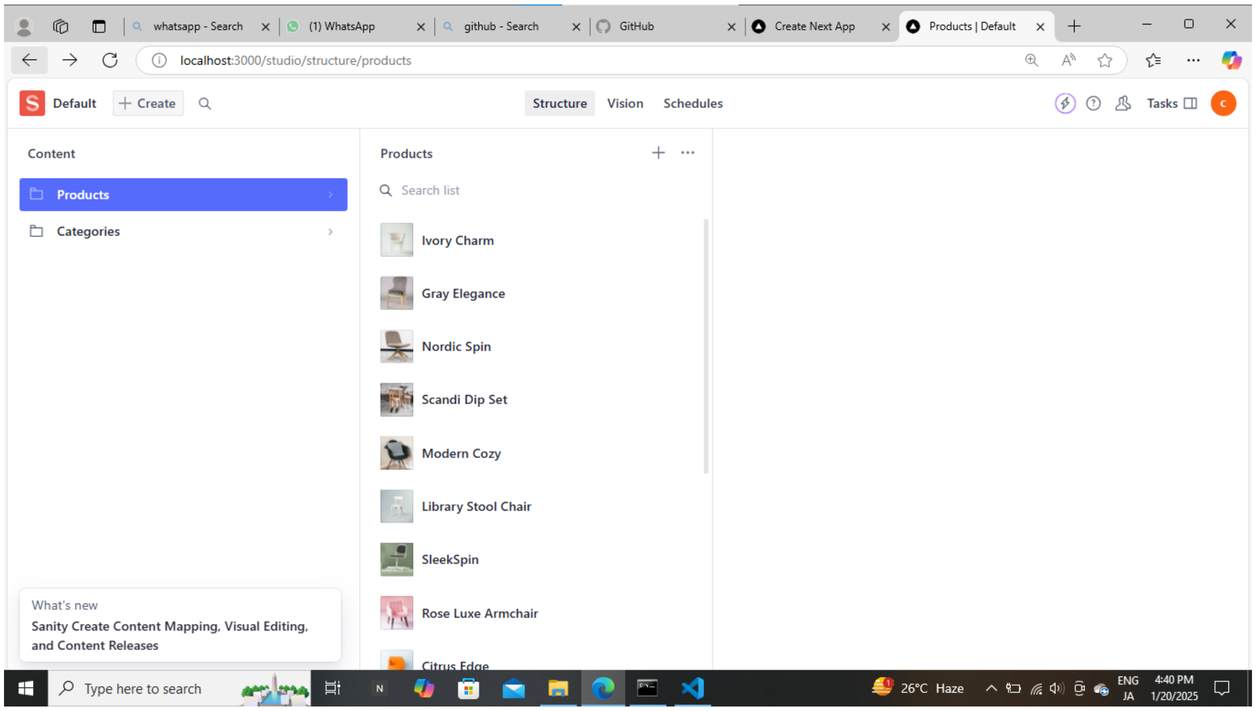
Data Migration



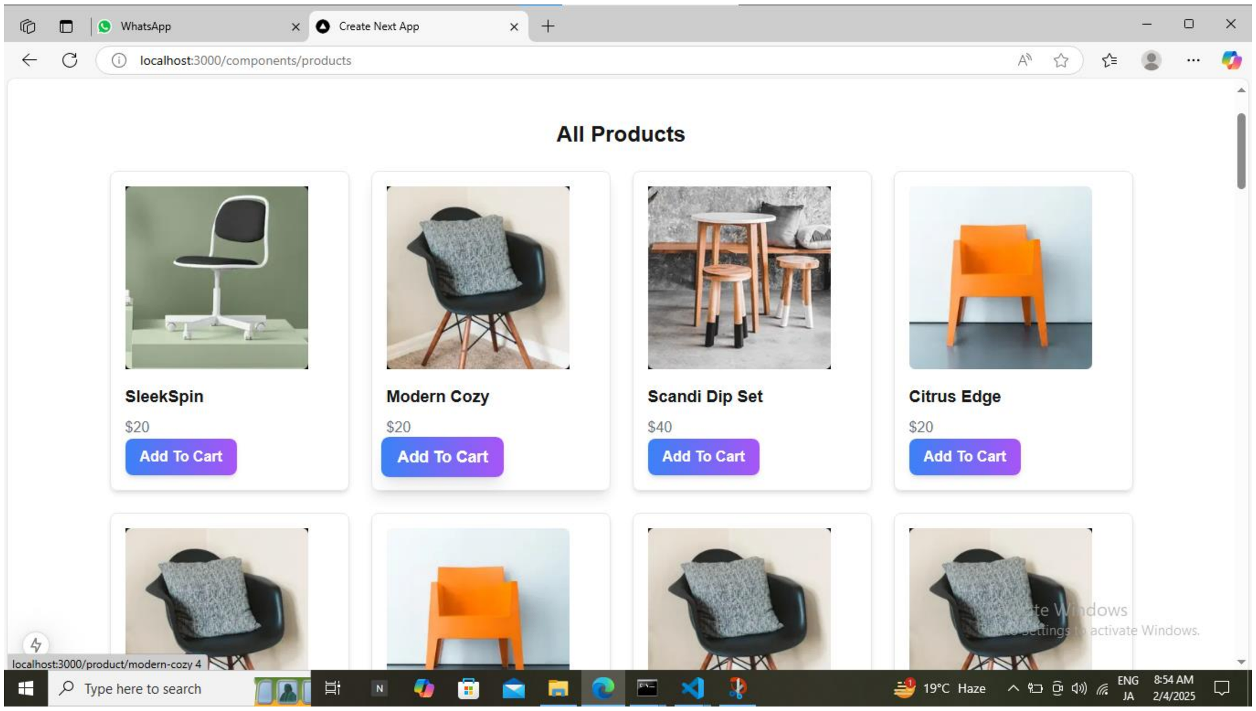
```
Migrating category: Wooden Chair
Migrated category: Wooden Chair (ID: 407a8583-6203-4f61-becf-8e8b4c5461b6)
Migrating category: Desk Chair
Migrated category: Desk Chair (ID: b5710116-09af-4d0e-aa9a-dcd02fe919a9)
Migrating product: SleekSpin
Migrated product: SleekSpin (ID: wPP85IKp4BBNa4TZwzHVfm)
Migrating product: Citrus Edge
Migrated product: Citrus Edge (ID: wPP85IKp4BBNa4TZwzHVlu)
Migrating product: Rose Luxe Armchair
Migrated product: Rose Luxe Armchair (ID: YlhK9NX1vGHZafm0J647lY)
Migrating product: Library Stool Chair
Migrated product: Library Stool Chair (ID: YlhK9NX1vGHZafm0J647z6)
Migrating product: Modern Cozy
Migrated product: Modern Cozy (ID: wPP85IKp4BBNa4TZwzHWAQ)
Migrating product: Scandi Dip Set
Migrated product: Scandi Dip Set (ID: wPP85IKp4BBNa4TZwzHWGY)
Migrating product: Citrus Edge
Migrated product: Citrus Edge (ID: YlhK9NX1vGHZafm0J648fg)
Migrating product: Rose Luxe Armchair
Migrated product: Rose Luxe Armchair (ID: T085KYtXNFHGKpMRVerJu0)
Migrating product: SleekSpin
Migrated product: SleekSpin (ID: YlhK9NX1vGHZafm0J649GS)
Migrating product: Library Stool Chair
Migrated product: Library Stool Chair (ID: T085KYtXNFHGKpMRVerLyi)
Migrating product: Modern Cozy
Migrated product: Modern Cozy (ID: T085KYtXNFHGKpMRVerMig)
Migrating product: Scandi Dip Set
Migrated product: Scandi Dip Set (ID: T085KYtXNFHGKpMRVerNbq)
Migrating product: Nordic Spin
Migrated product: Nordic Spin (ID: YlhK9NX1vGHZafm0J64AVs)
Migrating product: Gray Elegance
Migrated product: Gray Elegance (ID: YlhK9NX1vGHZafm0J64AhU)
Migrating product: Ivory Charm
Migrated product: Ivory Charm (ID: YlhK9NX1vGHZafm0J64Av2)
Data migration completed successfully!

C:\Users\M.LAPTOP\OneDrive\Documents\Quarter 2 Assignments\hackathon-3\my-app>
```

Data Import In Sanity





Data fetch to Frontend



Made by Fouzia Bibi!