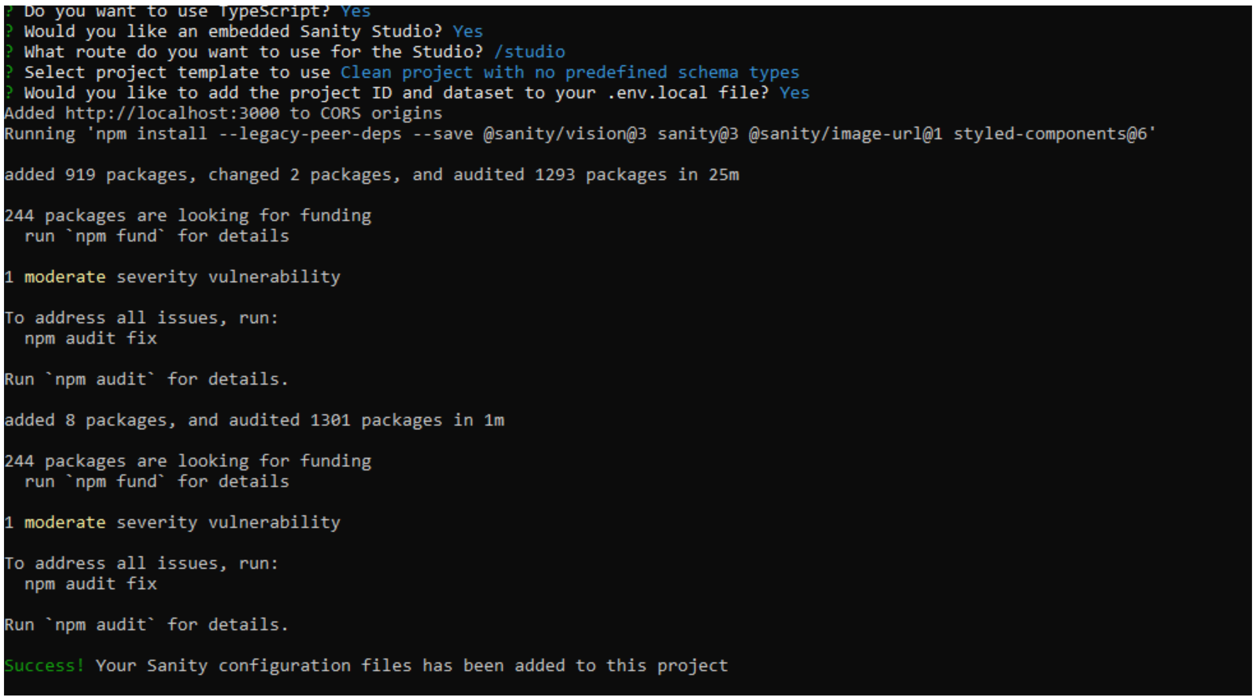
Hackathon 3

Template 8 Documentation

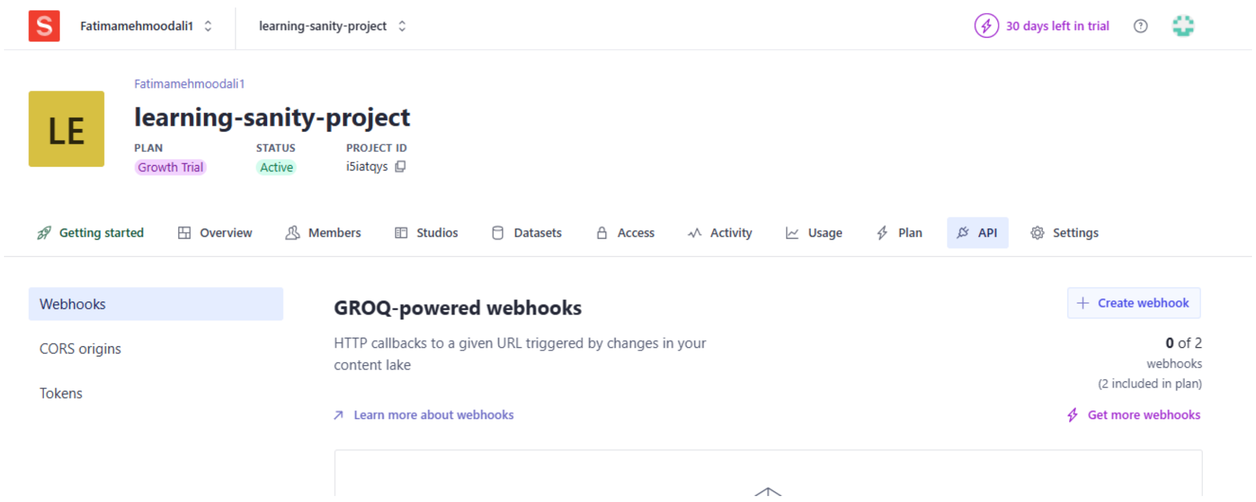
Introduction:

Welcome to the hackathon! Template number **8** in Figma. We are building **Comforty**, a chair marketplace. Previously, you've focused on developing the frontend. In this hackathon, we aim to extend functionality by integrating **Sanity** to manage product data efficiently.

Sanity Inatallation:



```
? 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'

added 919 packages, changed 2 packages, and audited 1293 packages in 25m

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

1 moderate severity vulnerability

To address all issues, run:
  npm audit fix

Run `npm audit` for details.

added 8 packages, and audited 1301 packages in 1m

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

1 moderate severity vulnerability

To address all issues, run:
  npm audit fix

Run `npm audit` for details.

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


Fatimamehmoodali1 | learning-sanity-project | 30 days left in trial

Fatimamehmoodali1

learning-sanity-project

PLAN	STATUS	PROJECT ID
Growth Trial	Active	i5iatqys

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

Webhooks · CORS origins · Tokens

GROQ-powered webhooks

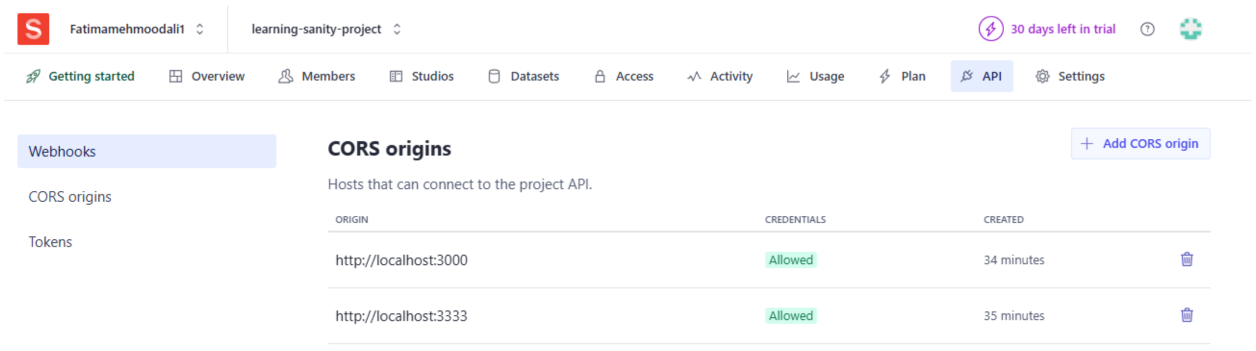
HTTP callbacks to a given URL triggered by changes in your content lake

Learn more about webhooks

Create webhook

0 of 2 webhooks (2 included in plan)

Get more webhooks


Fatimamehmoodali1 | learning-sanity-project | 30 days left in trial

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

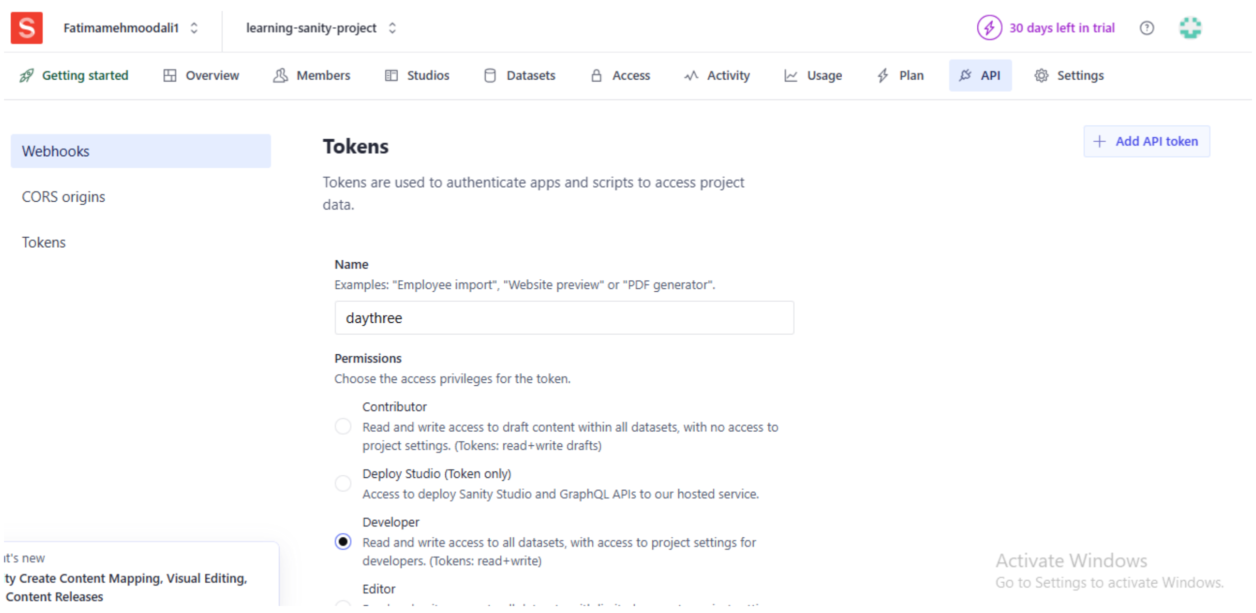
Webhooks · CORS origins · Tokens

CORS origins

Hosts that can connect to the project API.

Add CORS origin

ORIGIN	CREDENTIALS	CREATED
http://localhost:3000	Allowed	34 minutes
http://localhost:3333	Allowed	35 minutes


Fatimamehmoodali1 | learning-sanity-project | 30 days left in trial

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

Webhooks · CORS origins · Tokens

Tokens

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

Add API token

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

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)

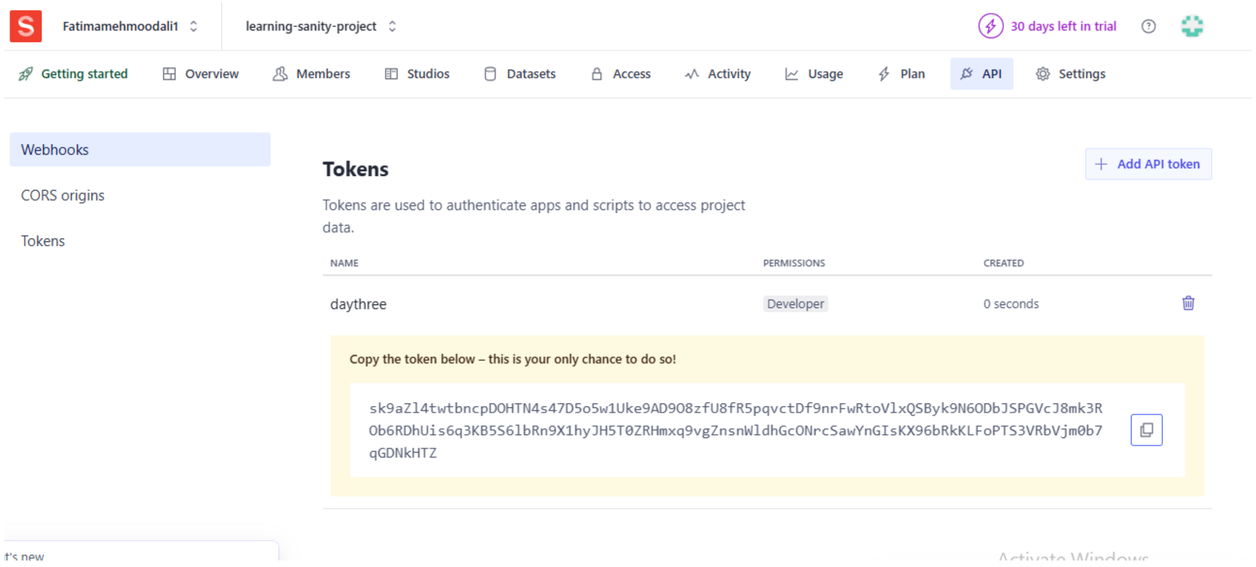
Deploy Studio (Token only)
Access to deploy Sanity Studio and GraphQL APIs to our hosted service.

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

Editor

ıt's new
ty Create Content Mapping, Visual Editing, Content Releases

Activate Windows
Go to Settings to activate Windows.



Tokens

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

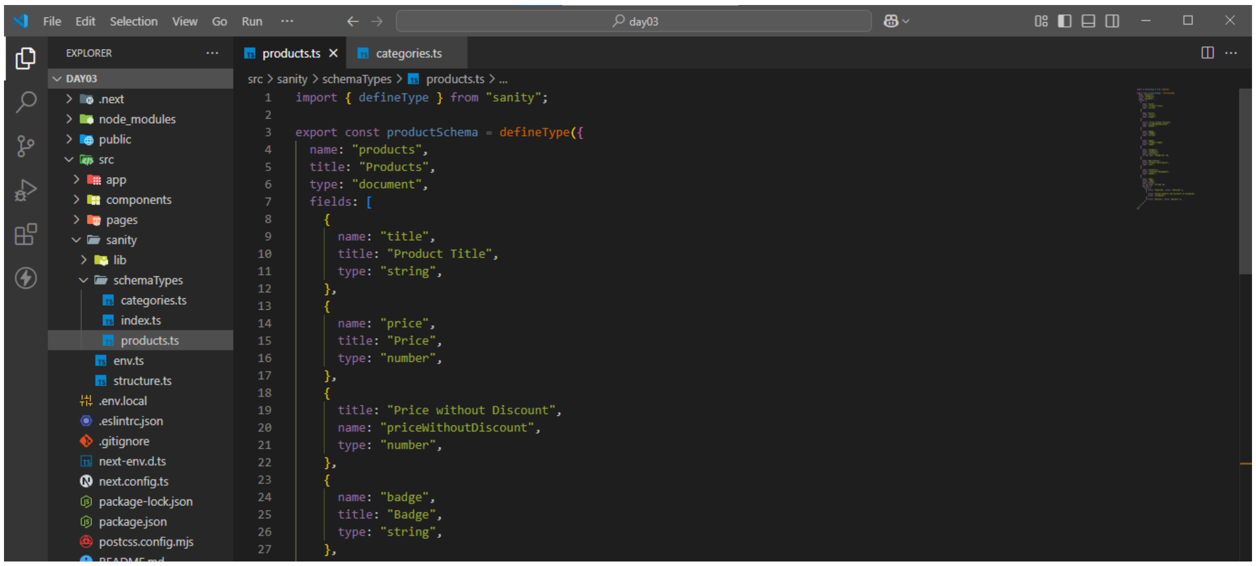
NAME	PERMISSIONS	CREATED
daythree	Developer	0 seconds

Copy the token below – this is your only chance to do so!

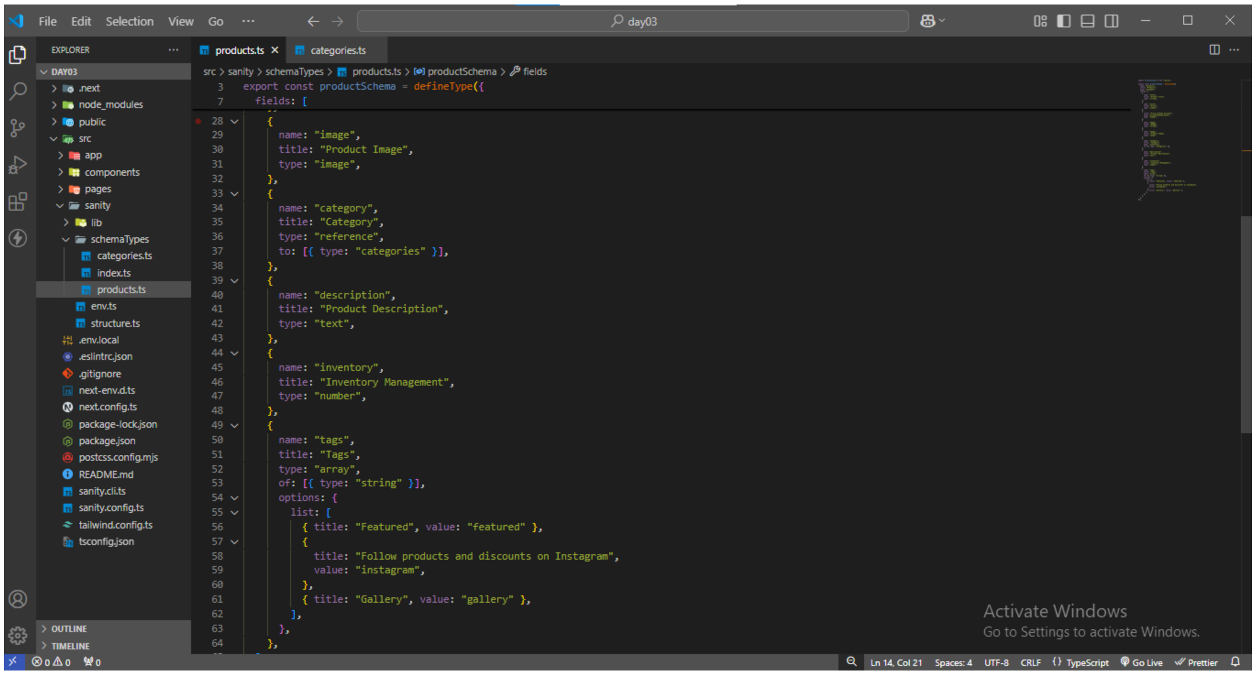
```
sk9aZl4twtbncpDOHTN4s47D5o5w1Uke9AD9O8zfU8fR5pqvctDf9nrFwRtoVlxQSByk9N6ODbJSPGVcJ8mk3R
Ob6RDhUis6q3KB5S6lbRn9X1hyJH5T0ZRHmxq9vgZnsnWldhGcONrcSawYnGIsKX96bRkKLFoPTS3VRbVjm0b7
qGDNkHTZ
```

Sanity Schema:

1. After the installation Navigate to your schema folder:
   a. If you have a `src` folder, go to `/src/sanity/schemaTypes`.
   b. Otherwise, go to `/sanity/schemaTypes`.
2. Create two new files and add the following code:
   - **Products.ts:**

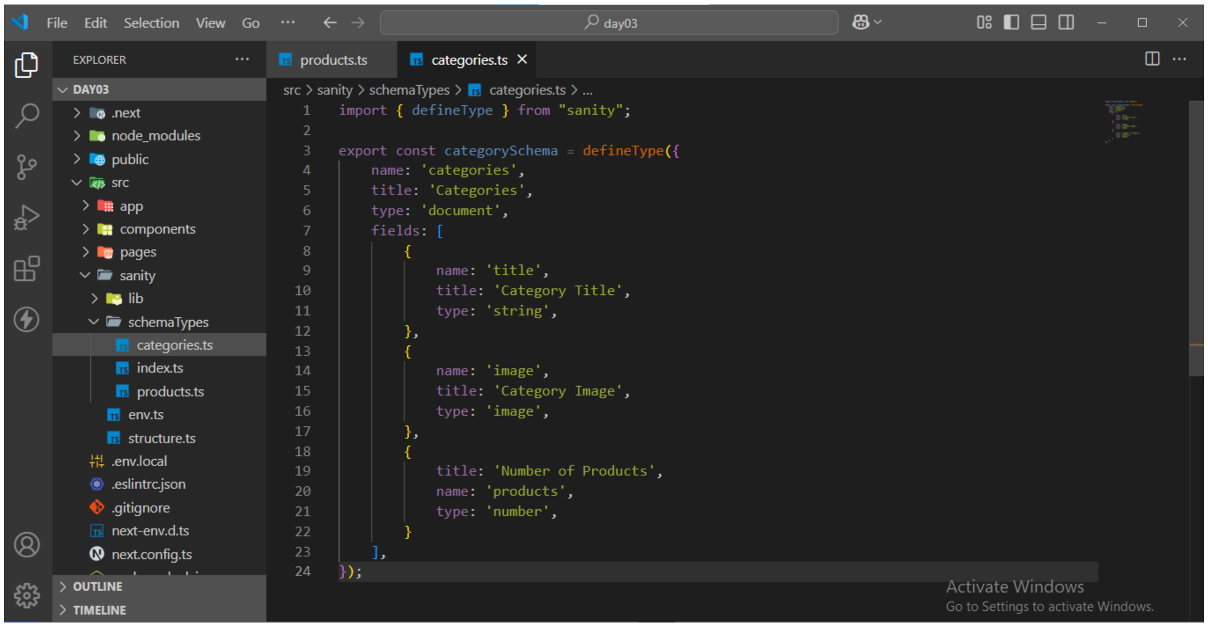


```ts
import { defineType } from "sanity";

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",
    },
```



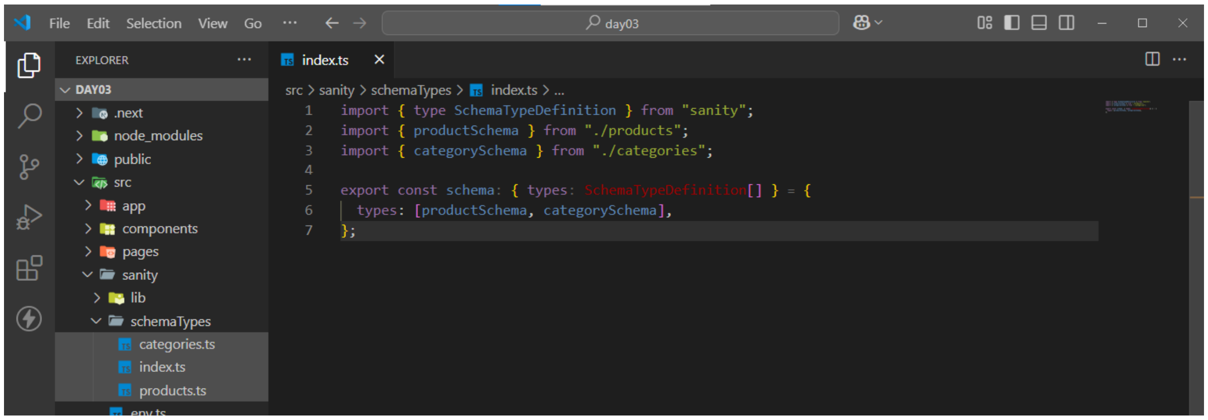
```
export const productSchema = defineType({
  fields: [
    {
      name: "image",
      title: "Product Image",
      type: "image",
    },
    {
      name: "category",
      title: "Category",
      type: "reference",
      to: [{ type: "categories" }],
    },
    {
      name: "description",
      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:**



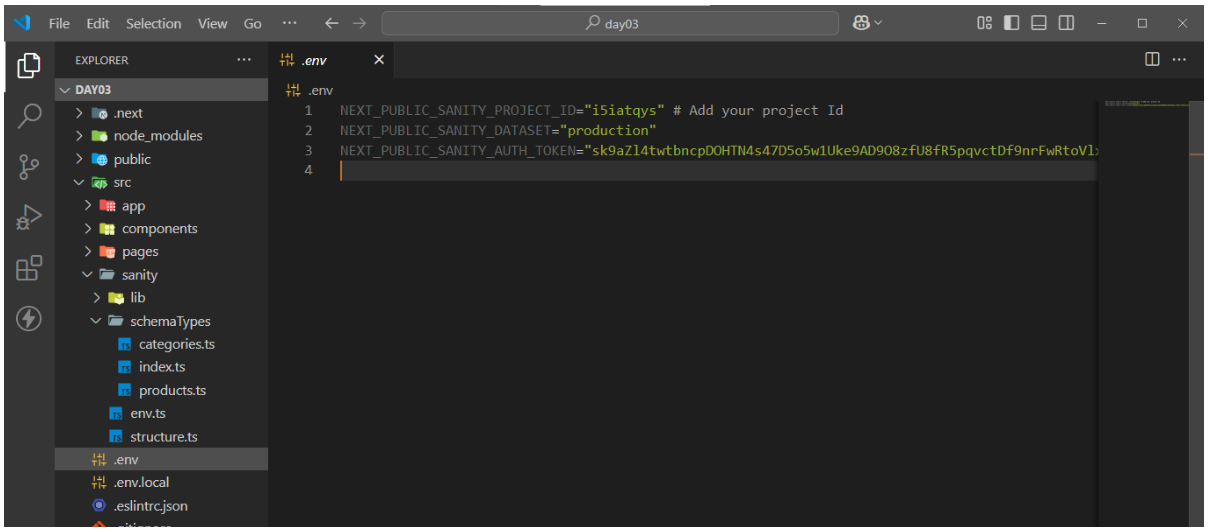
```
import { defineType } from "sanity";

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',
        }
    ],
});
```

-

- **Importing Schemas in `index.ts`:**
- 

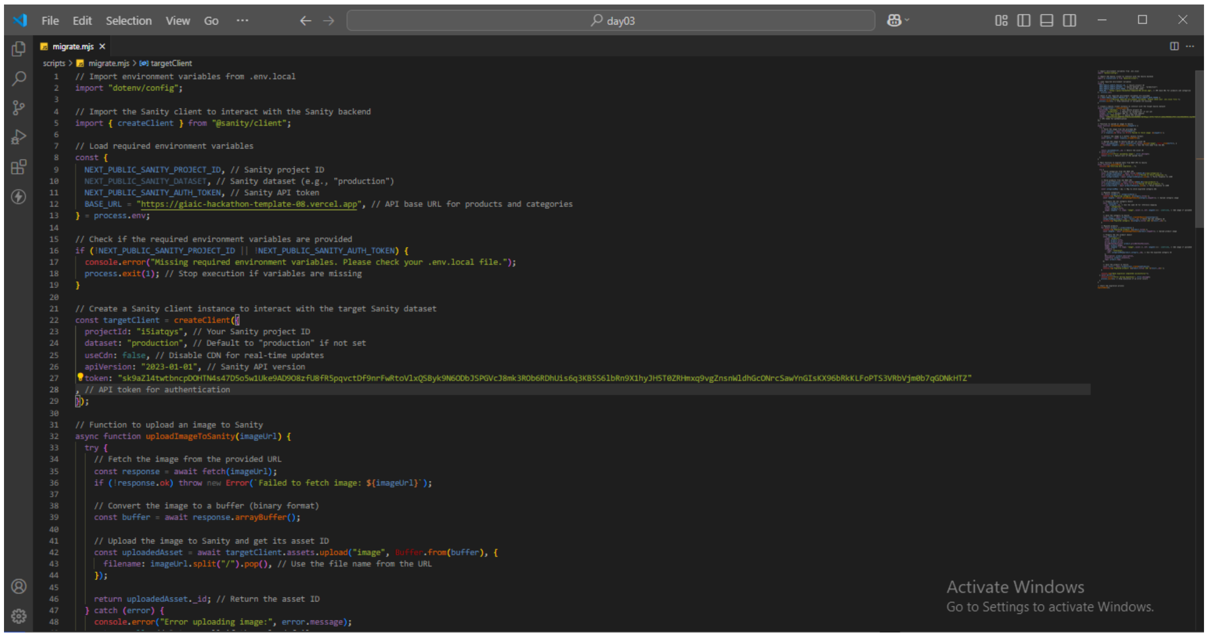
```
src > sanity > schemaTypes > index.ts > ...
import { type SchemaTypeDefinition } from "sanity";
import { productSchema } from "./products";
import { categorySchema } from "./categories";

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

- Setting Up Environment Variables

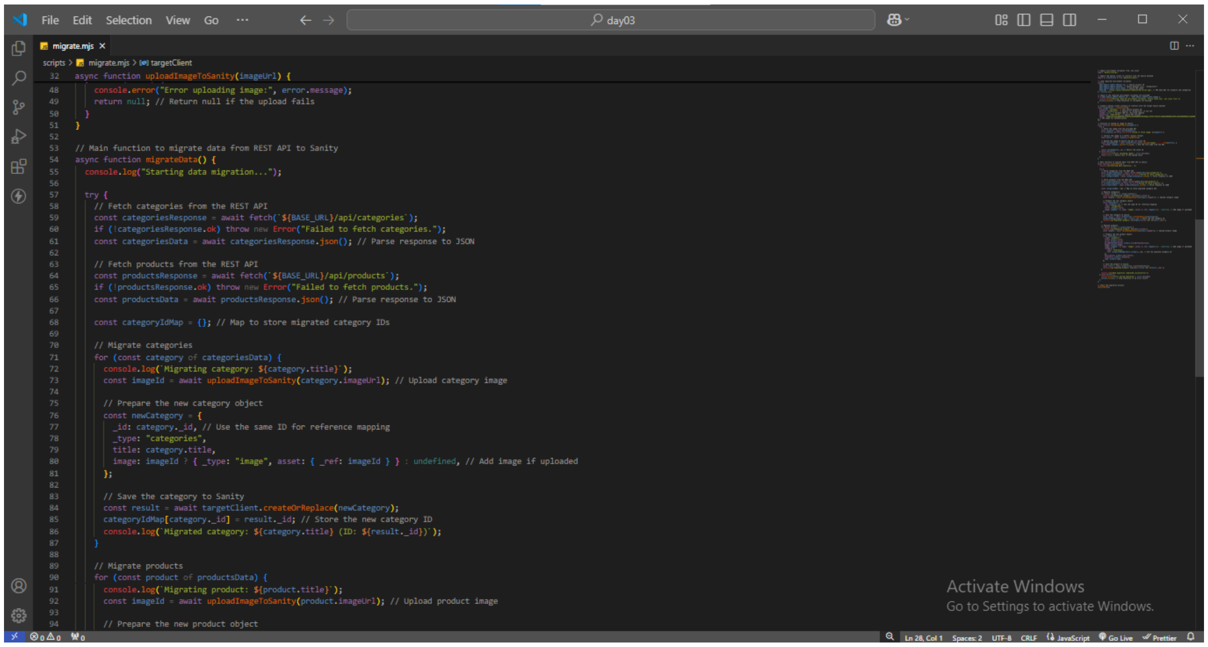
- Create a `.env` file in the root of your project. **Do not depend on .env.local file for the script, you need to create .env file to make the script work**
- Create `.env` file and add the following variables:
- 

```
.env
NEXT_PUBLIC_SANITY_PROJECT_ID="i5iatqys" # Add your project Id
NEXT_PUBLIC_SANITY_DATASET="production"
NEXT_PUBLIC_SANITY_AUTH_TOKEN="sk9aZl4twtbncpDOHTN4s47D5o5w1Uke9AD908zfU8fR5pqvctDf9nrFwRtoVl
```

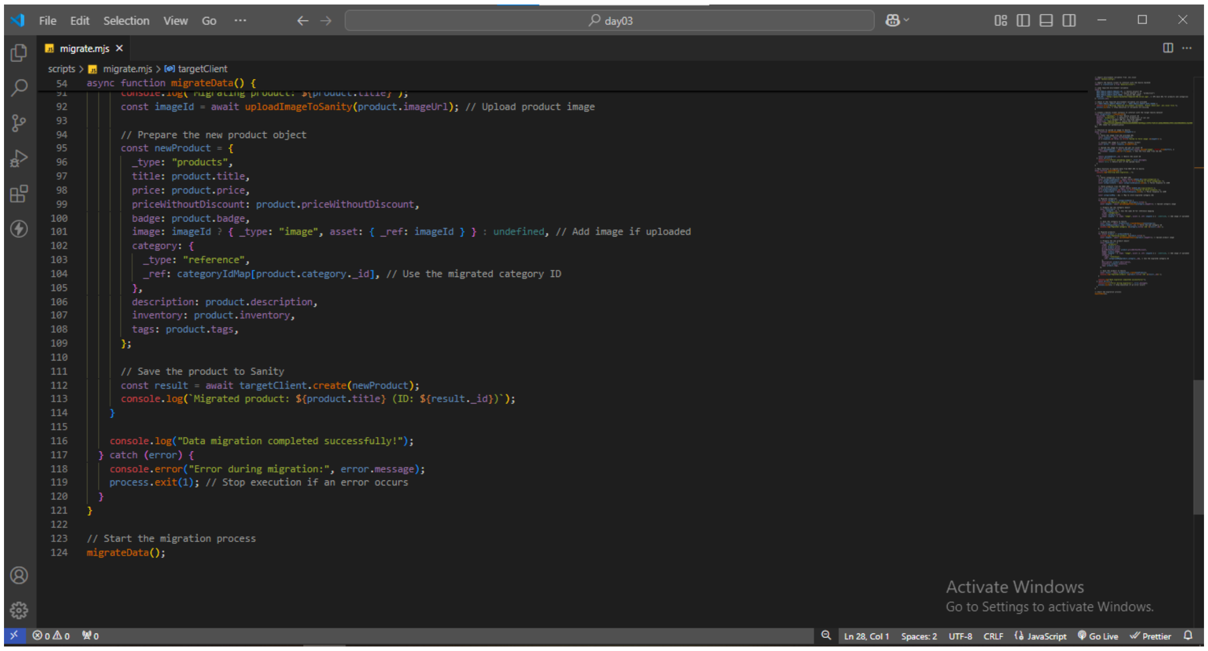
- `Create a script folder manually in the root of your project`
- Create `migrate.mjs` `inside of the script folder`



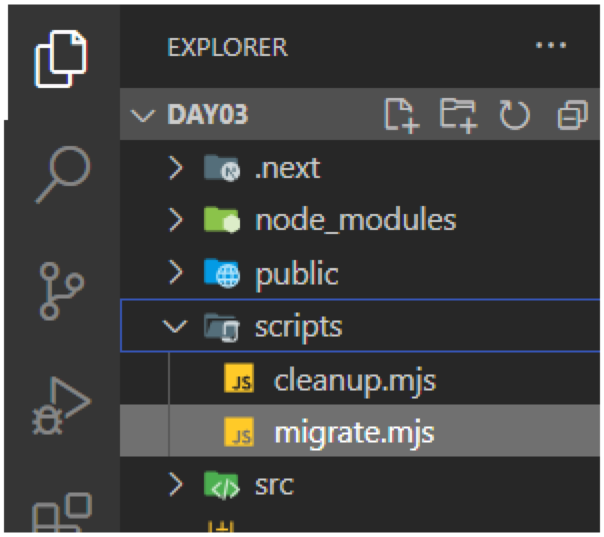
```javascript
// Import environment variables from .env.local
import "dotenv/config";

// 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: "i5iatqys", // Your Sanity project ID
  dataset: "production", // Default to "production" if not set
  useCdn: false, // Disable CDN for real-time updates
  apiVersion: "2023-01-01", // Sanity API version
  token: "sk9aZl4twtbncpDOHTN4s47D5o5w1Uke9AD9O8zfU8fR5pqvctDf9nrFwRtoVlxQSByk9N6ODbJSPGVcJ8mk3ROb6RDhUis6q3KB5S6lbRn9X1hyJH5T0ZRHmxq9vgZnsnWldhGcONrcSawYnGIsKX96bRkKLFoPTS3VRbVjm0b7qGDNkHTZ"
, // 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);
```



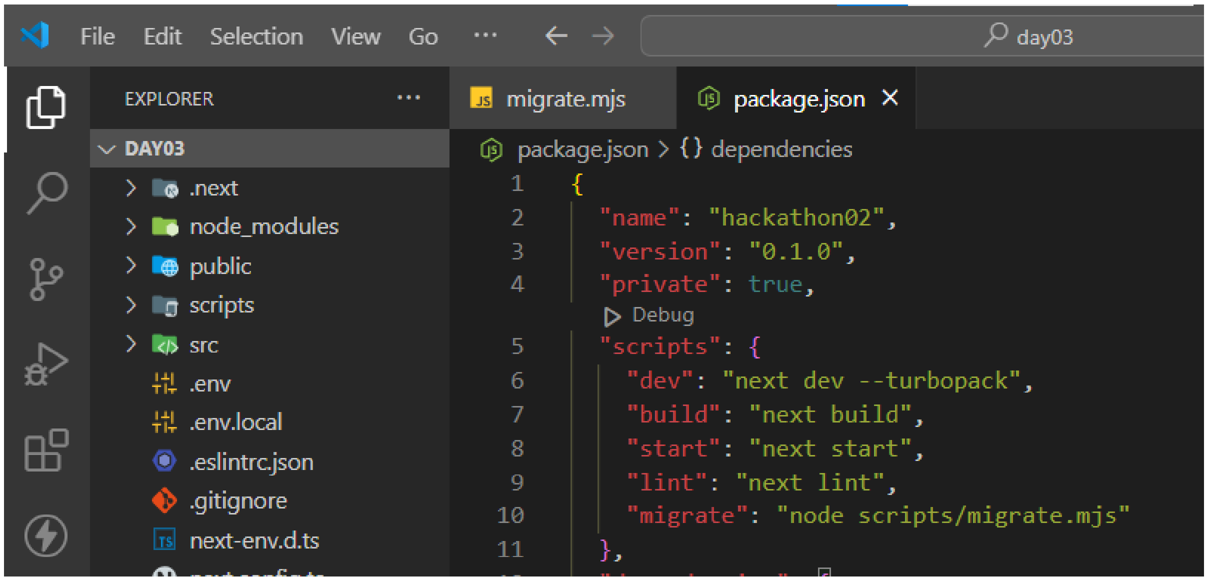
```javascript
async function uploadImageToSanity(imageUrl) {
    console.error("Error uploading image:", error.message);
    return null; // Return null if the upload fails
  }
}

// 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
```

- 

```javascript
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
  }
}

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

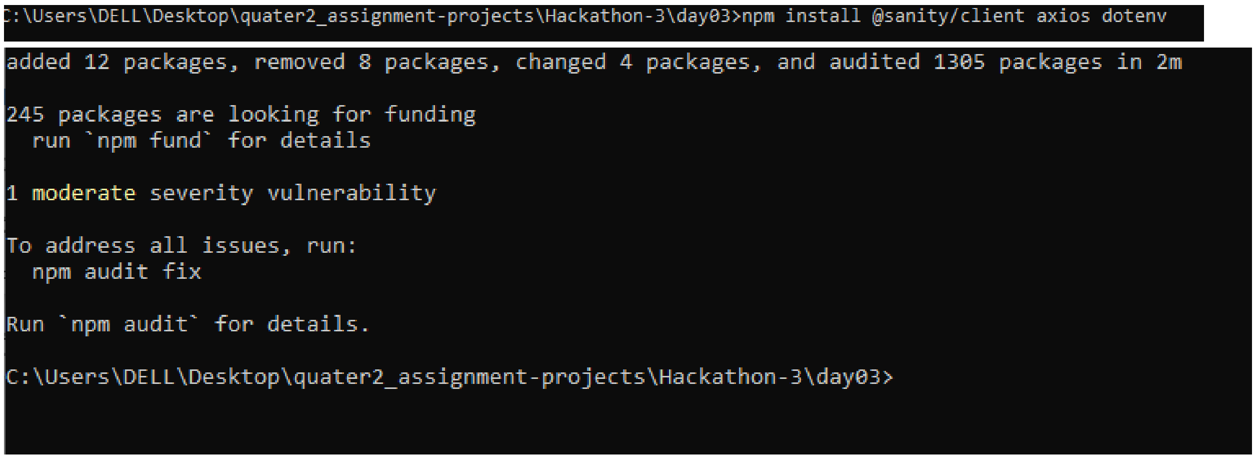
- 

- Open `package.json` file and add the following code:

- 

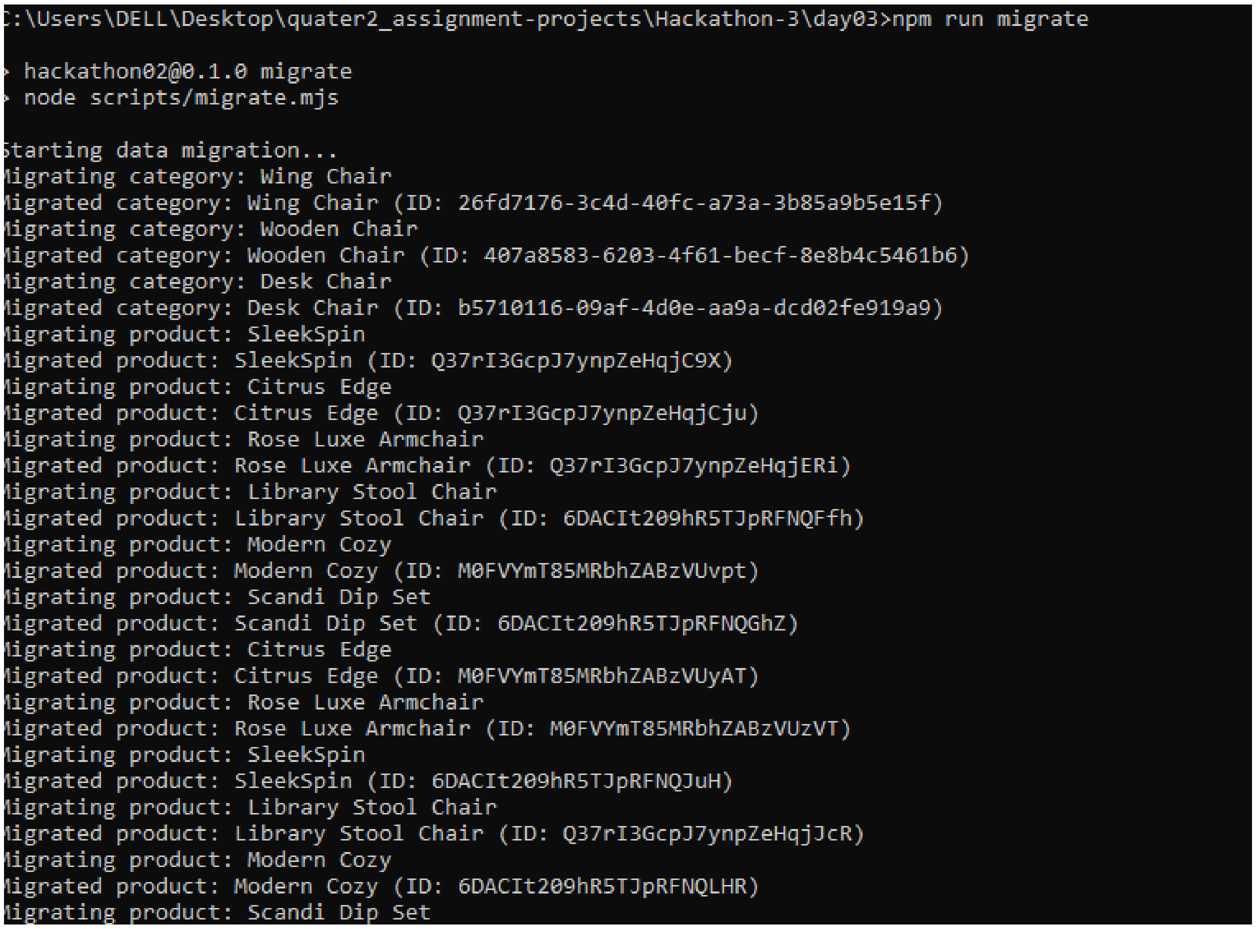
```json
{
  "name": "hackathon02",
  "version": "0.1.0",
  "private": true,
  "scripts": {
    "dev": "next dev --turbopack",
    "build": "next build",
    "start": "next start",
    "lint": "next lint",
    "migrate": "node scripts/migrate.mjs"
  },
```

- Install the following package before running the script

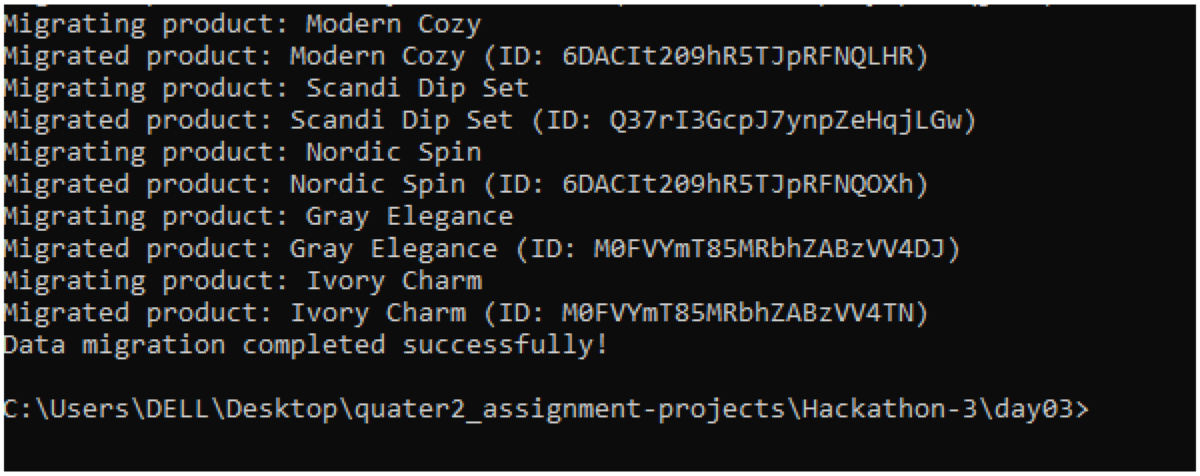


```
:\Users\DELL\Desktop\quater2_assignment-projects\Hackathon-3\day03>npm install @sanity/client axios dotenv

added 12 packages, removed 8 packages, changed 4 packages, and audited 1305 packages in 2m

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

1 moderate severity vulnerability

To address all issues, run:
  npm audit fix

Run `npm audit` for details.

C:\Users\DELL\Desktop\quater2_assignment-projects\Hackathon-3\day03>
```

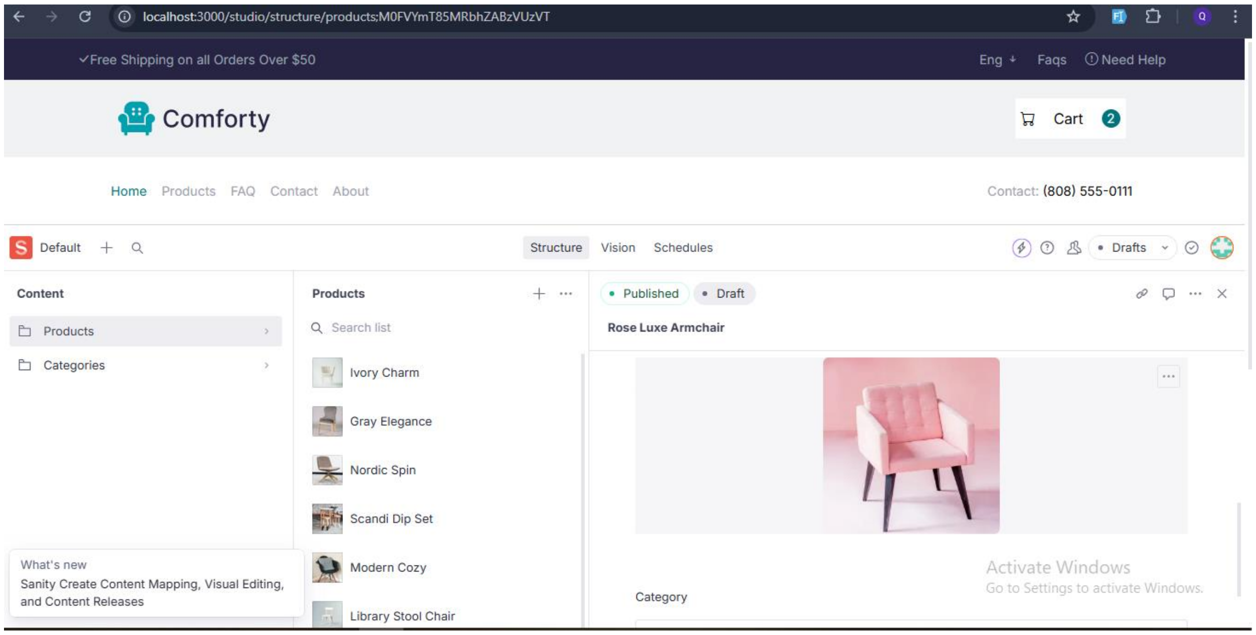
1. Now run the command `npm run migrate`



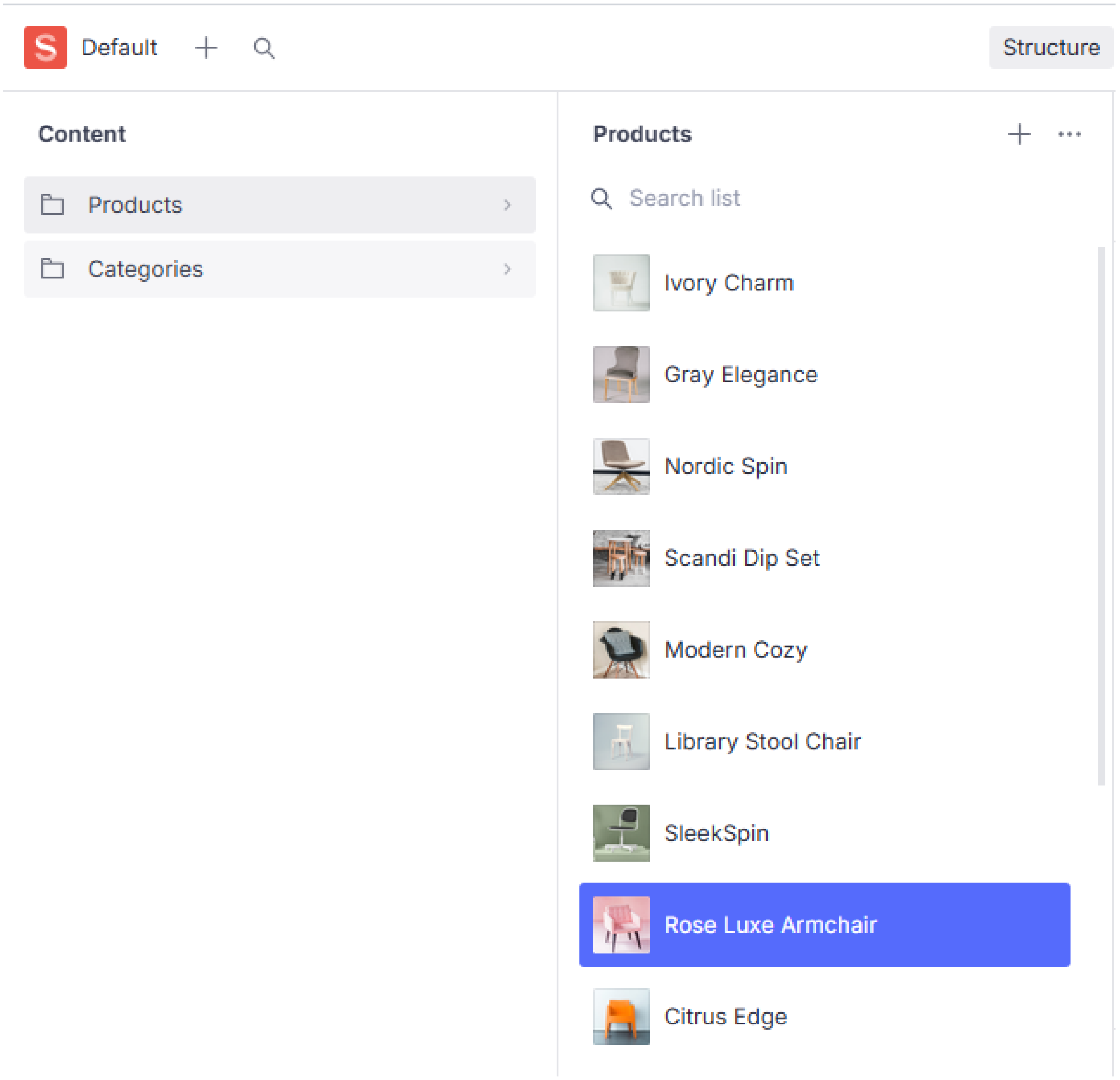
```
:\Users\DELL\Desktop\quater2_assignment-projects\Hackathon-3\day03>npm run migrate

 hackathon02@0.1.0 migrate
 node scripts/migrate.mjs

tarting data migration...
igrating category: Wing Chair
igrated category: Wing Chair (ID: 26fd7176-3c4d-40fc-a73a-3b85a9b5e15f)
igrating category: Wooden Chair
igrated category: Wooden Chair (ID: 407a8583-6203-4f61-becf-8e8b4c5461b6)
igrating category: Desk Chair
igrated category: Desk Chair (ID: b5710116-09af-4d0e-aa9a-dcd02fe919a9)
igrating product: SleekSpin
igrated product: SleekSpin (ID: Q37rI3GcpJ7ynpZeHqjC9X)
igrating product: Citrus Edge
igrated product: Citrus Edge (ID: Q37rI3GcpJ7ynpZeHqjCju)
igrating product: Rose Luxe Armchair
igrated product: Rose Luxe Armchair (ID: Q37rI3GcpJ7ynpZeHqjERi)
igrating product: Library Stool Chair
igrated product: Library Stool Chair (ID: 6DACIt209hR5TJpRFNQFfh)
igrating product: Modern Cozy
igrated product: Modern Cozy (ID: M0FVYmT85MRbhZABzVUvpt)
igrating product: Scandi Dip Set
igrated product: Scandi Dip Set (ID: 6DACIt209hR5TJpRFNQGhZ)
igrating product: Citrus Edge
igrated product: Citrus Edge (ID: M0FVYmT85MRbhZABzVUyAT)
igrating product: Rose Luxe Armchair
igrated product: Rose Luxe Armchair (ID: M0FVYmT85MRbhZABzVUzVT)
igrating product: SleekSpin
igrated product: SleekSpin (ID: 6DACIt209hR5TJpRFNQJuH)
igrating product: Library Stool Chair
igrated product: Library Stool Chair (ID: Q37rI3GcpJ7ynpZeHqjJcR)
igrating product: Modern Cozy
igrated product: Modern Cozy (ID: 6DACIt209hR5TJpRFNQLHR)
igrating product: Scandi Dip Set
```

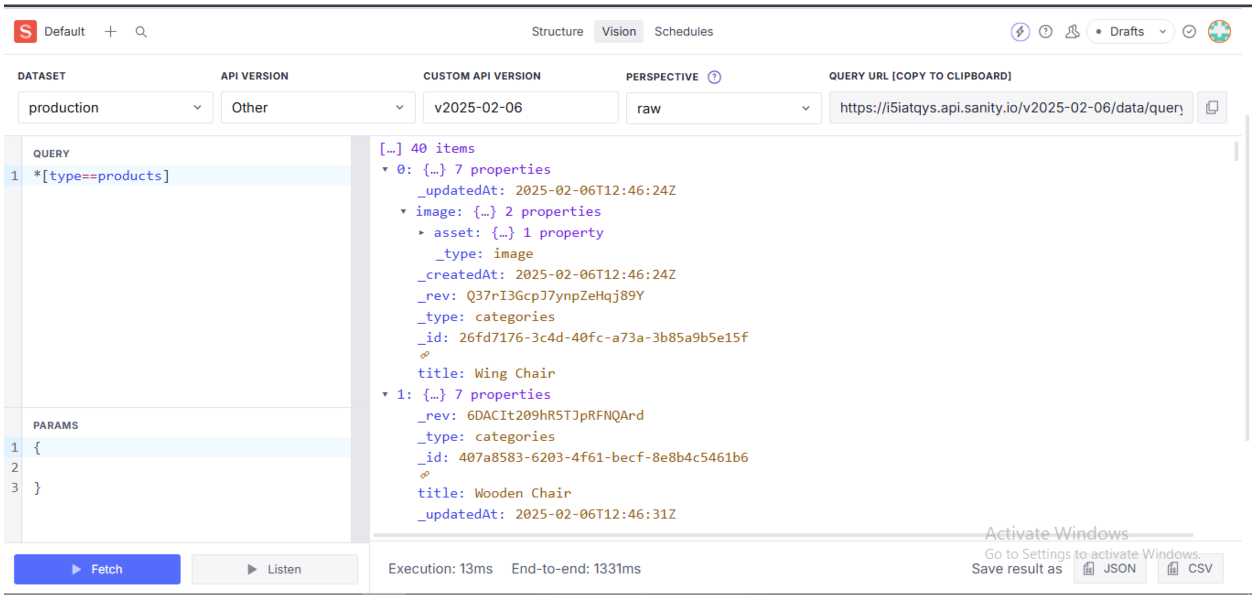


```
Migrating product: Modern Cozy
Migrated product: Modern Cozy (ID: 6DACIt209hR5TJpRFNQLHR)
Migrating product: Scandi Dip Set
Migrated product: Scandi Dip Set (ID: Q37rI3GcpJ7ynpZeHqjLGw)
Migrating product: Nordic Spin
Migrated product: Nordic Spin (ID: 6DACIt209hR5TJpRFNQOXh)
Migrating product: Gray Elegance
Migrated product: Gray Elegance (ID: M0FVYmT85MRbhZABzVV4DJ)
Migrating product: Ivory Charm
Migrated product: Ivory Charm (ID: M0FVYmT85MRbhZABzVV4TN)
Data migration completed successfully!

C:\Users\DELL\Desktop\quater2_assignment-projects\Hackathon-3\day03>
```



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


S
Default
Structure
Content
Products
Categories
Products
Search list
Ivory Charm
Gray Elegance
Nordic Spin
Scandi Dip Set
Modern Cozy
Library Stool Chair
SleekSpin
Rose Luxe Armchair
Citrus Edge



DATASET production
API VERSION Other
CUSTOM API VERSION v2025-02-06
PERSPECTIVE raw
QUERY URL [COPY TO CLIPBOARD] https://i5iatqys.api.sanity.io/v2025-02-06/data/query

QUERY
```
*[type==products]
```

PARAMS
```
{

}
```

```
[…] 40 items
▾ 0: {…} 7 properties
    _updatedAt: 2025-02-06T12:46:24Z
  ▾ image: {…} 2 properties
    ▸ asset: {…} 1 property
      _type: image
    _createdAt: 2025-02-06T12:46:24Z
    _rev: Q37rI3GcpJ7ynpZeHqj89Y
    _type: categories
    _id: 26fd7176-3c4d-40fc-a73a-3b85a9b5e15f
    title: Wing Chair
▾ 1: {…} 7 properties
    _rev: 6DACIt209hR5TJpRFNQArd
    _type: categories
    _id: 407a8583-6203-4f61-becf-8e8b4c5461b6
    title: Wooden Chair
    _updatedAt: 2025-02-06T12:46:31Z
```

Fetch Listen

Execution: 13ms End-to-end: 1331ms

Save result as JSON CSV