

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
```

S

Fatimamehmoodali1

learning-sanity-project

30 days left in trial

LE

Fatimamehmoodali1

learning-sanity-project

PLAN

Growth Trial

STATUS

Active

PROJECT ID

isiatqys

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

S

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.

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

+ Add CORS origin

S

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.

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

☐ Read and write access to all datasets, with no access to project settings.

+ Add API token

It's new

Create Content Mapping, Visual Editing, Content Releases

Activate Windows

Go to Settings to activate Windows.

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

+ Add API token

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!

```
sk9aZl4twbncpDOHTN4s47D5o5w1Uke9AD908zfU8FR5pqvctDf9nrFwRtoV1xQsByk9N60DbJSPGVcJ8mk3R0b6RDhUis6q3KB5S61bRn9X1hyJH5T0ZRHmxq9vgZnsnW1dhGcONrcSawYnGI sKX96bRkKLFoPTS3VRbVjm0b7qGDNkHTZ
```

Sanity Schema:

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

- Products.ts:**

```
1 import { defineType } from "sanity";
2
3 export const productSchema = defineType({
4   name: "products",
5   title: "Products",
6   type: "document",
7   fields: [
8     {
9       name: "title",
10      title: "Product Title",
11      type: "string",
12    },
13    {
14      name: "price",
15      title: "Price",
16      type: "number",
17    },
18    {
19      title: "Price without Discount",
20      name: "priceWithoutDiscount",
21      type: "number",
22    },
23    {
24      name: "badge",
25      title: "Badge",
26      type: "string",
27    },
28  ],
29 });
```

The screenshot shows the VS Code editor with the file explorer on the left. The file explorer shows the project structure with folders like .next, node_modules, public, src, app, components, pages, sanity, lib, and schemaTypes. The schemaTypes folder is expanded, showing categories.ts, index.ts, products.ts, env.ts, structure.ts, .env.local, .eslintrc.json, .gitignore, next-env.d.ts, next.config.ts, package-lock.json, package.json, postcss.config.js, README.md, sanity.d.ts, sanity.config.ts, tailwind.config.ts, and tsconfig.json. The products.ts file is open in the editor, showing the productSchema definition. The code is as follows:

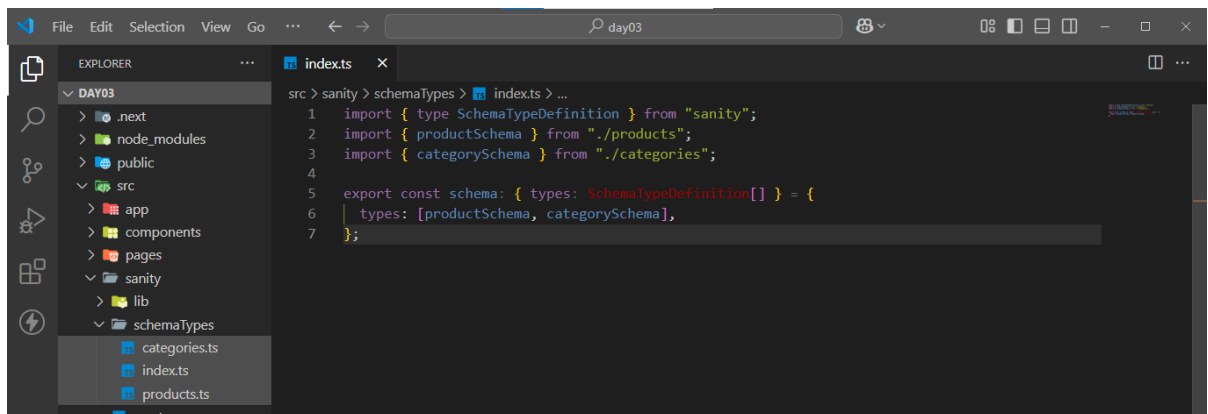
```
src > sanity > schemaTypes > products.ts > productsSchema > fields
3 export const productSchema = defineType({
7   fields: [
28     {
29       name: "image",
30       title: "Product Image",
31       type: "image",
32     },
33     {
34       name: "category",
35       title: "Category",
36       type: "reference",
37       to: [{ type: "categories" }],
38     },
39     {
40       name: "description",
41       title: "Product Description",
42       type: "text",
43     },
44     {
45       name: "inventory",
46       title: "Inventory Management",
47       type: "number",
48     },
49     {
50       name: "tags",
51       title: "Tags",
52       type: "array",
53       of: [{ type: "string" }],
54       options: {
55         list: [
56           { title: "Featured", value: "featured" },
57           { title: "Follow products and discounts on Instagram", value: "instagram" },
58           { title: "Gallery", value: "gallery" },
59         ],
60       },
61     },
62   ],
63 },
64 );
```

- **Categories.ts:**

The screenshot shows the VS Code editor with the file explorer on the left. The file explorer shows the project structure with folders like .next, node_modules, public, src, app, components, pages, sanity, lib, and schemaTypes. The schemaTypes folder is expanded, showing categories.ts, index.ts, products.ts, env.ts, structure.ts, .env.local, .eslintrc.json, .gitignore, next-env.d.ts, next.config.ts, package-lock.json, package.json, postcss.config.js, README.md, sanity.d.ts, sanity.config.ts, tailwind.config.ts, and tsconfig.json. The categories.ts file is open in the editor, showing the categorySchema definition. The code is as follows:

```
src > sanity > schemaTypes > categories.ts > ...
1 import { defineType } from "sanity";
2
3 export const categorySchema = defineType({
4   name: 'categories',
5   title: 'Categories',
6   type: 'document',
7   fields: [
8     {
9       name: 'title',
10      title: 'Category Title',
11      type: 'string',
12    },
13    {
14      name: 'image',
15      title: 'Category Image',
16      type: 'image',
17    },
18    {
19      title: 'Number of Products',
20      name: 'products',
21      type: 'number',
22    },
23  ],
24 });
```

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



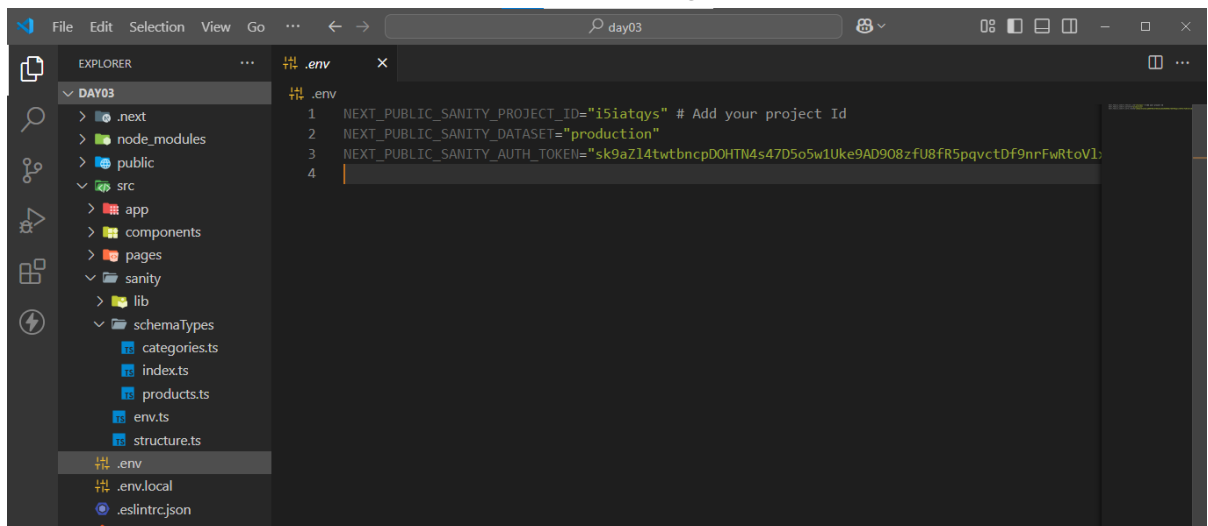
The screenshot shows the VS Code editor with the Explorer sidebar on the left. The 'src' directory is expanded, showing 'sanity' and 'schemaTypes'. Inside 'schemaTypes', 'index.ts' is selected. The main editor displays the content of 'index.ts'.

```
src > sanity > schemaTypes > index.ts > ...
1  import { type SchemaTypeDefinition } from "sanity";
2  import { productSchema } from "../products";
3  import { categorySchema } from "../categories";
4
5  export const schema: { types: SchemaTypeDefinition[] } = {
6    types: [productSchema, categorySchema],
7  };
8
```

- 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:



The screenshot shows the VS Code editor with the Explorer sidebar on the left. The root directory is expanded, showing files like '.env', '.env.local', and '.eslintrc.json'. The '.env' file is selected, and its content is displayed in the main editor.

```
1  NEXT_PUBLIC_SANITY_PROJECT_ID="i5iatqys" # Add your project Id
2  NEXT_PUBLIC_SANITY_DATASET="production"
3  NEXT_PUBLIC_SANITY_AUTH_TOKEN="sk9aZl4twbncpDOHTN4s47D5o5w1Uke9AD908zfU8fR5pqvctDf9nrFwRtoV1:
4
```

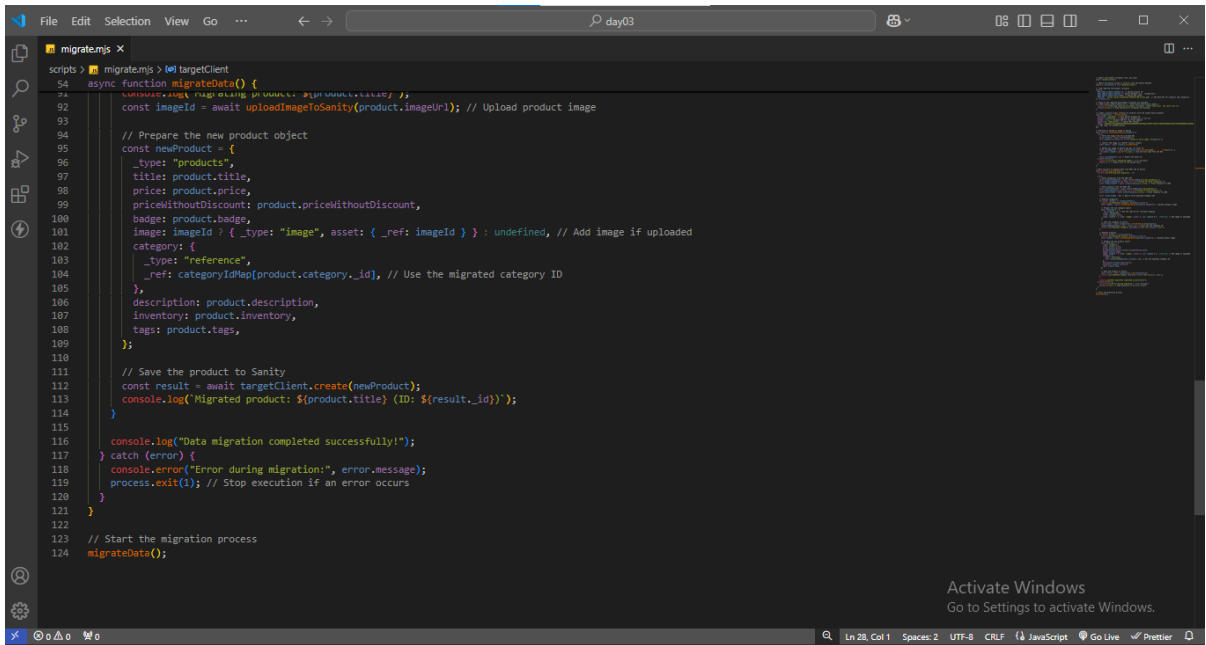
- Create a script folder manually in the root of your project
- Create **migrate.mjs** inside of the script folder

```
File Edit Selection View Go ... day03
migrate.mjs X
scripts > migrate.mjs > @targetClient
1 // Import environment variables from .env.local
2 import {dotenv} from "dotenv/config";
3
4 // Import the Sanity client to interact with the Sanity backend
5 import { createClient } from "@sanity/client";
6
7 // Load required environment variables
8 const {
9   NEXT_PUBLIC_SANITY_PROJECT_ID, // Sanity project ID
10   NEXT_PUBLIC_SANITY_DATASET, // Sanity dataset (e.g., "production")
11   NEXT_PUBLIC_SANITY_AUTH_TOKEN, // Sanity API token
12   BASE_URL = "https://gliaic-hackathon-template-00.vercel.app", // API base URL for products and categories
13 } = process.env;
14
15 // Check if the required environment variables are provided
16 if (!NEXT_PUBLIC_SANITY_PROJECT_ID || !NEXT_PUBLIC_SANITY_AUTH_TOKEN) {
17   console.error("Missing required environment variables. Please check your .env.local file.");
18   process.exit(1); // Stop execution if variables are missing
19 }
20
21 // Create a Sanity client instance to interact with the target Sanity dataset
22 const targetClient = createClient({
23   projectId: "151stqys", // Your Sanity project ID
24   dataset: "production", // Default to "production" if not set
25   useCdn: false, // Disable CDN for real-time updates
26   apiVersion: "2022-01-01", // Sanity API version
27   token: "sk9a21atutncrD0G7Hds4705o5u1lka9AD908zFUEFR85pqvctDF9nrFultotLq58yK9W6C0B7SPQVc7Bek3R0b6R0HUISeq3KBS561bRm9XJhy3h5T8Z8eqvpgVznsndkdGhc0ncrSawVn01sKX9G6RkKLFpT53VR0Vj9eb7qGDMkH7Z",
28   // API token for authentication
29 });
30
31 // Function to upload an image to Sanity
32 async function uploadImageToSanity(imageUrl) {
33   try {
34     // Fetch the image from the provided URL
35     const response = await fetch(imageUrl);
36     if (!response.ok) throw new Error("Failed to fetch image: ${imageUrl}");
37
38     // Convert the image to a buffer (Binary format)
39     const buffer = await response.arrayBuffer();
40
41     // Upload the image to Sanity and get its asset ID
42     const uploadedAsset = await targetClient.assets.upload("image", Buffer.from(buffer), {
43       filename: imageUrl.split("/").pop(), // Use the file name from the URL
44     });
45
46     return uploadedAsset._id; // Return the asset ID
47   } catch (error) {
48     console.error("Error uploading image:", error.message);
49   }
50 }
51
52 // Main function to migrate data from REST API to Sanity
53 async function migrateData() {
54   console.log("Starting data migration...");
55
56   try {
57     // Fetch categories from the REST API
58     const categoriesResponse = await fetch(`${BASE_URL}/api/categories`);
59     if (!categoriesResponse.ok) throw new Error("Failed to fetch categories.");
60     const categoriesData = await categoriesResponse.json(); // Parse response to JSON
61
62     // Fetch products from the REST API
63     const productsResponse = await fetch(`${BASE_URL}/api/products`);
64     if (!productsResponse.ok) throw new Error("Failed to fetch products.");
65     const productsData = await productsResponse.json(); // Parse response to JSON
66
67     const categoryIdMap = {}; // Map to store migrated category IDs
68
69     // Migrate categories
70     for (const category of categoriesData) {
71       console.log("Migrating category: ${category.title}");
72       const imageUrl = await uploadImageToSanity(category.imageUrl); // Upload category image
73
74       // Prepare the new category object
75       const newCategory = {
76         _id: category._id, // Use the same ID for reference mapping
77         type: "categories",
78         title: category.title,
79         image: imageUrl ? { _type: "image", asset: { _ref: imageUrl } } : undefined, // Add image if uploaded
80       };
81
82       // Save the category to Sanity
83       const result = await targetClient.createOrReplace(newCategory);
84       categoryIdMap[category._id] = result._id; // Store the new category ID
85       console.log("Migrated category: ${category.title} (ID: ${result._id})");
86     }
87
88     // Migrate products
89     for (const product of productsData) {
90       console.log("Migrating product: ${product.title}");
91       const imageUrl = await uploadImageToSanity(product.imageUrl); // Upload product image
92
93       // Prepare the new product object
94       const newProduct = {
95         _id: product._id, // Use the same ID for reference mapping
96         type: "products",
97         title: product.title,
98         image: imageUrl ? { _type: "image", asset: { _ref: imageUrl } } : undefined, // Add image if uploaded
99       };
100
101       // Save the product to Sanity
102       const result = await targetClient.createOrReplace(newProduct);
103       productIdMap[product._id] = result._id; // Store the new product ID
104       console.log("Migrated product: ${product.title} (ID: ${result._id})");
105     }
106
107     // Log the migration completion
108     console.log("Data migration completed successfully!");
109   } catch (error) {
110     console.error("Error during data migration:", error.message);
111   }
112 }
113
114 // Execute the migrateData function
115 migrateData().catch(console.error);
```

Activate Windows
Go to Settings to activate Windows.

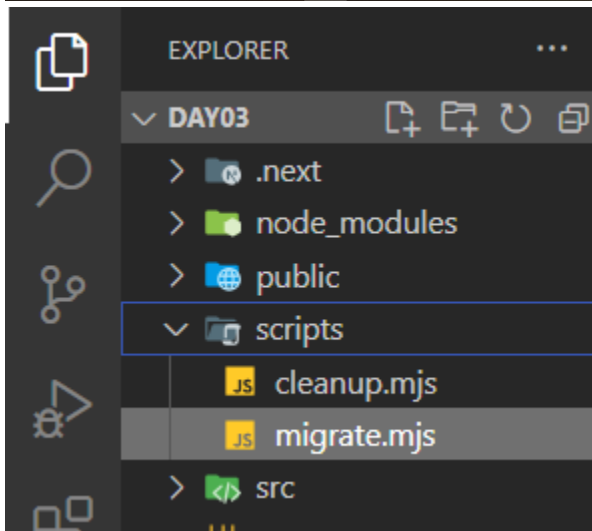
```
File Edit Selection View Go ... day03
migrate.mjs X
scripts > migrate.mjs > @targetClient
52 async function uploadImageToSanity(imageUrl) {
53   console.error("Error uploading image:", error.message);
54   return null; // Return null if the upload fails
55 }
56
57 // Main function to migrate data from REST API to Sanity
58 async function migrateData() {
59   console.log("Starting data migration...");
60
61   try {
62     // Fetch categories from the REST API
63     const categoriesResponse = await fetch(`${BASE_URL}/api/categories`);
64     if (!categoriesResponse.ok) throw new Error("Failed to fetch categories.");
65     const categoriesData = await categoriesResponse.json(); // Parse response to JSON
66
67     // Fetch products from the REST API
68     const productsResponse = await fetch(`${BASE_URL}/api/products`);
69     if (!productsResponse.ok) throw new Error("Failed to fetch products.");
70     const productsData = await productsResponse.json(); // Parse response to JSON
71
72     const categoryIdMap = {}; // Map to store migrated category IDs
73
74     // Migrate categories
75     for (const category of categoriesData) {
76       console.log("Migrating category: ${category.title}");
77       const imageUrl = await uploadImageToSanity(category.imageUrl); // Upload category image
78
79       // Prepare the new category object
80       const newCategory = {
81         _id: category._id, // Use the same ID for reference mapping
82         type: "categories",
83         title: category.title,
84         image: imageUrl ? { _type: "image", asset: { _ref: imageUrl } } : undefined, // Add image if uploaded
85       };
86
87       // Save the category to Sanity
88       const result = await targetClient.createOrReplace(newCategory);
89       categoryIdMap[category._id] = result._id; // Store the new category ID
90       console.log("Migrated category: ${category.title} (ID: ${result._id})");
91     }
92
93     // Migrate products
94     for (const product of productsData) {
95       console.log("Migrating product: ${product.title}");
96       const imageUrl = await uploadImageToSanity(product.imageUrl); // Upload product image
97
98       // Prepare the new product object
99       const newProduct = {
100         _id: product._id, // Use the same ID for reference mapping
101         type: "products",
102         title: product.title,
103         image: imageUrl ? { _type: "image", asset: { _ref: imageUrl } } : undefined, // Add image if uploaded
104       };
105
106       // Save the product to Sanity
107       const result = await targetClient.createOrReplace(newProduct);
108       productIdMap[product._id] = result._id; // Store the new product ID
109       console.log("Migrated product: ${product.title} (ID: ${result._id})");
110     }
111
112     // Log the migration completion
113     console.log("Data migration completed successfully!");
114   } catch (error) {
115     console.error("Error during data migration:", error.message);
116   }
117 }
118
119 // Execute the migrateData function
120 migrateData().catch(console.error);
```

Activate Windows
Go to Settings to activate Windows.

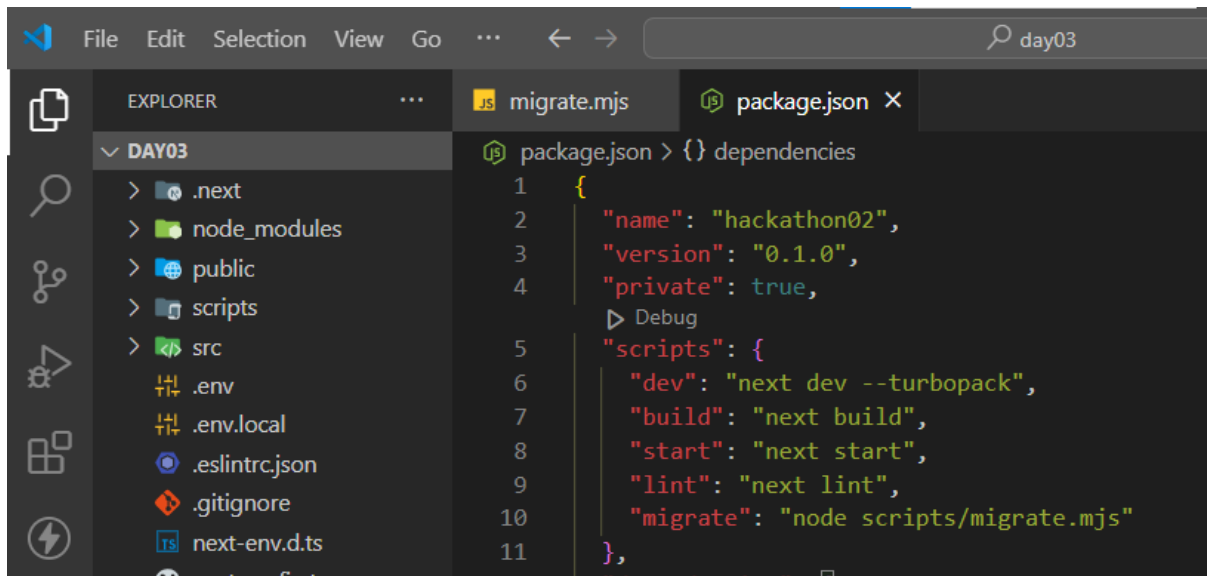


The screenshot shows a VS Code editor window with a file named 'migrate.mjs' open. The code is written in JavaScript and includes comments in Chinese. It defines an async function 'migrateData()' that performs a data migration. The function uploads a product image to Sanity, prepares a new product object with fields like title, price, badge, image, category, description, inventory, and tags, and then saves it to Sanity using 'targetClient.create()'. It includes error handling and a success log. The file is located in the 'scripts' directory of a project named 'day03'.

```
54 // migrate.mjs > | targetClient
55 async function migrateData() {
56   // Upload product image
57   const imageUrl = await uploadImageToSanity(product.imageUrl); // Upload product image
58
59   // Prepare the new product object
60   const newProduct = {
61     _type: "products",
62     title: product.title,
63     price: product.price,
64     priceWithoutDiscount: product.priceWithoutDiscount,
65     badge: product.badge,
66     image: imageUrl ? { _type: "image", asset: { _ref: imageUrl } } : undefined, // Add image if uploaded
67     category: {
68       _type: "reference",
69       _ref: categoryIdMap[product.category_id], // Use the migrated category ID
70     },
71     description: product.description,
72     inventory: product.inventory,
73     tags: product.tags,
74   };
75
76   // Save the product to Sanity
77   const result = await targetClient.create(newProduct);
78   console.log('Migrated product: ${product.title} (ID: ${result._id})');
79 }
80
81 console.log("Data migration completed successfully!");
82 } catch (error) {
83   console.error("Error during migration:", error.message);
84   process.exit(1); // Stop execution if an error occurs
85 }
86
87 // Start the migration process
88 migrateData();
```



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



-
- Install the following package before running the script

```
C:\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`


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

```
> hackathon02@0.1.0 migrate  
> node scripts/migrate.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: Q37rI3GcpJ7ynpZeHqjC9X)
```

```
Migrating product: Citrus Edge
```

```
Migrated product: Citrus Edge (ID: Q37rI3GcpJ7ynpZeHqjCju)
```

```
Migrating product: Rose Luxe Armchair
```

```
Migrated product: Rose Luxe Armchair (ID: Q37rI3GcpJ7ynpZeHqjERi)
```

```
Migrating product: Library Stool Chair
```

```
Migrated product: Library Stool Chair (ID: 6DACIt209hR5TJpRFNQFfh)
```

```
Migrating product: Modern Cozy
```

```
Migrated product: Modern Cozy (ID: M0FVYmT85MRbhZABzVUvpt)
```

```
Migrating product: Scandi Dip Set
```

```
Migrated product: Scandi Dip Set (ID: 6DACIt209hR5TJpRFNQGhZ)
```

```
Migrating product: Citrus Edge
```

```
Migrated product: Citrus Edge (ID: M0FVYmT85MRbhZABzVUyAT)
```

```
Migrating product: Rose Luxe Armchair
```

```
Migrated product: Rose Luxe Armchair (ID: M0FVYmT85MRbhZABzVUzVT)
```

```
Migrating product: SleekSpin
```

```
Migrated product: SleekSpin (ID: 6DACIt209hR5TJpRFNQJuH)
```

```
Migrating product: Library Stool Chair
```

```
Migrated product: Library Stool Chair (ID: Q37rI3GcpJ7ynpZeHqjJcR)
```

```
Migrating product: Modern Cozy
```

```
Migrated product: Modern Cozy (ID: 6DACIt209hR5TJpRFNQLHR)
```

```
Migrating 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
```

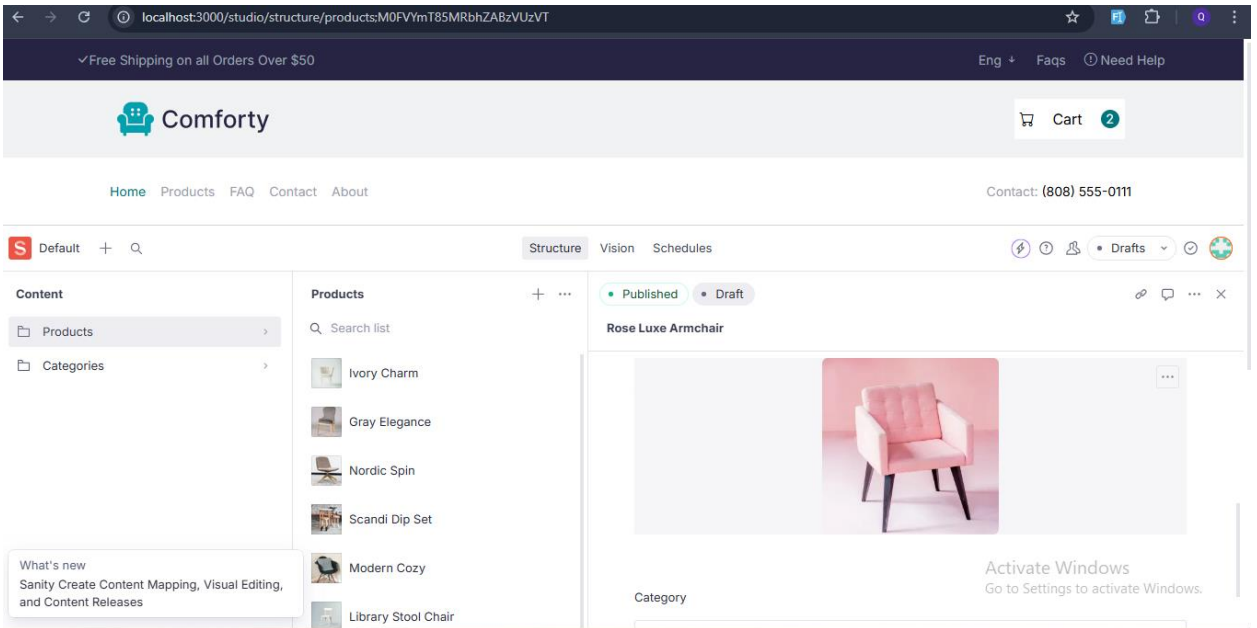
```
Migrated product: Gray Elegance (ID: M0FVYmT85MRbhZABzVV4DJ)
```

```
Migrating product: Ivory Charm
```

```
Migrated product: Ivory Charm (ID: M0FVYmT85MRbhZABzVV4TN)
```

```
Data migration completed successfully!
```

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





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

S

Default

+

Q

Structure

Vision

Schedules

Drafts

DATASET

production

API VERSION

Other

CUSTOM API VERSION

v2025-02-06

PERSPECTIVE ⓘ

raw

QUERY URL [COPY TO CLIPBOARD]

https://5iatqys.api.sanity.io/v2025-02-06/data/query

QUERY

1

*[type==products]

PARAMS

1

{

2

3

}

[...] 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: 6DACIt209hRSTJpRFNQAr

_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

Activate Windows

Go to Settings to activate Windows.

Save result as

JSON

CSV