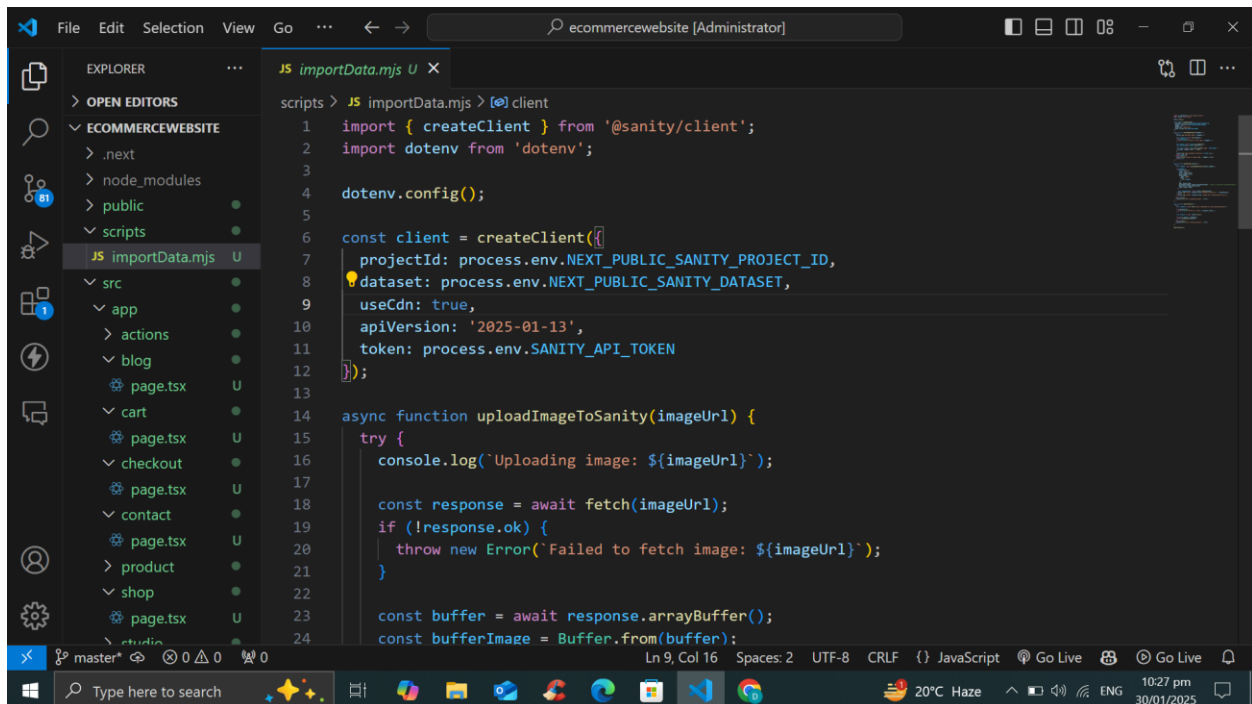


DAY 3 - API INTEGRATION

AND DATA MIGRATION

Data Migration:

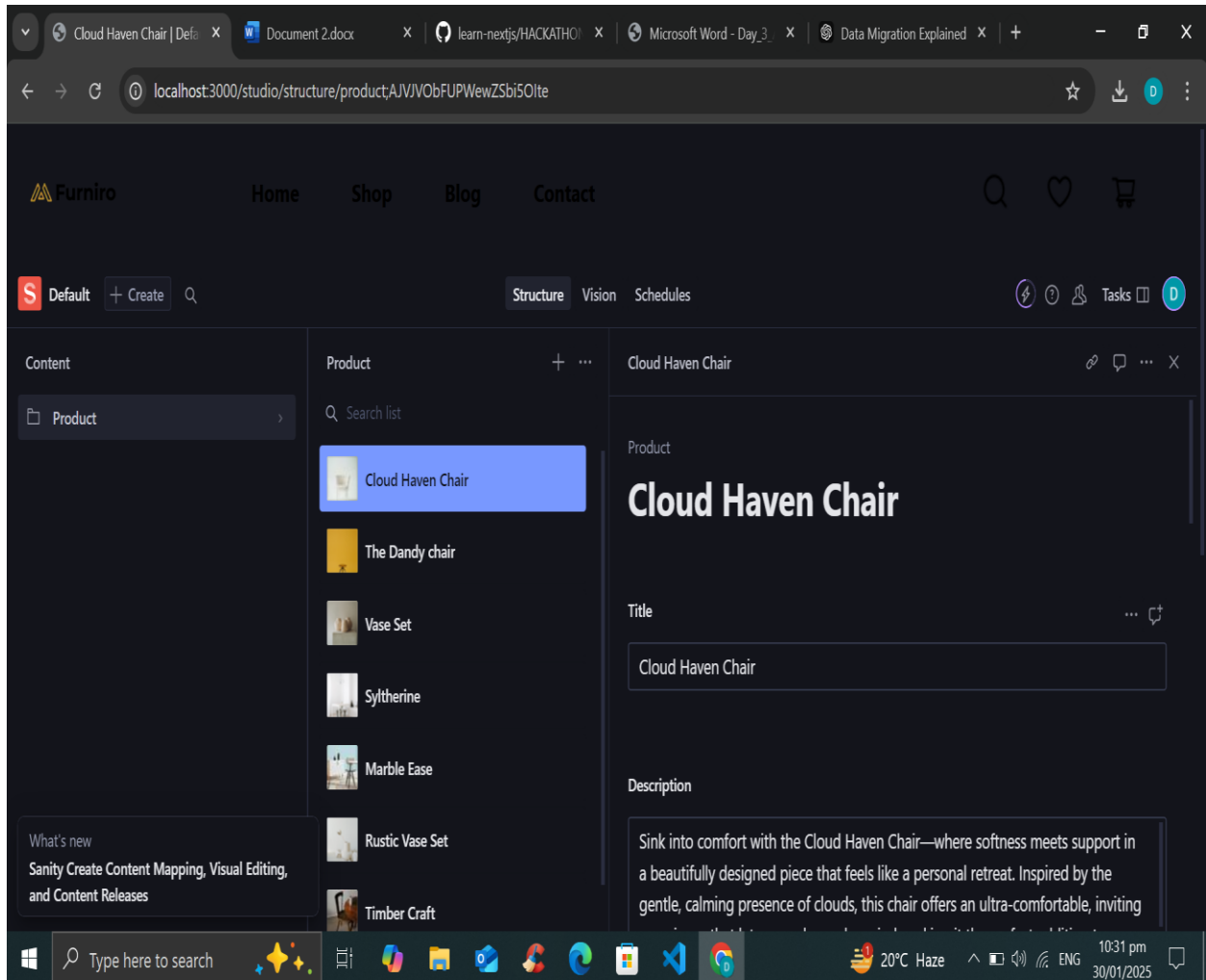
Migrating data into Sanity CMS to build a functional marketplace backend.. Sanity allows developers to structure, manage, and query content in a flexible and API-driven way, often used in modern web and app development. Migrating data into Sanity means you're moving or importing existing content from another system or database into Sanity's content structure.



```
scripts > JS importData.mjs > client
1 import { createClient } from '@sanity/client';
2 import dotenv from 'dotenv';
3
4 dotenv.config();
5
6 const client = createClient({
7   projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
8   dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
9   useCdn: true,
10  apiVersion: '2025-01-13',
11  token: process.env.SANITY_API_TOKEN
12 });
13
14 async function uploadImageToSanity(imageUrl) {
15   try {
16     console.log(`Uploading image: ${imageUrl}`);
17
18     const response = await fetch(imageUrl);
19     if (!response.ok) {
20       throw new Error(`Failed to fetch image: ${imageUrl}`);
21     }
22
23     const buffer = await response.arrayBuffer();
24     const bufferImage = Buffer.from(buffer);
```

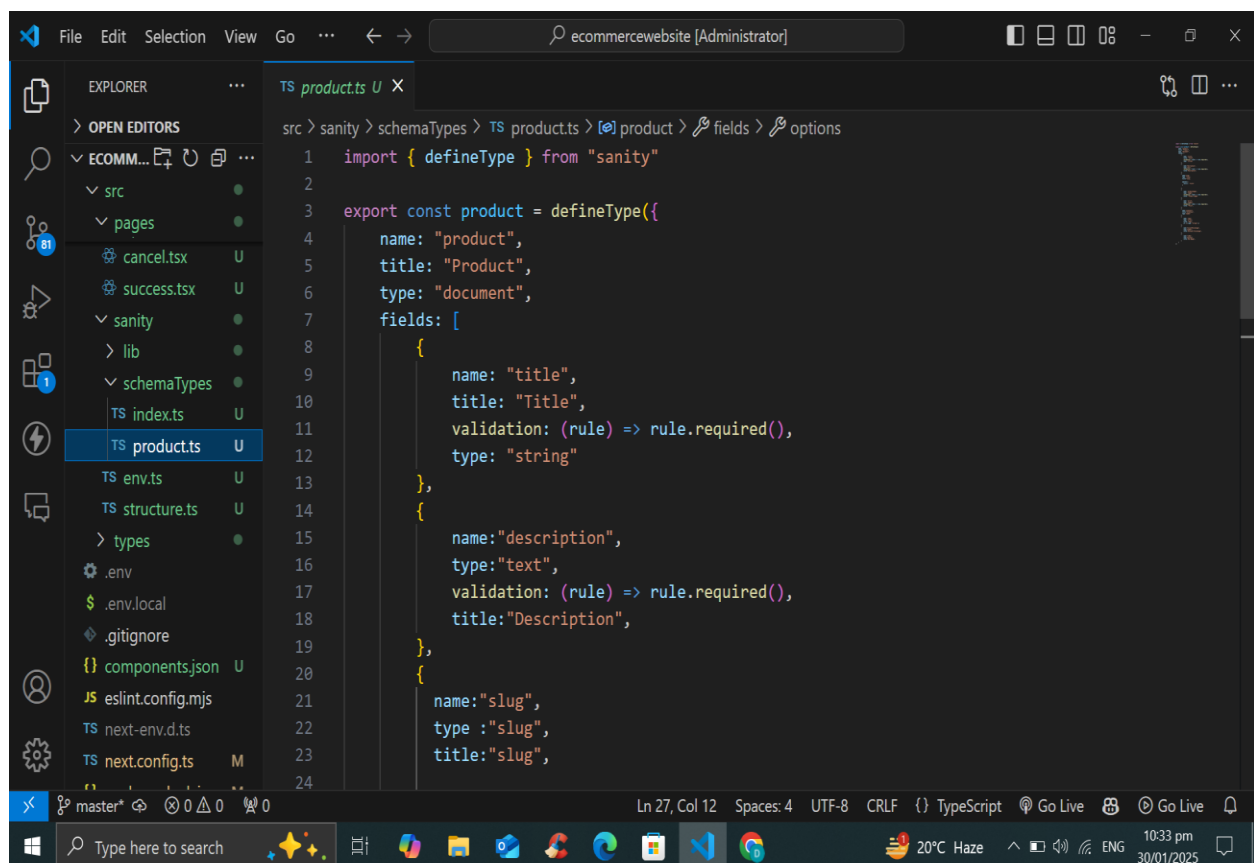
Sanity Studio:

Here is an Sanity Studio Where I migrated my data



Here is my Schema Of My Product:

In **Sanity**, a **schema** defines the structure of the content you want to manage and how it will be stored. It's essentially a blueprint that specifies the types of content (like blog posts, products, users, etc.) and their respective fields (like titles, images, dates, etc.). The schema determines how content is created, organized, and queried in Sanity.



The screenshot shows a Visual Studio Code editor window with a TypeScript file named `products.ts` open. The file is located at `src > sanity > schemaTypes > TS products.ts`. The code defines a Sanity schema for a product with the following fields:

```
1  import { defineType } from "sanity"
2
3  export const product = defineType({
4    name: "product",
5    title: "Product",
6    type: "document",
7    fields: [
8      {
9        name: "title",
10       title: "Title",
11       validation: (rule) => rule.required(),
12       type: "string"
13     },
14     {
15       name: "description",
16       type: "text",
17       validation: (rule) => rule.required(),
18       title: "Description",
19     },
20     {
21       name: "slug",
22       type: "slug",
23       title: "slug",
24     }
25   ]
26 })
```

The Explorer sidebar on the left shows the project structure, including files like `cancel.tsx`, `success.tsx`, `sanity`, `lib`, `schemaTypes`, `TS index.ts`, `TS products.ts` (selected), `TS env.ts`, `TS structure.ts`, `types`, `.env`, `.env.local`, `.gitignore`, `components.json`, `eslint.config.mjs`, `next-env.d.ts`, and `next.config.ts`. The status bar at the bottom indicates the current line and column (Ln 27, Col 12), encoding (UTF-8), line endings (CRLF), and the file type (TypeScript).

