

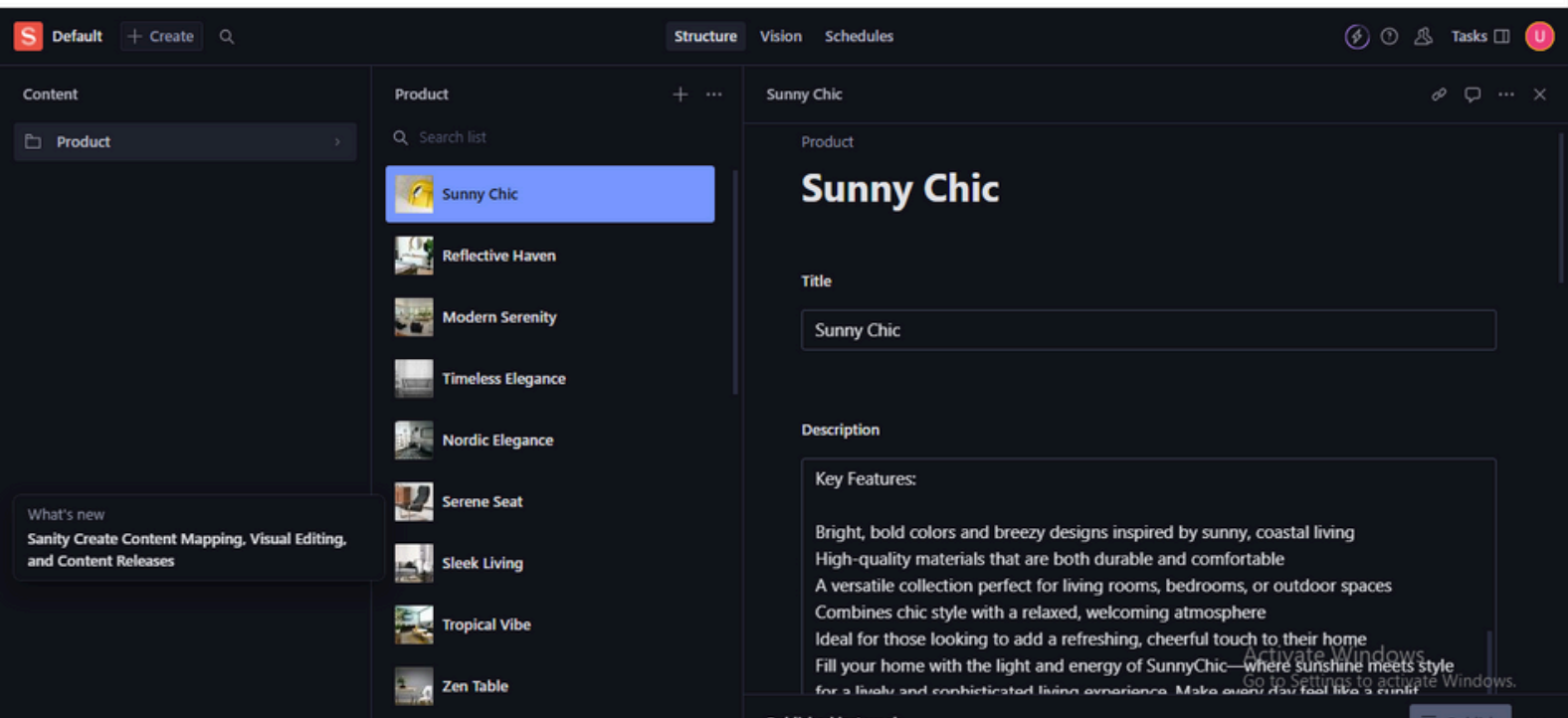
# Documentation of Day 3:

## API INTEGRATION AND DATA MIGRATION steps:

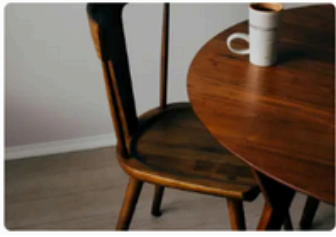
- Created a new project in Sanity using the Sanity CLI.
- Installed the Sanity project into my Next.js project and configured the Sanity client.
- Fetched data into Sanity Studio using the provided API URL.
- Added the provided schema to Sanity.
- Verified that the data was successfully displayed in Sanity Studio and on the frontend..

## Screenshots of data displayed on your frontend and populated in Sanity CMS

Data within Sanity and in the Browser



## Products From Api's data



### Timber Craft

Introducing TimberCraft—a collection that celebrates the timeless beauty of wood craftsmanship and t...

\$ 320

wooden craftsmanship furniture modern nature inspired

Add to Cart



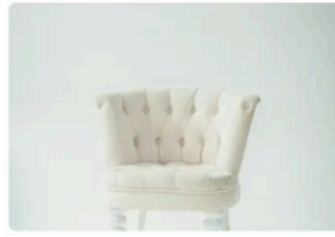
### Amber Haven

Step into a world of warmth and tranquility with Amber Haven—a collection inspired by the golden glo...

\$ 150

amber luxury cozy elegant furniture

Add to Cart



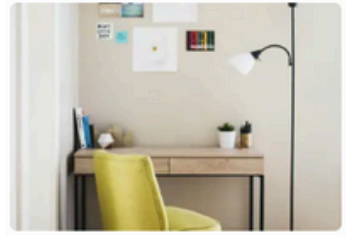
### Cloud Haven Chair

Sink into comfort with the Cloud Haven Chair—where softness meets support in a beautifully designed ...

\$ 230

cloud chair comfy home decor modern furniture

Add to Cart



### Bright Space

Welcome to BrightSpace—a collection designed to infuse your home with light, energy, and vibrancy. I...

\$ 180

bright space minimalist modern decor

Add to Cart

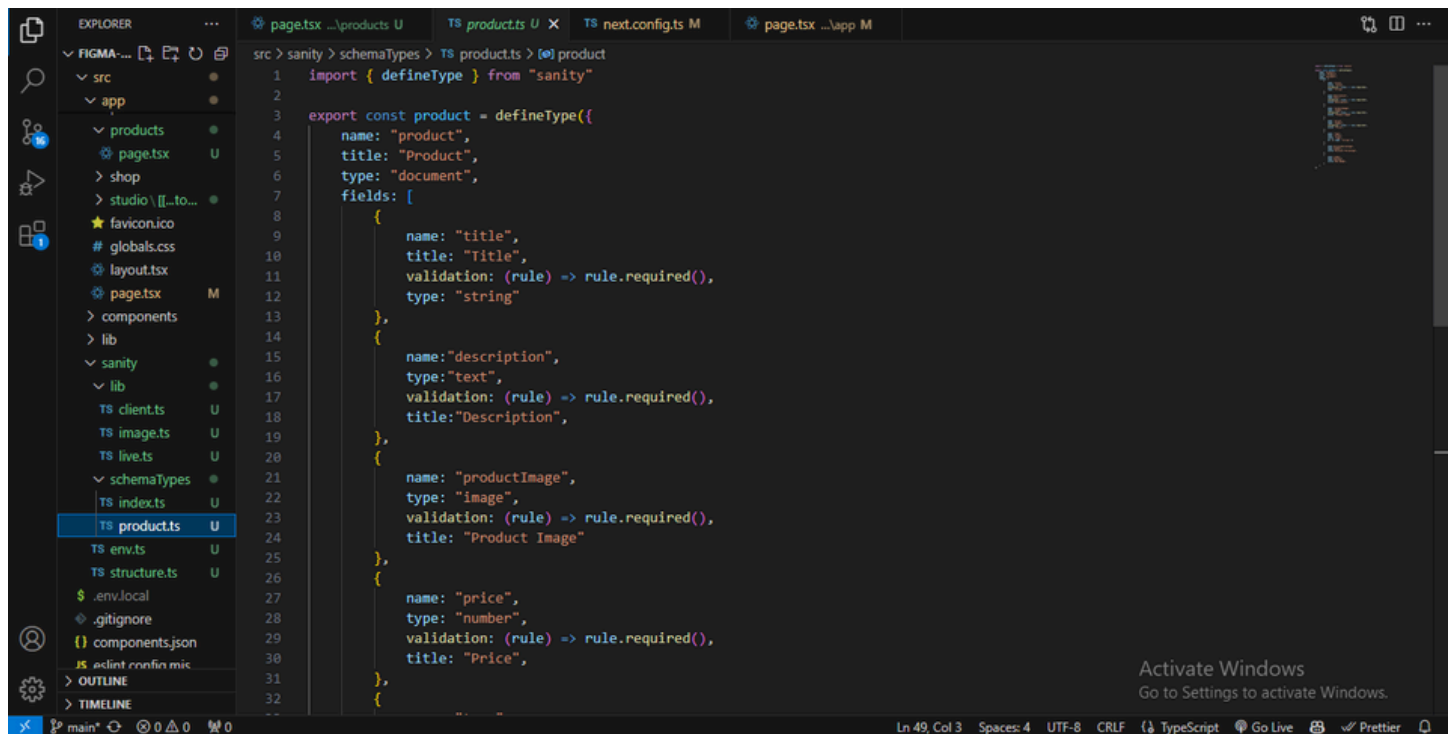
Activate Windows  
Go to Settings to activate Windows.

# Code snippets of integration and migration scripts

## migration file

```
script > JS importData.js > client
1 import { createClient } from '@sanity/client';
2
3 const client = createClient({
4   projectId: 'f0y4t0re',
5   dataset: 'production',
6   useCdn: true,
7   apiVersion: '2025-01-13',
8   token: 'skGRZrVKyAGV06R2Vo6kf0L1v3q4zDmGrVBp4NZRhozFNgaoLHOzAOMQh9bR6lXO1RksE64B3i5wGwpX0pw5Pz0B5XsnfARmKnAq59ncrv0JRAM8'
9 });
10
11 async function uploadImageToSanity(imageUrl) {
12   try {
13     console.log(`Uploading image: ${imageUrl}`);
14
15     const response = await fetch(imageUrl);
16     if (!response.ok) {
17       throw new Error(`Failed to fetch image: ${imageUrl}`);
18     }
19
20     const buffer = await response.arrayBuffer();
21     const bufferImage = Buffer.from(buffer);
22
23     const asset = await client.assets.upload('image', bufferImage, {
24       filename: imageUrl.split('/').pop(),
25     });
26
27     console.log(`Image uploaded successfully: ${asset._id}`);
28     return asset._id;
29   } catch (error) {
30     console.error(`Failed to upload image:`, imageUrl, error);
31     return null;
32   }
33 }
```

## scheme



## data fetching

