

Day 3: API Integration Report

Project Focus: Furniro (Furniture Marketplace)

API Integration Process

Overview

This process focused on integrating an external API that provides furniture-related data into a Sanity CMS project.

Steps Taken

1. Environment Setup

- Leveraged dotenv to securely manage environment variables from a .env file.
- The key environment variables included:
 - NEXT_PUBLIC_SANITY_PROJECT_ID
 - NEXT_PUBLIC_SANITY_DATASET
 - SANITY_TOKEN

2. Sanity Client Creation

- Established a connection to the Sanity project using @sanity/client.
- The client configuration required:
 - Project ID
 - Dataset
 - API version
 - Authentication token

3. Data Fetching

- Made concurrent API calls to retrieve furniture-related data, such as product listings and designer details.
- Specific endpoints were accessed to gather the necessary information.

4. Data Processing

- Structured the fetched furniture data into a format compatible with Sanity's schema.
- Uploaded product images and other assets to Sanity's asset library using an appropriate upload method.

5. Sanity Document Creation

- Transformed processed data into Sanity-compatible document structures, ensuring alignment with Furniro's CMS requirements.
- Successfully uploaded documents through `client.create()`.

Migration Steps and Tools Used

Migration Steps

1. Backup Data

- Use the Sanity CLI or dashboard to take a backup of the existing dataset.
- This ensures you can restore your data if issues arise during the migration.

2. Update Schema

- Make necessary changes to the schema, such as:
 - Adding new fields
 - Modifying existing ones
 - Updating validation rules

3. Deploy Schema Updates

- Deploy the updated schema to Sanity Studio to make the new structure available in your environment.

4. Validate Data

- Check data alignment with the updated schema, especially for required fields or validation rules.
- Assign default values to affected documents either manually or programmatically if new required fields are introduced.

5. Test Locally

- Run Sanity Studio locally to verify changes.
- Ensure that the updates work as expected without breaking functionality.

6. Migrate Data

- For large-scale migrations, used JavaScript with the Sanity Client to automate updates across multiple documents.

7. Deploy to Production

- Push changes to the production environment.
-

Tools Used

1. Sanity CLI

- For exporting and importing datasets and deploying schema updates.

2. Sanity Studio

- To update and test schema changes locally before deploying to production.

3. Sanity Dashboard

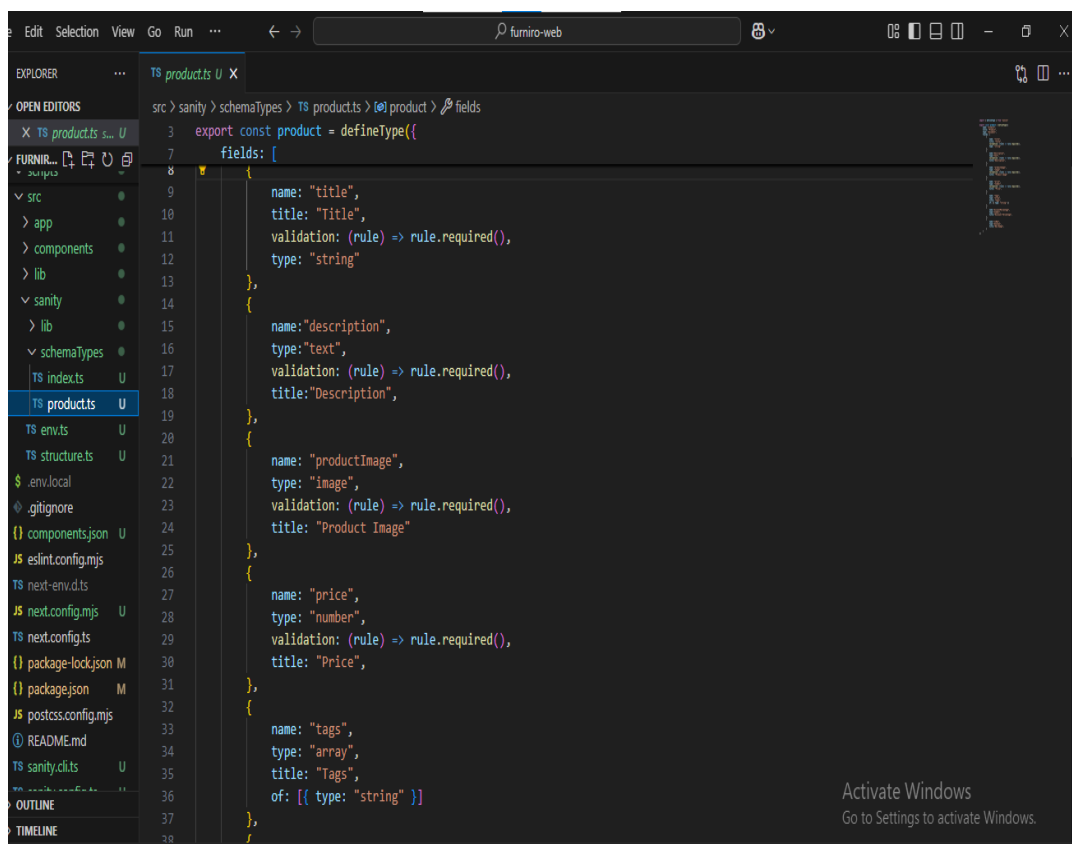
- To monitor and manage the live dataset and environment.

4. Sanity Client with JavaScript

- For automating updates to multiple documents during large-scale migrations.

BY ANOSHA HASSAN

SCHEMAS



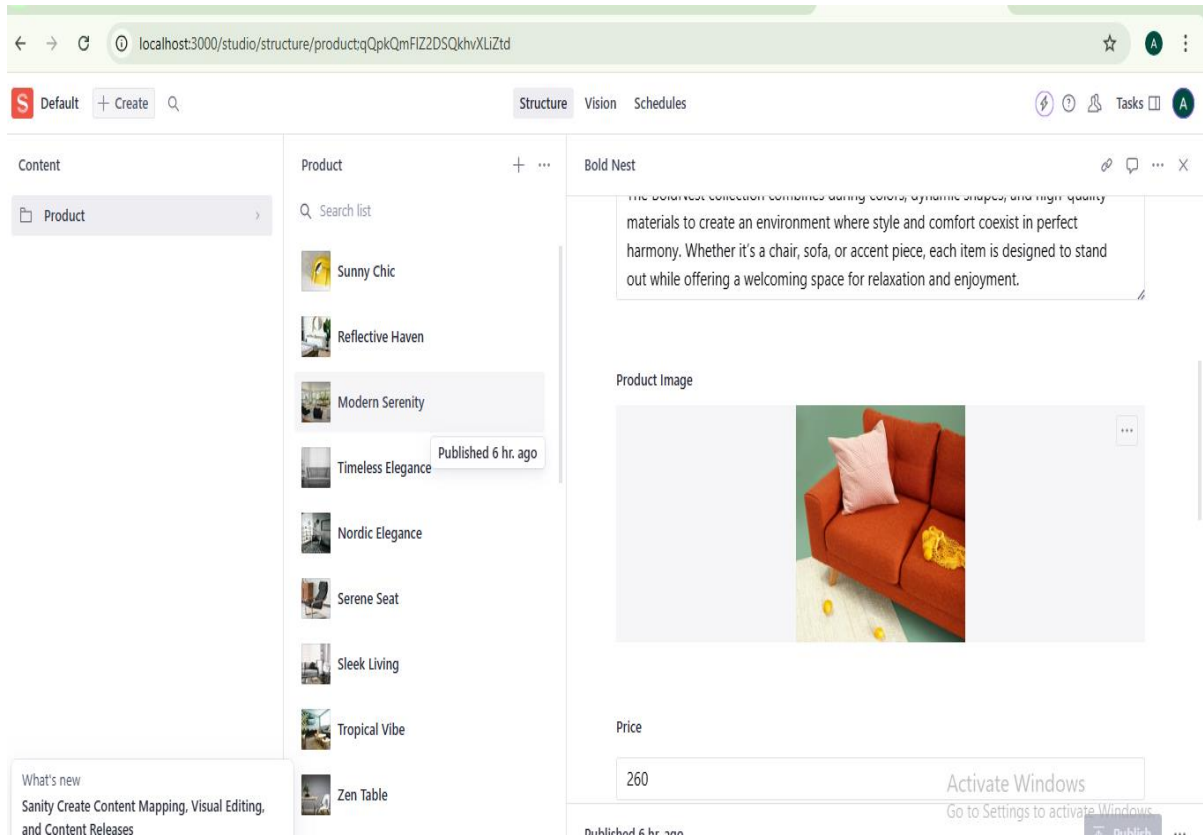
The screenshot shows a Visual Studio Code editor with a TypeScript file named `products.ts` open. The file is located in the `src > sanity > schemaTypes > products` directory. The code defines a `product` type with the following fields:

```
export const product = defineType({
  fields: [
    {
      name: "title",
      title: "Title",
      validation: (rule) => rule.required(),
      type: "string"
    },
    {
      name: "description",
      type: "text",
      validation: (rule) => rule.required(),
      title: "Description",
    },
    {
      name: "productImage",
      type: "image",
      validation: (rule) => rule.required(),
      title: "Product Image"
    },
    {
      name: "price",
      type: "number",
      validation: (rule) => rule.required(),
      title: "Price",
    },
    {
      name: "tags",
      type: "array",
      title: "Tags",
      of: [{ type: "string" }]
    }
  ]
})
```

The Explorer sidebar on the left shows the project structure, including `src`, `components`, `lib`, `sanity`, `schemaTypes`, and `products`. The `products` directory is selected, showing the `products.ts` file. The Timeline sidebar at the bottom is empty.

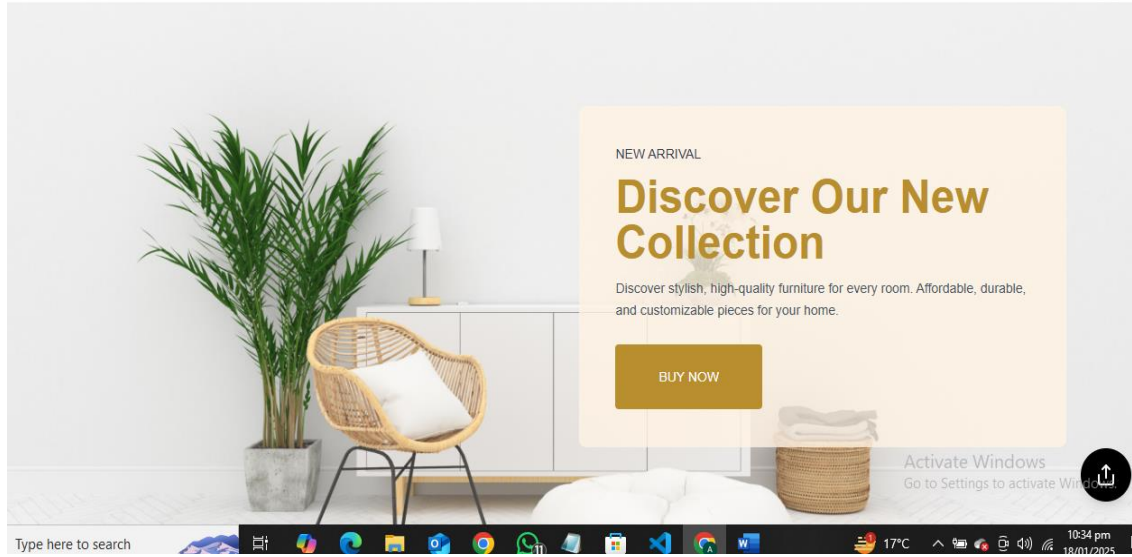
Populated sanity CMS fields:

Products :



```
Image with src "/r1.png" has legacy prop "objectFit". Did you forget to run the
codemod?
Read more: https://nextjs.org/docs/messages/next-image-upgrade-to-13
GET / 200 in 19056ms
  o Compiling /studio/[...tool] ...
GET / 200 in 2324ms
✓ Compiled /studio/[...tool] in 65.9s
GET /studio 200 in 80543ms
GET /studio 200 in 516ms
  o Compiling /favicon.ico ...
✓ Compiled /favicon.ico in 3s
GET /favicon.ico?favicon.45db1c09.ico 200 in 3370ms
GET /studio 200 in 558ms
GET /favicon.ico?favicon.45db1c09.ico 200 in 434ms
```

Ln 8, Col 10 Spaces: 4 UTF-8 CRLF TypeScript Go Liv



Type here to search



keen eye for detail, each piece in the Amber Haven collection is designed to evoke feelings of comfort and relaxation. The soft, amber-toned accents, paired with sleek lines and refined craftsmanship, bring a touch of nature's beauty into your home. Whether you're furnishing your living room, bedroom, or dining space, Amber Haven creates an inviting atmosphere that radiates peace and sophistication. The collection's warmth and versatility allow it to seamlessly integrate into various décor styles, from modern to traditional, making it a perfect choice for those who love both timeless elegance and contemporary flair. Amber Haven offers a sanctuary of comfort and luxury, where every piece is designed to enhance your home and your well-being. **Key Features:** Warm amber tones and elegant design create a cozy, inviting ambiance. High-quality craftsmanship and materials ensure durability and long-lasting appeal. Versatile style complements a variety of interior designs, from modern to traditional. Perfect for creating a serene and luxurious space in any room. Ideal for those seeking a combination of beauty, comfort, and sophistication. Bring the golden glow of Amber Haven into your home—where warmth, luxury, and timeless design come together to create a peaceful and elegant retreat.

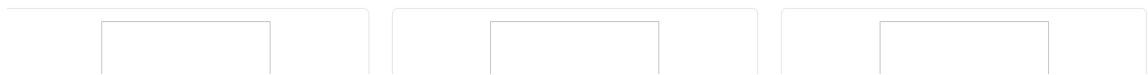
\$150

from sleek and modern to classic and traditional. The thoughtful mix of shapes, sizes, and finishes makes this set ideal for creating eye-catching displays in your home or office. Place them together for a coordinated look or spread them throughout different spaces for a cohesive, stylish touch. The Vase Set is not just about aesthetic appeal—it's also about quality and durability. Each piece is designed with precision, ensuring that the vases maintain their beauty for years to come. Whether you're gifting this set for a special occasion or adding it to your own home, it is sure to be a cherished piece that adds charm and character to any setting. **Key Features:** Includes multiple vases with varying shapes and sizes. Perfect for displaying flowers, plants, or as standalone decor. High-quality materials for long-lasting beauty and durability. Versatile design complements a wide range of interior styles. Ideal for both gifting and personal use. Enhance your space with the Vase Set—a blend of elegance, quality, and timeless design that transforms any room into a haven of beauty.

\$150

enhances your living space. From its sleek lines to its unique color palette, every aspect of RetroVibe is carefully crafted to evoke the spirit of retro design while seamlessly fitting into today's modern interiors. Whether you're decorating a living room, bedroom, or workspace, RetroVibe serves as a conversation starter and a statement piece. Its versatility allows it to complement a variety of decor styles, from mid-century modern to eclectic and contemporary. The perfect way to add a touch of nostalgia and style, RetroVibe is ideal for those who love vintage aesthetics with a modern twist. **Key Features:** Retro-inspired design with modern touches. Bold color palette and classic materials that evoke nostalgia. High-quality craftsmanship for durability and lasting appeal. Versatile style that complements various home decor themes. Ideal for creating a fun, stylish, and unique atmosphere. Bring back the charm of the past with RetroVibe—where classic design meets contemporary flair. Perfect for those who love a dash of nostalgia in their modern home.

\$340



API MIGRATION SCRIPTS

```
scripts > JS importData.mjs > uploadImageToSanity
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  }
34
35  async function uploadProduct(product) {
36    try {
37      const imageId = await uploadImageToSanity(product.imageUrl);
38
39      if (imageId) {

```

Activate Windows
Go to Settings to activate

BY ANOSHA HASSAN