Day 3 - API Integration Report - Furniva

BY WANIA AZAM

• *API Integration Process*

Step 1: Understanding Requirements

- First, I studied what was needed for the marketplace.
- Found out that we needed more fields in the product schema to match new ideas and user needs.

Step 2: Adding Fields to Schema

- **Updated the Sanity CMS schema to include extra fields like:**
    - **Code**: Unique product code.
    - **Inventory**: Number of products in stock.
    - **Tags**: Keywords to help search.
    - **Discount Percentage:** To manage sales or discounts.

Step 3: Manually Importing Data

- I didn't use scripts. Instead, I added the data one by one into Sanity CMS. This helped me double-check everything for accuracy.
- Used Sanity Studio's interface to create new product entries, fill out all fields, and upload images.

Step 4: Validating Data

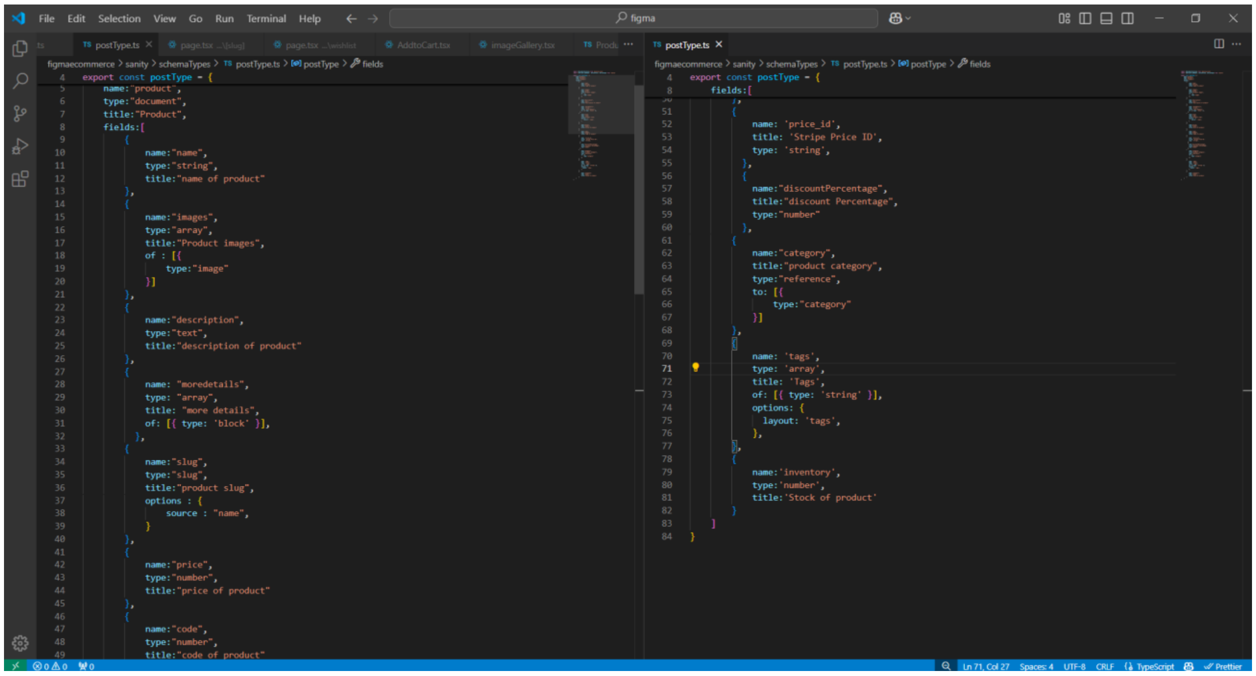
- After adding the data, I checked each product to make sure the fields were correct.
- Used Sanity's GROQ query tool to test and confirm the data was saved properly.

• *Adjustments made to schemas:*

I added several new fields to the product schema:

- **Code**: A unique product code for identification.
- **Inventory**: Tracks the stock count of each product.
- **Tags**: Helps categorize products and improves search optimization.
- **Discount Percentage**: For managing product discounts or sales.
- **Images**: Multiple images to showcase the product from different angles

Here image of my Updated schema:


File Edit Selection View Go Run Terminal Help
figma
postType.ts
page.tsx ...\[slug]
page.tsx ...\wishlist
AddtoCart.tsx
imageGallery.tsx
figmaecommerce > sanity > schemaTypes > postType.ts > postType > fields
Ln 71, Col 27 Spaces: 4 UTF-8 CRLF TypeScript Prettier

```
export const postType = {
    name:"product",
    type:"document",
    title:"Product",
    fields:[
        {
            name:"name",
            type:"string",
            title:"name of product"
        },
        {
            name:"images",
            type:"array",
            title:"Product images",
            of : [{
                type:"image"
            }]
        },
        {
            name:"description",
            type:"text",
            title:"description of product"
        },
        {
            name: "moredetails",
            type: "array",
            title: "more details",
            of: [{ type: 'block' }],
          },
        {
            name:"slug",
            type:"slug",
            title:"product slug",
            options : {
                source : "name",
            }
        },
        {
            name:"price",
            type:"number",
            title:"price of product"
        },
        {
            name:"code",
            type:"number",
            title:"code of product"
```

```
export const postType = {
    fields:[
        {
            name: 'price_id',
            title: 'Stripe Price ID',
            type: 'string',
          },
          {
            name:"discountPercentage",
            title:"discount Percentage",
            type:"number"
          },
        {
            name:"category",
            title:"product category",
            type:"reference",
            to: [{
                type:"category"
            }]
        },
        {
            name: 'tags',
            type: 'array',
            title: 'Tags',
            of: [{ type: 'string' }],
            options: {
              layout: 'tags',
            },
        },
        {
            name:'inventory',
            type:'number',
            title:'Stock of product'
        }
    ]
}
```

Manual Data Entry Steps:

Tools I Used:

- **Sanity Studio:** For entering data.

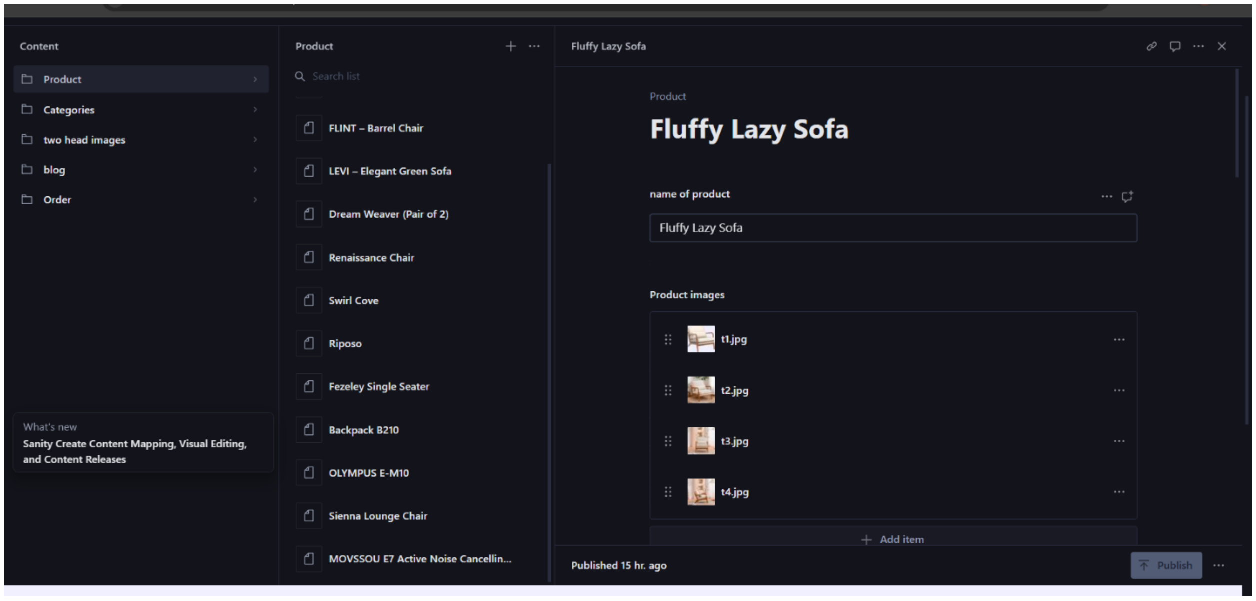
Steps I Followed:

1. **Prepare Data:**
    - I collected product information Amazon and other Ecommerce Website.
    - Organized the information so it was easy to add.
2. **Add to Sanity:**
    - Went into Sanity Studio.
    - Created new entries for each product.
    - Filled in fields like name, price, code, tags, etc.
    - Uploaded images for each product.
3. **Check Entries:**
    - Made sure all fields were filled correctly.
    - Run queries in Sanity to test the data.

Screenshots of Working process:

Sanity CMS Data Entry:

- ***Screenshot: Product entry with all fields filled out.***

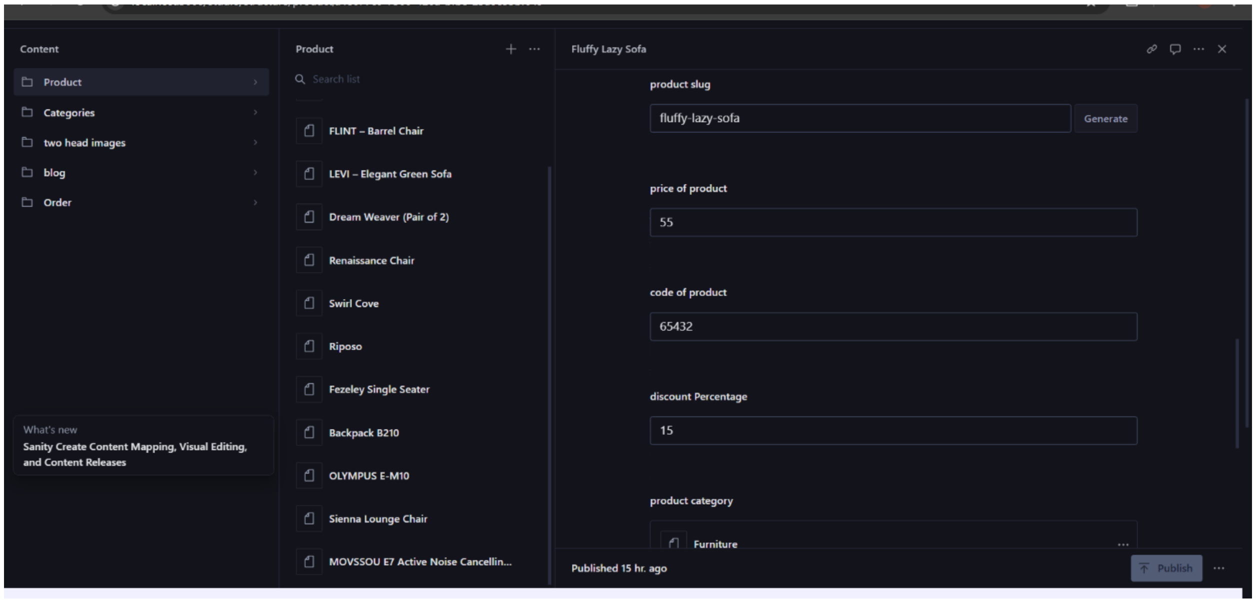


Product

Fluffy Lazy Sofa

name of product: Fluffy Lazy Sofa

Product images: t1.jpg, t2.jpg, t3.jpg, t4.jpg



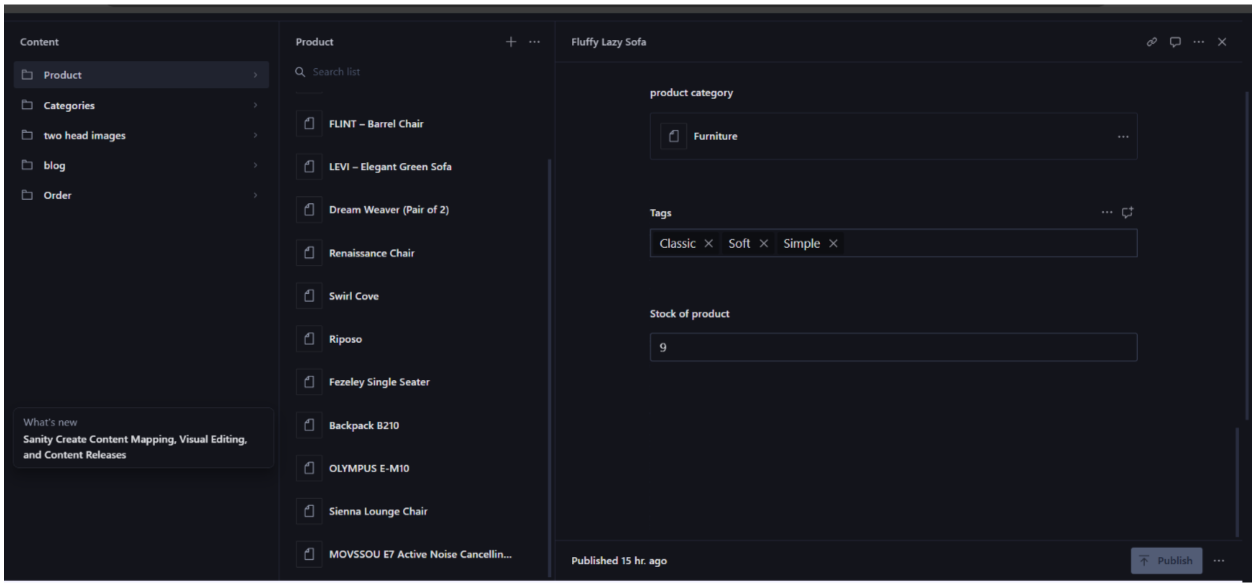
product slug: fluffy-lazy-sofa

price of product: 55

code of product: 65432

discount Percentage: 15

product category: Furniture



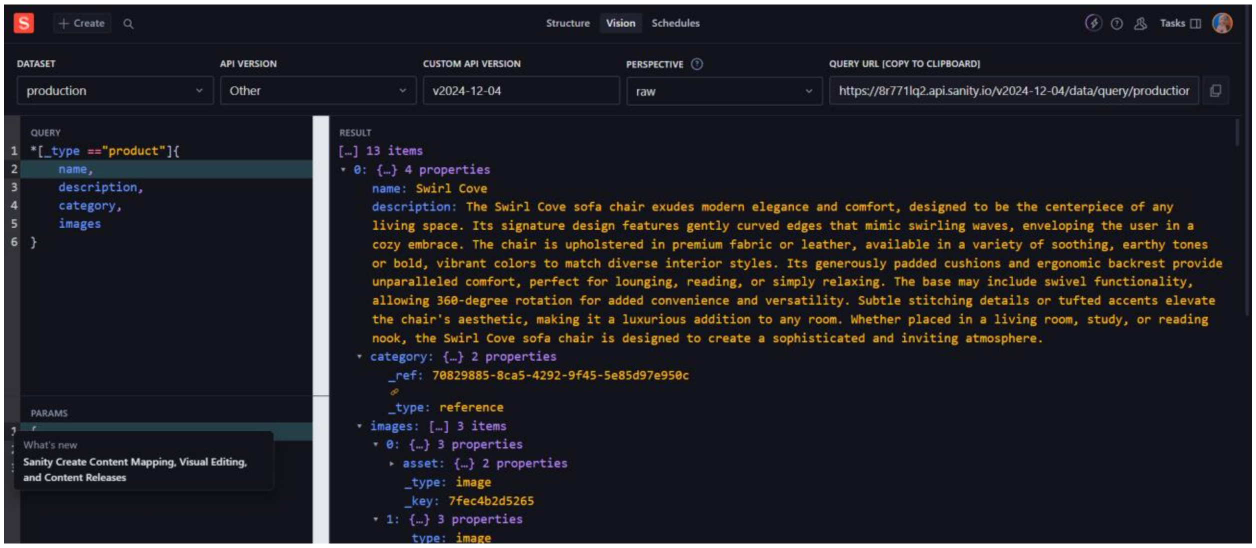
product category: Furniture

Tags: Classic, Soft, Simple

Stock of product: 9

Data validation:

- ***Screenshot : GROQ query showing correct data.***



Create
Structure
Vision
Schedules
Tasks
What's new
Sanity Create Content Mapping, Visual Editing, and Content Releases

DATASET: production
API VERSION: Other
CUSTOM API VERSION: v2024-12-04
PERSPECTIVE: raw
QUERY URL [COPY TO CLIPBOARD]: https://8r771lq2.api.sanity.io/v2024-12-04/data/query/productior

QUERY

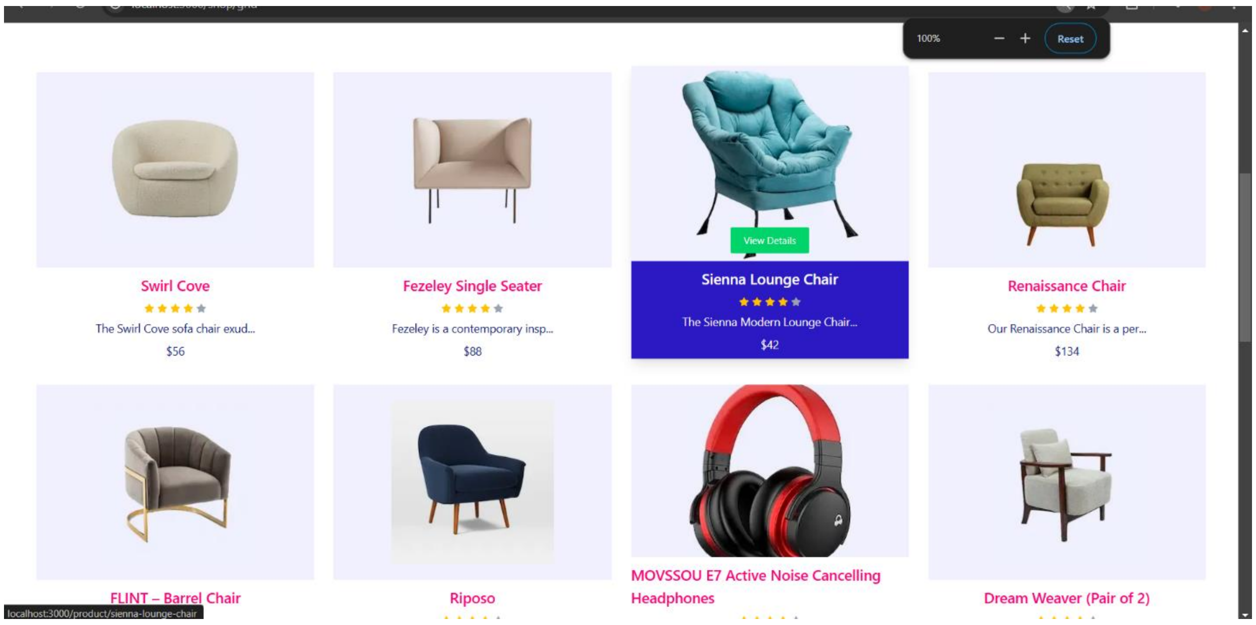
```
*[_type =="product"]{
    name,
    description,
    category,
    images
}
```

PARAMS

RESULT

```
[…] 13 items
▾ 0: {…} 4 properties
    name: Swirl Cove
    description: The Swirl Cove sofa chair exudes modern elegance and comfort, designed to be the centerpiece of any
    living space. Its signature design features gently curved edges that mimic swirling waves, enveloping the user in a
    cozy embrace. The chair is upholstered in premium fabric or leather, available in a variety of soothing, earthy tones
    or bold, vibrant colors to match diverse interior styles. Its generously padded cushions and ergonomic backrest provide
    unparalleled comfort, perfect for lounging, reading, or simply relaxing. The base may include swivel functionality,
    allowing 360-degree rotation for added convenience and versatility. Subtle stitching details or tufted accents elevate
    the chair's aesthetic, making it a luxurious addition to any room. Whether placed in a living room, study, or reading
    nook, the Swirl Cove sofa chair is designed to create a sophisticated and inviting atmosphere.
  ▾ category: {…} 2 properties
      _ref: 70829885-8ca5-4292-9f45-5e85d97e950c
      _type: reference
  ▾ images: […] 3 items
    ▾ 0: {…} 3 properties
      ▸ asset: {…} 2 properties
        _type: image
        _key: 7fec4b2d5265
    ▾ 1: {…} 3 properties
        _type: image
```

- ***Screenshot : Frontend display of the products.***


100%
Reset
Swirl Cove
The Swirl Cove sofa chair exud...
$56
Fezeley Single Seater
Fezeley is a contemporary insp...
$88
View Details
Sienna Lounge Chair
The Sienna Modern Lounge Chair...
$42
Renaissance Chair
Our Renaissance Chair is a per...
$134
FLINT – Barrel Chair
Riposo
MOVSSOU E7 Active Noise Cancelling Headphones
Dream Weaver (Pair of 2)
localhost:3000/product/sienna-lounge-chair

Conclusion:

Throughout this project, I have not only updated the product schema to include new fields like Code, Inventory, Tags, Images, Discount Percentage, More Details, and Slug, but I also gained hands-on experience with API migration. This included using migration scripts to transfer data efficiently and ensuring that the data matches the schema. By validating and adjusting the schema, I learned how to align it with the API structure, ensuring compatibility and smooth data migration.

In addition, I learned how to integrate APIs in a Next.js project, testing them with tools like Postman to ensure that product listings, categories, and other relevant data were displayed correctly. The overall process also improved my skills in error handling, creating utility functions, and testing data integration to maintain a high-quality user experience.

In short, I learned how to:

- Migrate data using API integration and manual imports.
- Adjust and validate schemas for API compatibility.

- Handle API errors effectively for smooth user experiences.
- Integrate data seamlessly in Next.js, ensuring functional frontend displays.

This project was a great opportunity to enhance my understanding of working with APIs, data migration, and frontend integration using Next.js.