

DAY 3 API INTEGRATION (Furniture)

1. Introduction

This report documents the integration process of migration external data from a REST API into the Sainty CMS for “Hekto”, a furniture e-commerce platform .This goal is to automate content management for products such as sofas,chair,office desk, and other furniture categories,ensuring seamless backend update and synchronization with the next.js frontend.

Key objective include:

- Fetching product and category data (furniture-related)from an external API.
- Structuring and migrating this data to Sanity CMS.
- Displaying the migrated data dynamically on the frontend.

2. API Integration Overview

▪**API Name:**The external marketplace API,hosted at:
[http:// next-ecommerce-template-8.vercel.app/api/product](http://next-ecommerce-template-8.vercel.app/api/product).

▪Data Fetched:

- **Products:** Details such as products title,prices,descriptive,categories(e.g., sofas ,chairs), images,etc.
- **Categories:** Furniture categories like sofas,etc associated with products.
- **Sainty Client:**Used to manage and store fetched data in the CMS ensuring compatibility with the defined schema structure.

3. Schema structure changes:

The products schema was enhanced to better represent furniture product details, Key updates include:

Updated Fields:

the furniture products's name(required)

.name(string):the furniture products's name(required)

.image(image) products price(required)

.description(text):A brief product description(max150 characters).

.discountPercentage(number):the discount percentage (0 to 100).

.isFeaturedProducts(boolean):flag indicating if the product is featured.

.stocklevel (number):track stock levels()must be positive).

.category(reference):links products to specific furniture categories.

Snipped of Updated product Schema:

```
32         name: image ,
33         title:"Image",
34         type:"image",
35         options:{
36             hotspot:true,
37         },
38     }},
39     defineField({
40         name:"category",
41         title:"Category",
42         type:"array",
43         of:[{type:"reference", to: {type:"category"} }],
44         validation:(rule) => rule.required(),
45     }),
46     defineField({
47         name:"price",
48         title:"Price",
49         type:"number",
50         validation:(rule) => rule.required(),
51     }),
52     defineField({
53         name:"rowprice",
54         title:"Row Title",
55         type:"number",
56     }),
57     defineField({
58         name:"ratings",
59         title:"Rating",
```

.4 Migration Step and Tools Used

Tools used :

1. **Sanity Client:** To interact with the sanity CMS(via @sainty/client).
2. **Node.js &Fetch API:** For fetching data from the API and uploading it to Sanity.
3. **Environment Variables:** For secure storage of sensitive data like API tokens.
4. **Sanity Image Tool:**To upload and manage furniture images in the CMS.

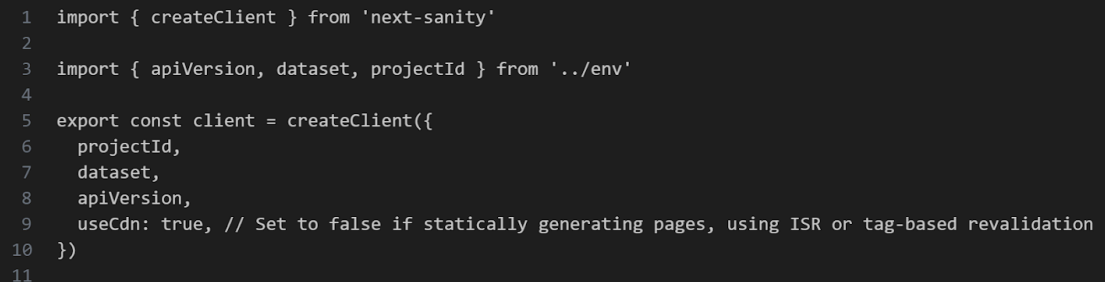
Migration Steps:

. 1 Set up Environment :

- .installed required dependencies:@sanity/client,dotenv.
- .Configured.env.local for secure storage of API keya.

. 2 Sanity Client Setup:

- . Create a sanityClient instance using project ID, dataset, and API token.



```
1 import { createClient } from 'next-sanity'
2
3 import { apiVersion, dataset, projectId } from '../env'
4
5 export const client = createClient({
6   projectId,
7   dataset,
8   apiVersion,
9   useCdn: true, // Set to false if statically generating pages, using ISR or tag-based revalidation
10 })
11
```

.3 Fetch Data from API:

.Used fetch to retrieve furniture products and category data from the API.

.4 Upload Images to Sanity:

. Uploading furniture images using the sanity assets pipeline and retrieved assets IDs for linking.



```
1  Function to upload an image to Sanity
2  async function uploadImageToSanity(imageUrl) {
3    try {
4      // Fetch the image from the provided URL
5      const response = await fetch(imageUrl);
6      if (!response.ok) throw new Error(`Failed to fetch image: ${imageUrl}`);
7
8      // Convert the image to a buffer (binary format)
9      const buffer = await response.arrayBuffer();
10
11     // Upload the image to Sanity and get its asset ID
12     const uploadedAsset = await targetClient.assets.upload("image", Buffer.from(buffer), {
13       filename: imageUrl.split("/").pop(), // Use the file name from the URL
14     });
15
16     return uploadedAsset._id; // Return the asset ID
17   } catch (error) {
18     console.error("Error uploading image:", error.message);
19     return null; // Return null if the upload fails
20   }
21 }
22
```

(Screen shot)

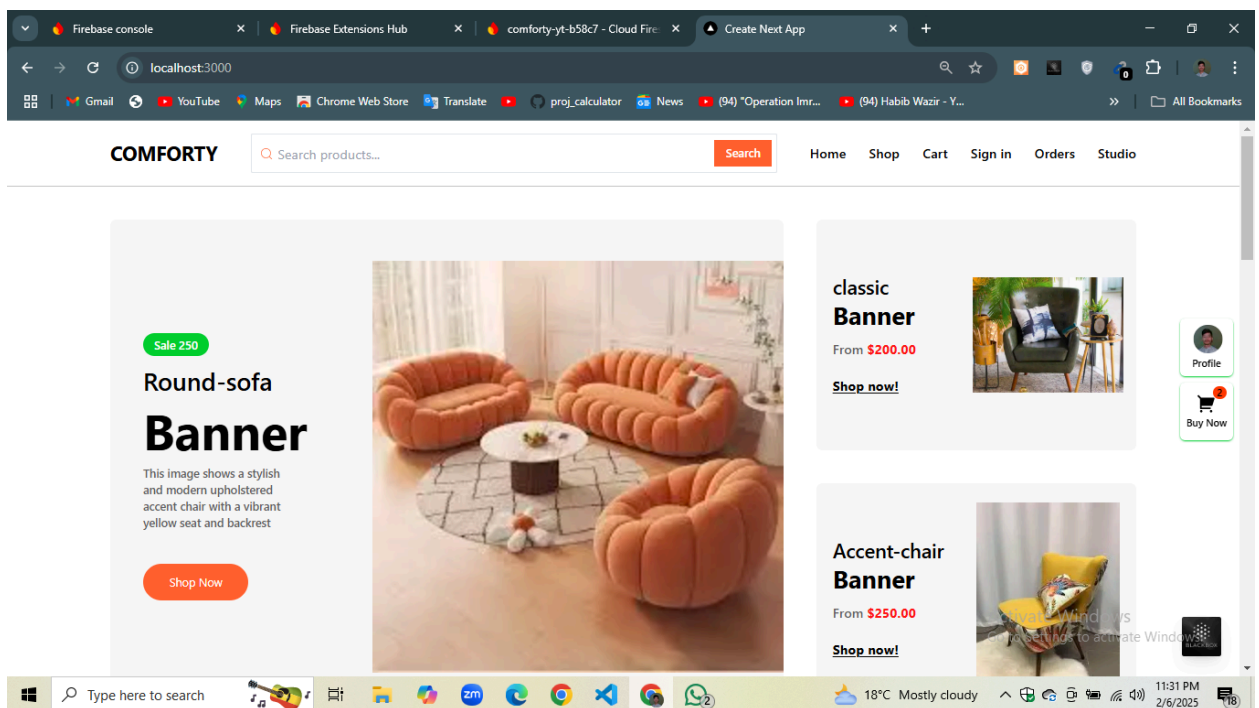
```
1 import { ProductData } from "@type";
2 import { NextRequest, NextResponse } from "next/server"
3 import Stripe from 'stripe'
4 export const POST = async(request:NextRequest)=>{
5     const stripe = new Stripe(process.env.STRIPE_SECRET_KEY as string);
6
7     try{
8         const reqBody = await request.json()
9         // console.log(reqBody);
10        const {items,email} = await reqBody;
11
12        const extractingItems = await items.map((item: ProductData) => ({
13            quantity: item?.quantity,
14            price_data: {
15                currency: "usd",
16                unit_amount: Math.round(item.price * 100),
17                product_data: {
18                    name: item?.title,
19                    description: item?.description,
20
21                },
22            },
23        }));
24
25        const origin = request.headers.get("origin")
26        // console.log(origin)
27        const session = await stripe.checkout.sessions.create({
28            payment_method_types:["card"],
29            line_items:extractingItems,
30            mode:'payment',
31            success_url:`${origin}/success/?session_id={CHECKOUT_SESSION_ID}`,
32            cancel_url:`${origin}/cancel/?canceled=true`,
33            metadata:{
34                email,
35            }
36        })
37
38        return NextResponse.json({url:session?.url},{status:200})
39    }catch(error){
40        return NextResponse.json({error:error},{status:500})
41    }
42 }
```

API call in response

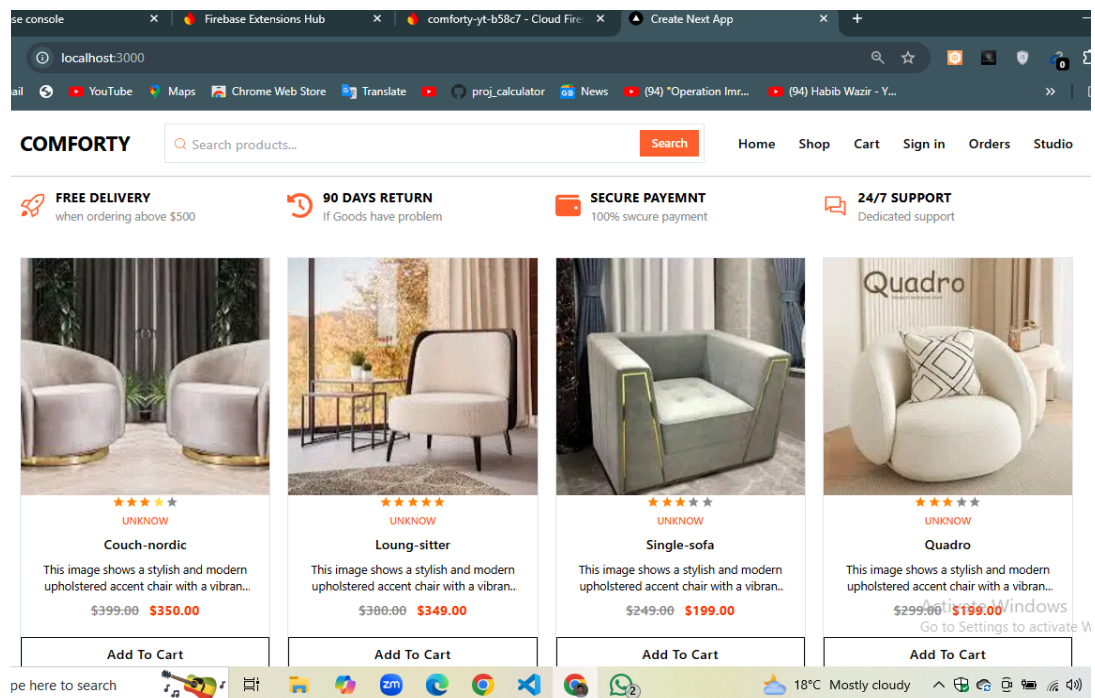
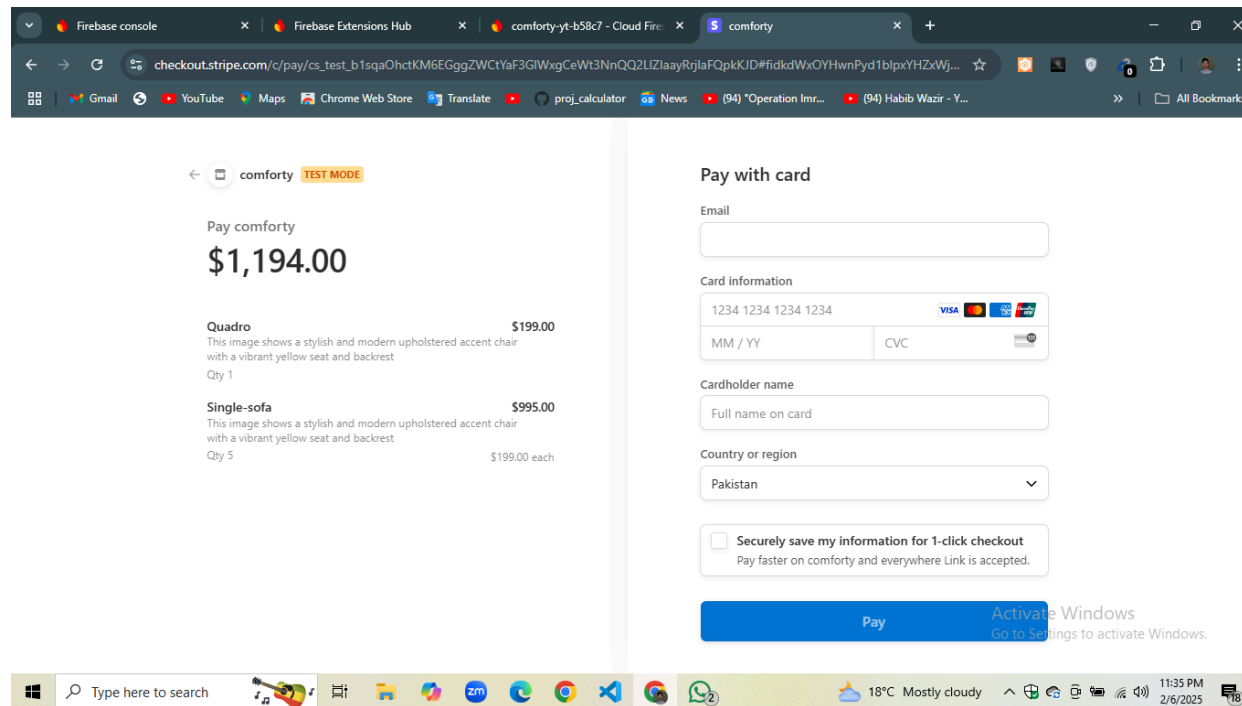
Day 4 Dynamic Routing

.1 Components to build :

*product listing Component:



- Product Details Component :



● Checkout Flow Component

Firestore console

Firestore Extensions Hub

comforty-yt-b58c7 - Cloud Fire...

Create Next App



localhost3000/cart

COMFORTY

Search products...

Search

HomeShopCartSign inOrdersStudio

Product	Price	Quantity	Sub Total
<div><div><div>X</div><div></div></div><div>Quadro</div></div>	\$199.00	<div>-</div> 1 <div>+</div>	\$199.00
<div><div><div>X</div><div></div></div><div>Single-sofa</div></div>	\$199.00	<div>-</div> 5 <div>+</div>	\$995.00

RESET CART

Profile

Buy Now

Cart totals

Subtotal\$1,194.00

Shipping Charge\$0.00

Type here to search

18°C Mostly cloudy

11:35 PM 2/6/2025

Firestore console

Firestore Extensions Hub

comforty-yt-b58c7 - Cloud Fire...

comforty

checkout.stripe.com/c/pay/cs_test_b1sqOhctKM6EGggZWctYaf3GIWxgCeWt3NnQQ2LIZlaayRrjaFQpkKID#fidkdWxOYHwnPyd1blpxYHZxWj...

comforty TEST MODE

Pay comforty

\$1,194.00

Quadro

This image shows a stylish and modern upholstered accent chair with a vibrant yellow seat and backrest

\$199.00

Qty 1

Single-sofa

This image shows a stylish and modern upholstered accent chair with a vibrant yellow seat and backrest

\$995.00

Qty 5

\$199.00 each

Pay with card

Email

Card information

1234 1234 1234 1234

MM / YY

CVC

Cardholder name

Full name on card

Country or region

Pakistan

☐ Securely save my information for 1-click checkout

Pay faster on comforty and everywhere Link is accepted.

Pay