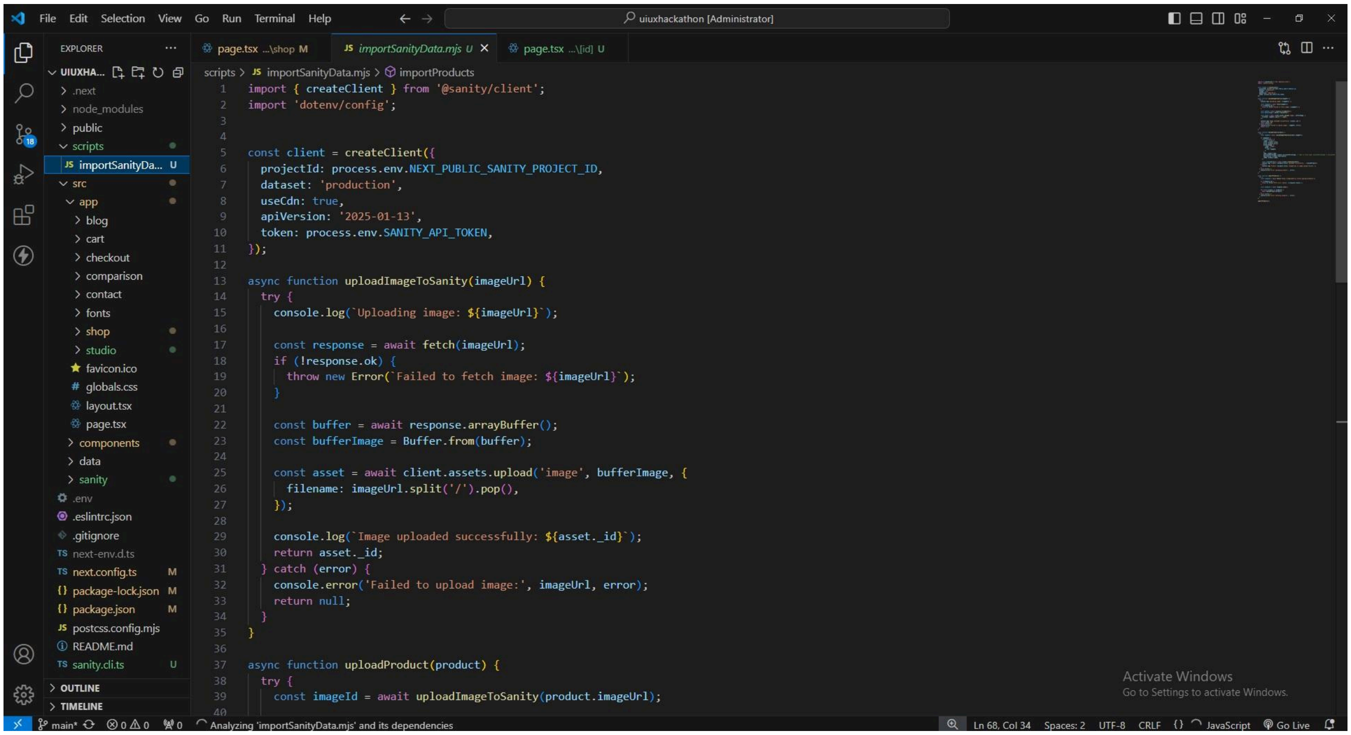
Day 3 - API Integration Report - General E-commerce

The main task for Day 3 was to integrate data from an external API into our platform and display it in the frontend, while storing it in Sanity CMS for easy content management. This was a crucial step in automating content population and ensuring the data was consistently available for display.

1. API Integration Process :

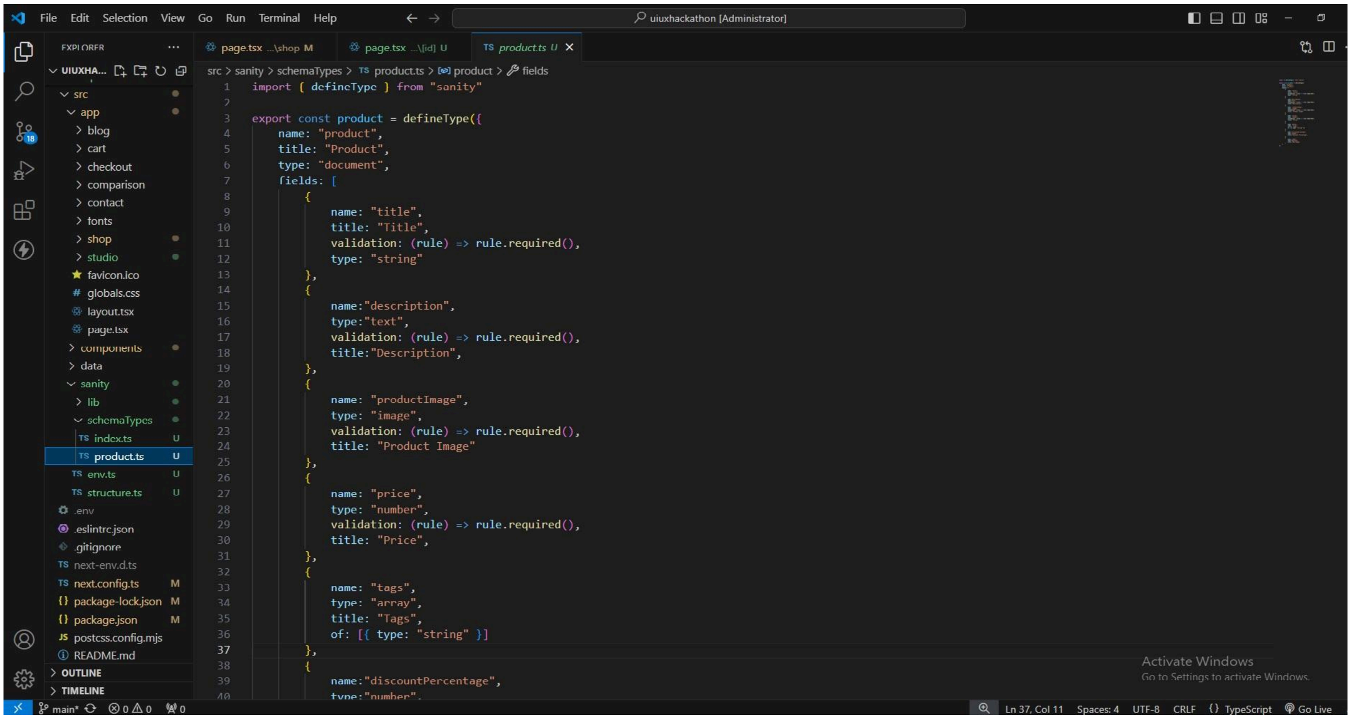
- **API Confirmation :** Reviewed the API provided in the designated template to check and verify the incoming data.
- **Data Migration :** Used the custom script designed to populate data coming from API to Sanity CMS.



```javascript
import { createClient } from '@sanity/client';
import 'dotenv/config';


const client = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset: 'production',
  useCdn: true,
  apiVersion: '2025-01-13',
  token: process.env.SANITY_API_TOKEN,
});

async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading image: ${imageUrl}`);

    const response = await fetch(imageUrl);
    if (!response.ok) {
      throw new Error(`Failed to fetch image: ${imageUrl}`);
    }

    const buffer = await response.arrayBuffer();
    const bufferImage = Buffer.from(buffer);

    const asset = await client.assets.upload('image', bufferImage, {
      filename: imageUrl.split('/').pop(),
    });

    console.log(`Image uploaded successfully: ${asset._id}`);
    return asset._id;
  } catch (error) {
    console.error('Failed to upload image:', imageUrl, error);
    return null;
  }
}

async function uploadProduct(product) {
  try {
    const imageId = await uploadImageToSanity(product.imageUrl);
```

2. Adjustments Made to Schema :

- Schema was adjusted to match the data coming from API.
- Ensured the data coming from API is compatible with schema.



```
import { defineType } from "sanity"

export const product = defineType({
    name: "product",
    title: "Product",
    type: "document",
    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" }]
        },
        {
            name:"discountPercentage",
            type:"number",
```

3. Migration Steps and Tools Used :

I created optimized GROQ queries to pull the newly migrated data from Sanity CMS. These queries were carefully crafted to ensure we only fetched the necessary fields, making the data retrieval process more efficient Once the data was fetched, I integrated it into the Next.js frontend.

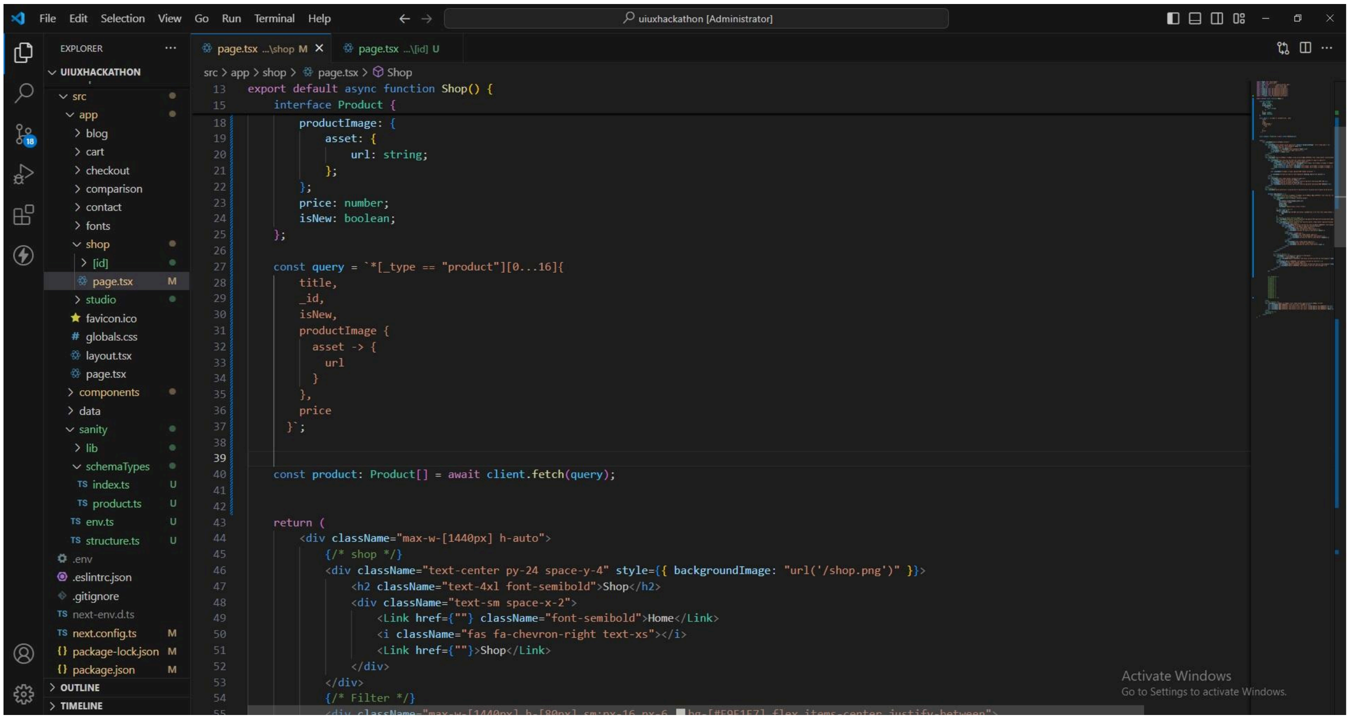
Tools Utilized:

- Sanity CLI.
- Groq Queries.
- Custom script for data migration.

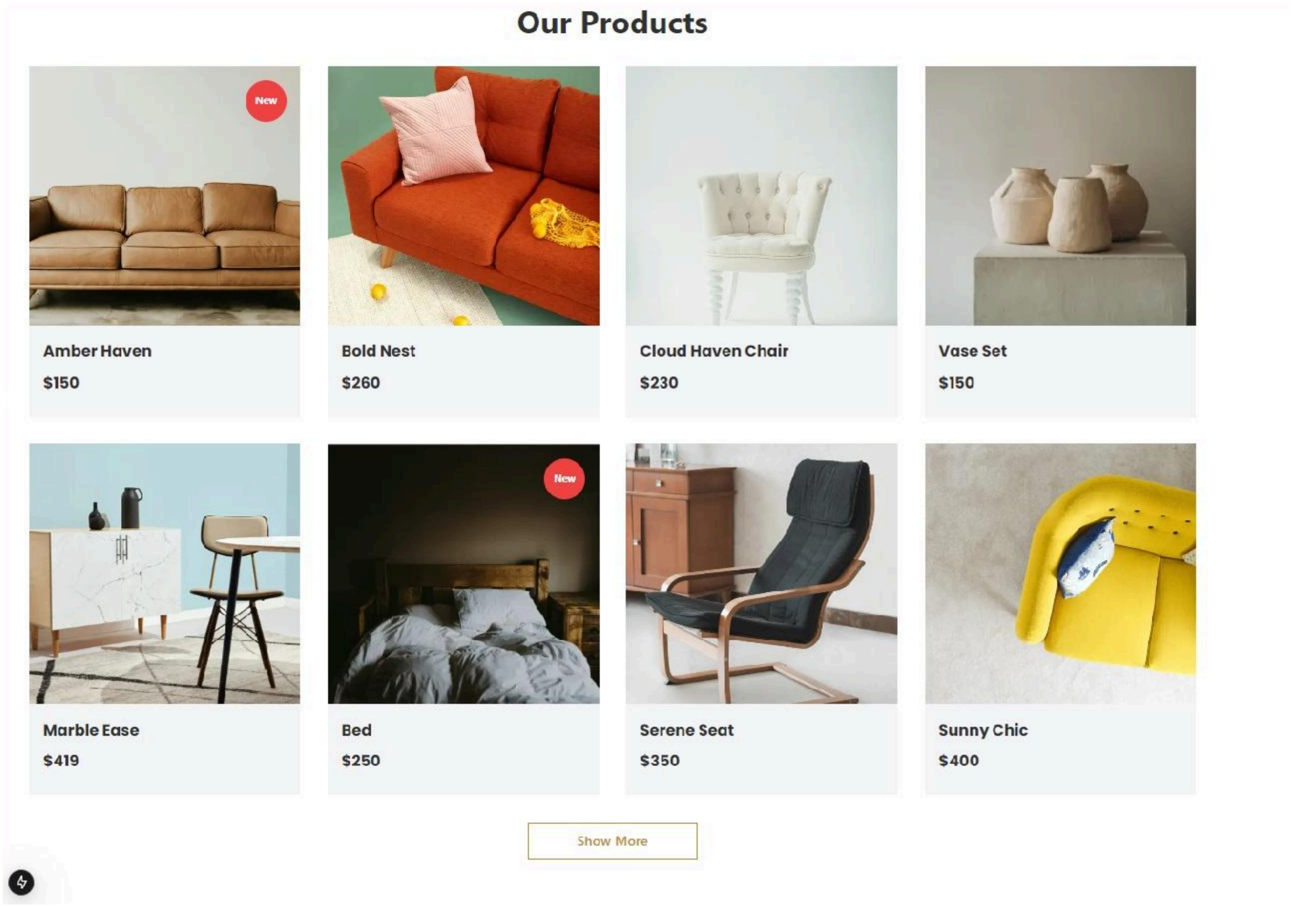
Migration Process:

- Passed project Id and Token to custom script.

- Ran the command “ npm run -[Key name]- ” to transfer data from API to Sanity CMS.
- Ensured the data before rendering it into front-end.



```
export default async function Shop() {
    interface Product {
        productImage: {
            asset: {
                url: string;
            };
        };
        price: number;
        isNew: boolean;
    };

    const query = `*[_type == "product"][0...16]{
        title,
        _id,
        isNew,
        productImage {
          asset -> {
            url
          }
        },
        price
      }`;


    const product: Product[] = await client.fetch(query);


    return (
        <div className="max-w-[1440px] h-auto">
            {/* shop */}
            <div className="text-center py-24 space-y-4" style={{ backgroundImage: "url('/shop.png')" }}>
                <h2 className="text-4xl font-semibold">Shop</h2>
                <div className="text-sm space-x-2">
                    <Link href={""} className="font-semibold">Home</Link>
                    <i className="fas fa-chevron-right text-xs"></i>
                    <Link href={""}>Shop</Link>
                </div>
            </div>
            {/* Filter */}
            <div className="max-w-[1440px] h-[80px] sm:px-16 px-6 bg-[#F9F1E7] flex items-center justify-between">
```



Our Products

Amber Haven
$150

Bold Nest
$260

Cloud Haven Chair
$230

Vase Set
$150

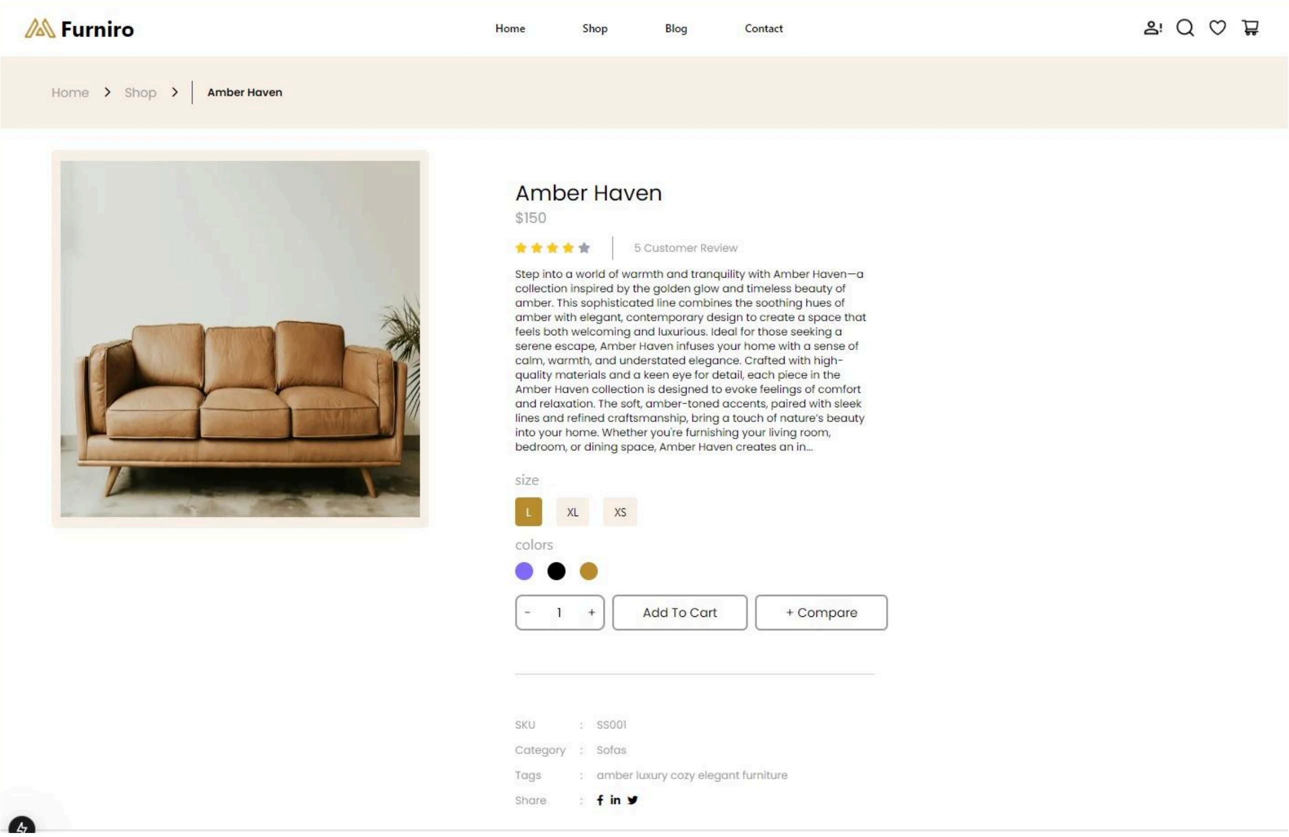
Marble Ease
$419

Bed
$250

Serene Seat
$350

Sunny Chic
$400

Screenshots :



Amber Haven

$150

★★★★★ | 5 Customer Review

Step into a world of warmth and tranquility with Amber Haven—a collection inspired by the golden glow and timeless beauty of amber. This sophisticated line combines the soothing hues of amber with elegant, contemporary design to create a space that feels both welcoming and luxurious. Ideal for those seeking a serene escape, Amber Haven infuses your home with a sense of calm, warmth, and understated elegance. Crafted with high-quality materials and a 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 in...

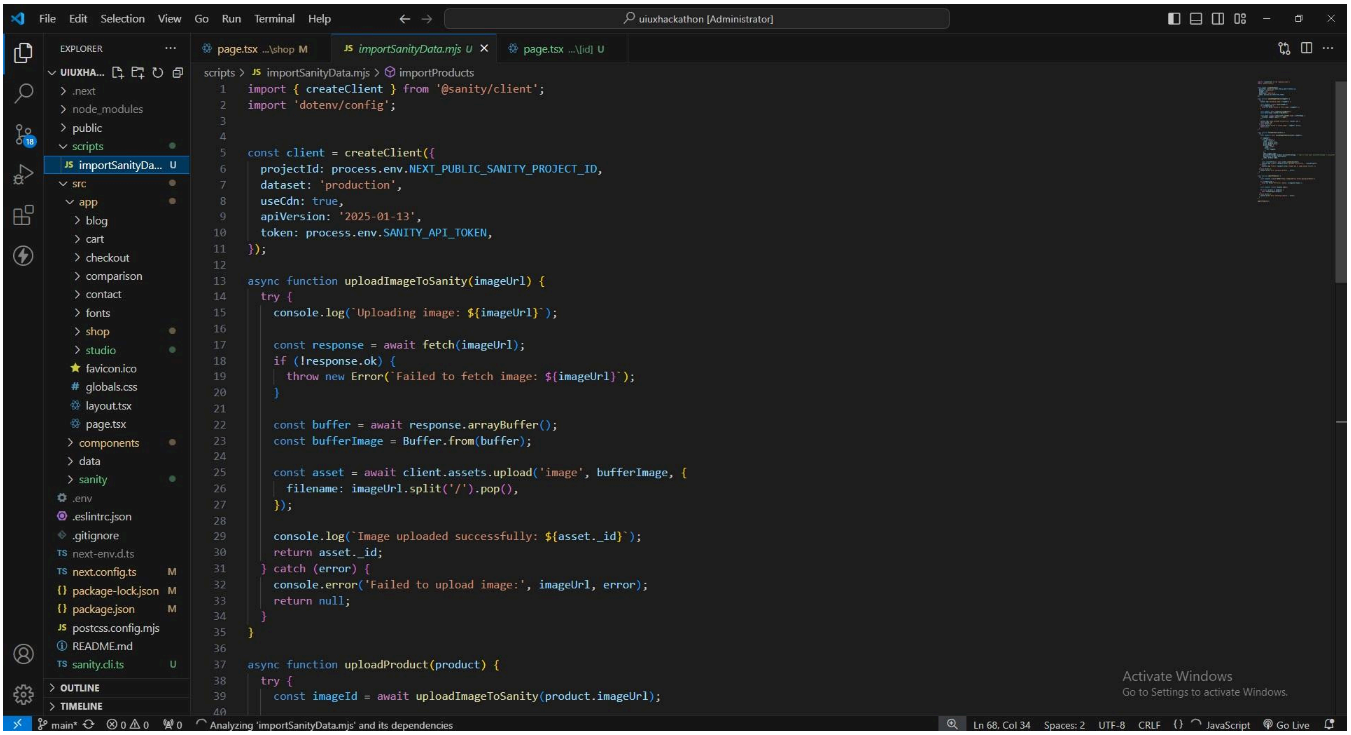
size

L XL XS

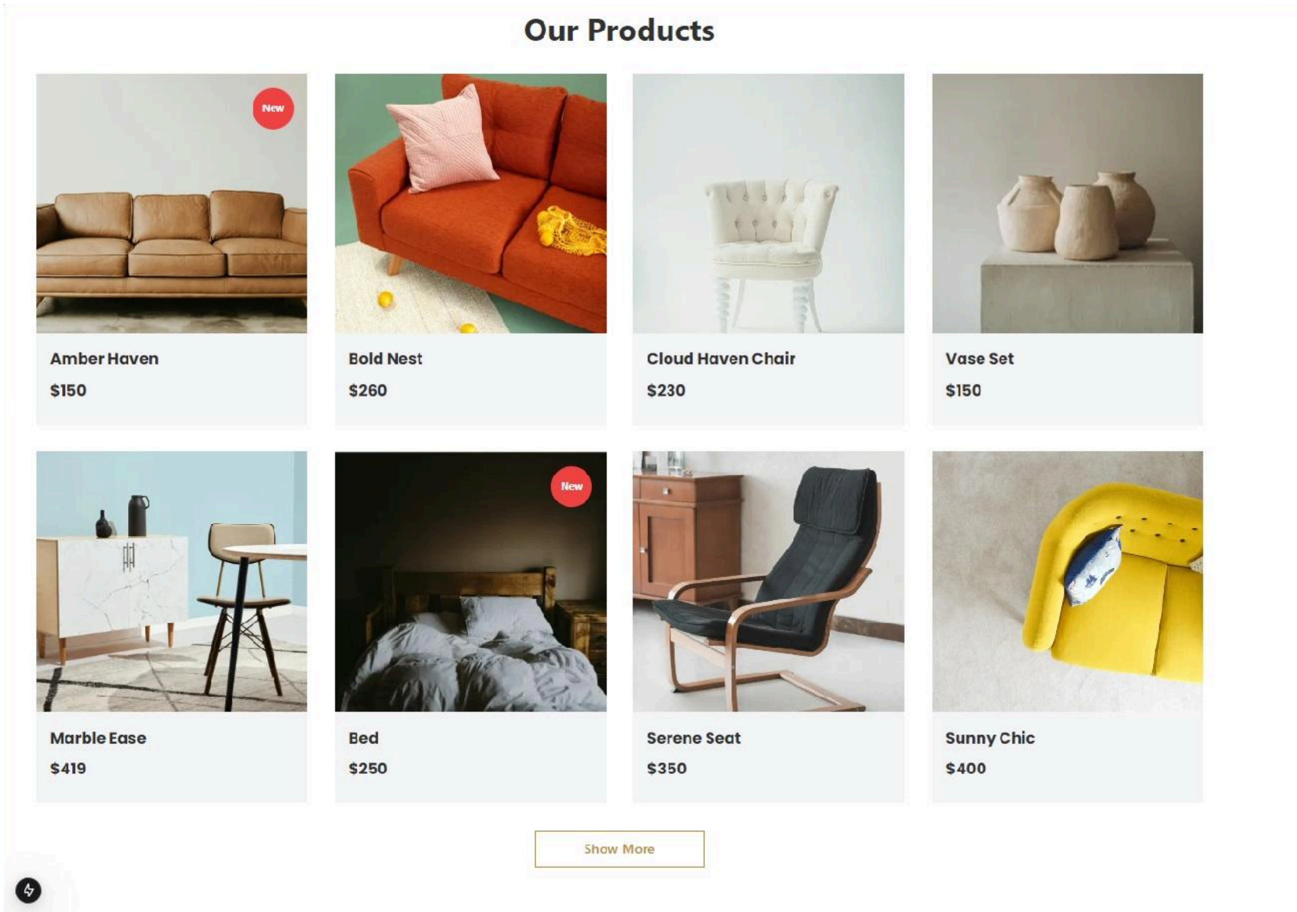
colors

● ● ●

SKU	:	SS001
Category	:	Sofas
Tags	:	amber luxury cozy elegant furniture
Share	:	f in



```javascript
import { createClient } from '@sanity/client';
import 'dotenv/config';


const client = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset: 'production',
  useCdn: true,
  apiVersion: '2025-01-13',
  token: process.env.SANITY_API_TOKEN,
});

async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading image: ${imageUrl}`);

    const response = await fetch(imageUrl);
    if (!response.ok) {
      throw new Error(`Failed to fetch image: ${imageUrl}`);
    }

    const buffer = await response.arrayBuffer();
    const bufferImage = Buffer.from(buffer);

    const asset = await client.assets.upload('image', bufferImage, {
      filename: imageUrl.split('/').pop(),
    });

    console.log(`Image uploaded successfully: ${asset._id}`);
    return asset._id;
  } catch (error) {
    console.error('Failed to upload image:', imageUrl, error);
    return null;
  }
}

async function uploadProduct(product) {
  try {
    const imageId = await uploadImageToSanity(product.imageUrl);
```



Our Products

New

Amber Haven

$150

Bold Nest

$260

Cloud Haven Chair

$230

Vase Set

$150

Marble Ease

$419

New

Bed

$250

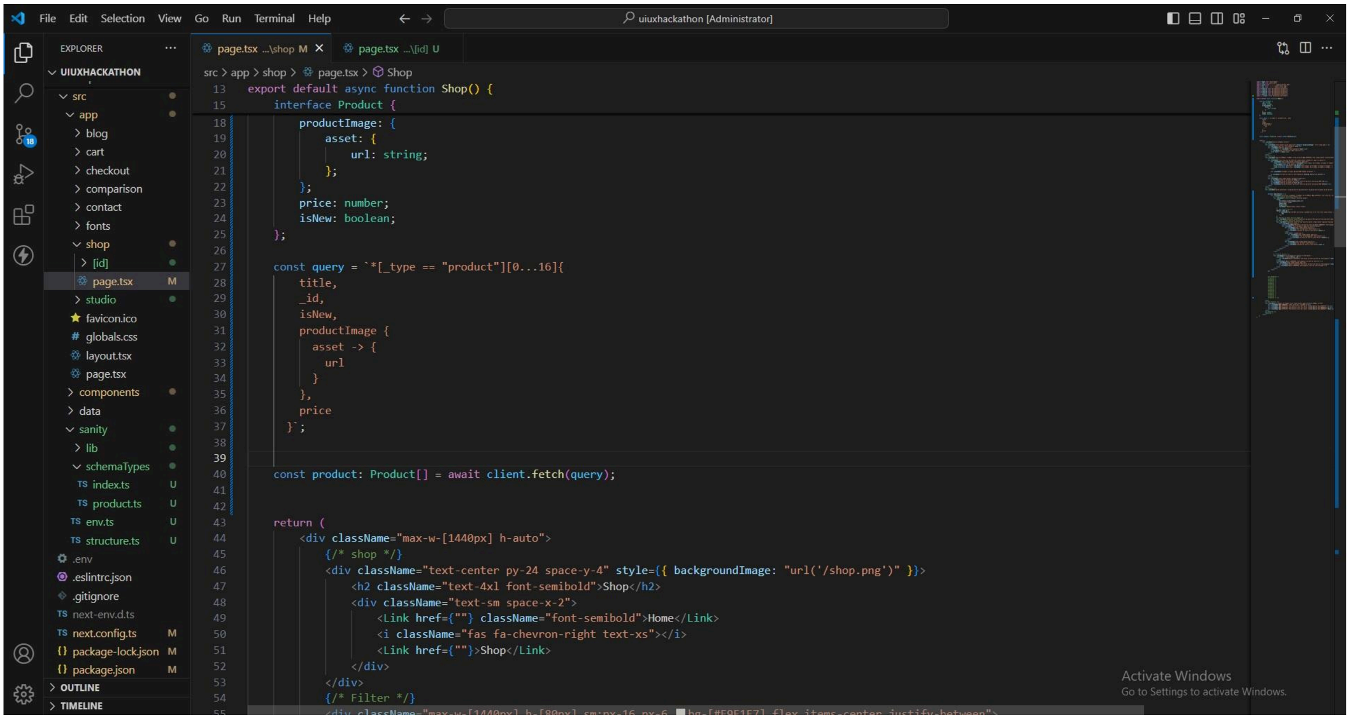
Serene Seat

$350

Sunny Chic

$400

Show More



```
export default async function Shop() {
    interface Product {
        productImage: {
            asset: {
                url: string;
            };
        };
        price: number;
        isNew: boolean;
    };

    const query = `*[_type == "product"][0...16]{
        title,
        _id,
        isNew,
        productImage {
          asset -> {
            url
          }
        },
        price
      }`;


    const product: Product[] = await client.fetch(query);


    return (
        <div className="max-w-[1440px] h-auto">
            {/* shop */}
            <div className="text-center py-24 space-y-4" style={{ backgroundImage: "url('/shop.png')" }}>
                <h2 className="text-4xl font-semibold">Shop</h2>
                <div className="text-sm space-x-2">
                    <Link href={""} className="font-semibold">Home</Link>
                    <i className="fas fa-chevron-right text-xs"></i>
                    <Link href={""}>Shop</Link>
                </div>
            </div>
            {/* Filter */}
            <div className="max-w-[1440px] h-[80px] sm:px-16 px-6 bg-[#F9F1E7] flex items-center justify-between">
```