Day 3 - API Integration Report - [Bandage]

API Integration Process

API Url : https://template6-six.vercel.app/api/products

1. **Identify Requirements**:
   - Defined endpoints and data requirements for the project.
   - Assessed API authentication (e.g., tokens, keys).
2. **Set Up API Calls**:
   - Used `fetch` function for HTTP requests.
   - Created reusable utility functions for API calls.
3. **Integrate with Sanity CMS**:
   - Queried data using Sanity's GROQ queries.
   - Used `@sanity/client` to communicate with the Sanity API.
4. **Error Handling**:
   - Implemented try-catch blocks and fallback data for better user experience.
5. **Testing**:
   - Tested endpoints with tools like Postman or Insomnia.

- Verified integration in local and production environments.

Adjustments Made to Schemas

1. **Field Updates**:
   - Added or updated fields to match API response structure.
   - Used field types like `string`, `array`, and `reference`.
2. **Validation**:
   - Applied validation rules to ensure data consistency.
3. **Relationships**:
   - Linked documents using `reference` fields.
4. **Versioning**:
   - Kept track of schema changes with version comments.

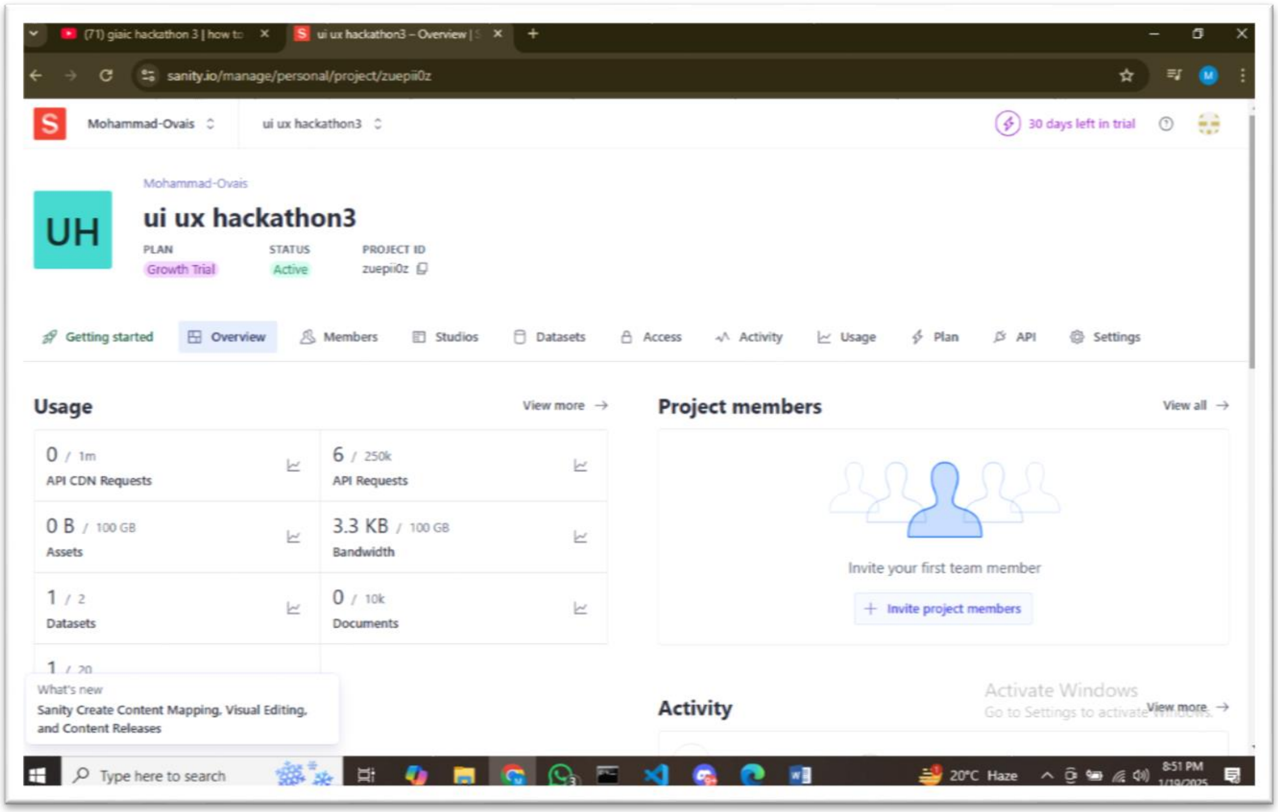
Migration Steps and Tools Used

1. **Data Backup**:
   - Exported existing data from Sanity CMS.
2. **Schema Update**:
   - Updated schema files locally.

   - Deployed updated schemas using the Sanity CLI (`sanity deploy`).
3. **Data Migration**:
   - Used custom scripts or the Sanity import/export feature for data migration.
4. **Testing**:
   - Verified schema and data alignment in both local and production environments.
   - Checked API integration after migration.
5. **Deployment**:
   - Deployed the Next.js app to the hosting platform (e.g., Vercel).

Tools Used:

- **Sanity CLI**: For managing schemas and data.
- **Next.js**: For server-side rendering and API routes.
- **Postman**: For API testing.
- **Git**: For version control.
- **Vercel**: For deployment.



sanity.io/manage/personal/project/zuepii0z

Mohammad-Ovais ui ux hackathon3 30 days left in trial

Mohammad-Ovais

ui ux hackathon3

PLAN Growth Trial STATUS Active PROJECT ID zuepii0z

Getting started · Overview · Members · Studios · Datasets · Access · Activity · Usage · Plan · API · Settings

Usage View more

0 / 1m API CDN Requests	6 / 250k API Requests
0 B / 100 GB Assets	3.3 KB / 100 GB Bandwidth
1 / 2 Datasets	0 / 10k Documents
1 / 20	

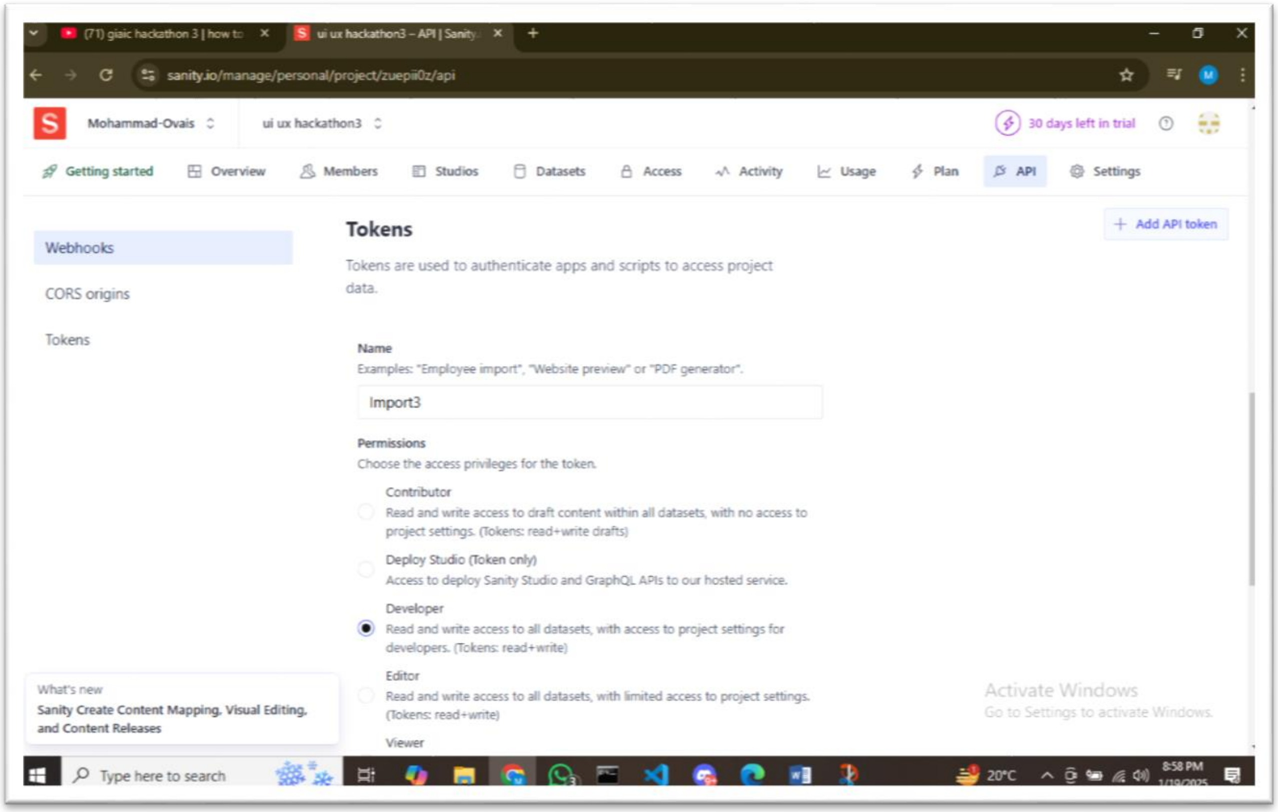
Project members View all

Invite your first team member

+ Invite project members

What's new
Sanity Create Content Mapping, Visual Editing, and Content Releases

Activity View more



sanity.io/manage/personal/project/zuepii0z/api

Mohammad-Ovais ui ux hackathon3 30 days left in trial

Getting started · Overview · Members · Studios · Datasets · Access · Activity · Usage · Plan · API · Settings

Webhooks
CORS origins
Tokens

Tokens + Add API token

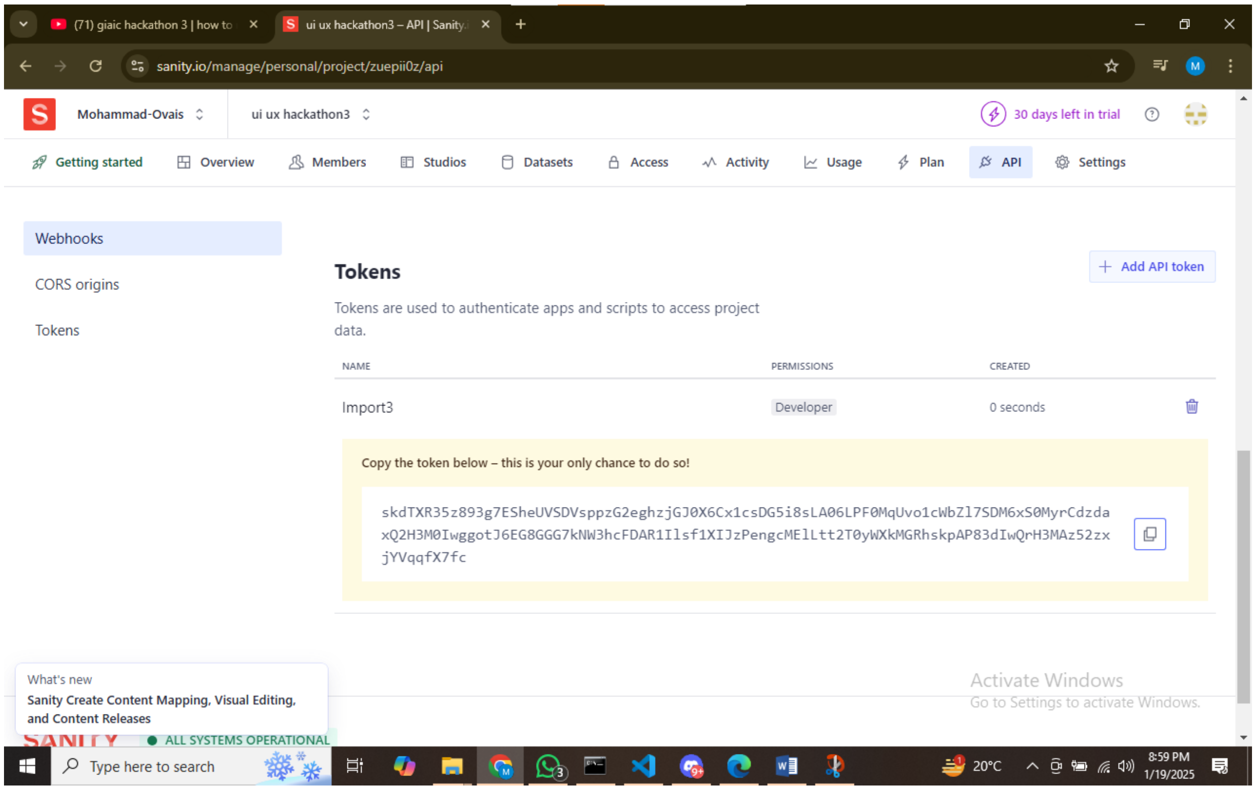
Tokens are used to authenticate apps and scripts to access project data.

Name
Examples: "Employee import", "Website preview" or "PDF generator".
Import3

Permissions
Choose the access privileges for the token.

- Contributor — Read and write access to draft content within all datasets, with no access to project settings. (Tokens: read+write drafts)
- Deploy Studio (Token only) — Access to deploy Sanity Studio and GraphQL APIs to our hosted service.
- [✓ Developer] — Read and write access to all datasets, with access to project settings for developers. (Tokens: read+write)
- Editor — Read and write access to all datasets, with limited access to project settings. (Tokens: read+write)
- Viewer

What's new
Sanity Create Content Mapping, Visual Editing, and Content Releases


sanity.io/manage/personal/project/zuepii0z/api

Mohammad-Ovais · ui ux hackathon3 · 30 days left in trial

Getting started · Overview · Members · Studios · Datasets · Access · Activity · Usage · Plan · API · Settings

Webhooks · CORS origins · Tokens

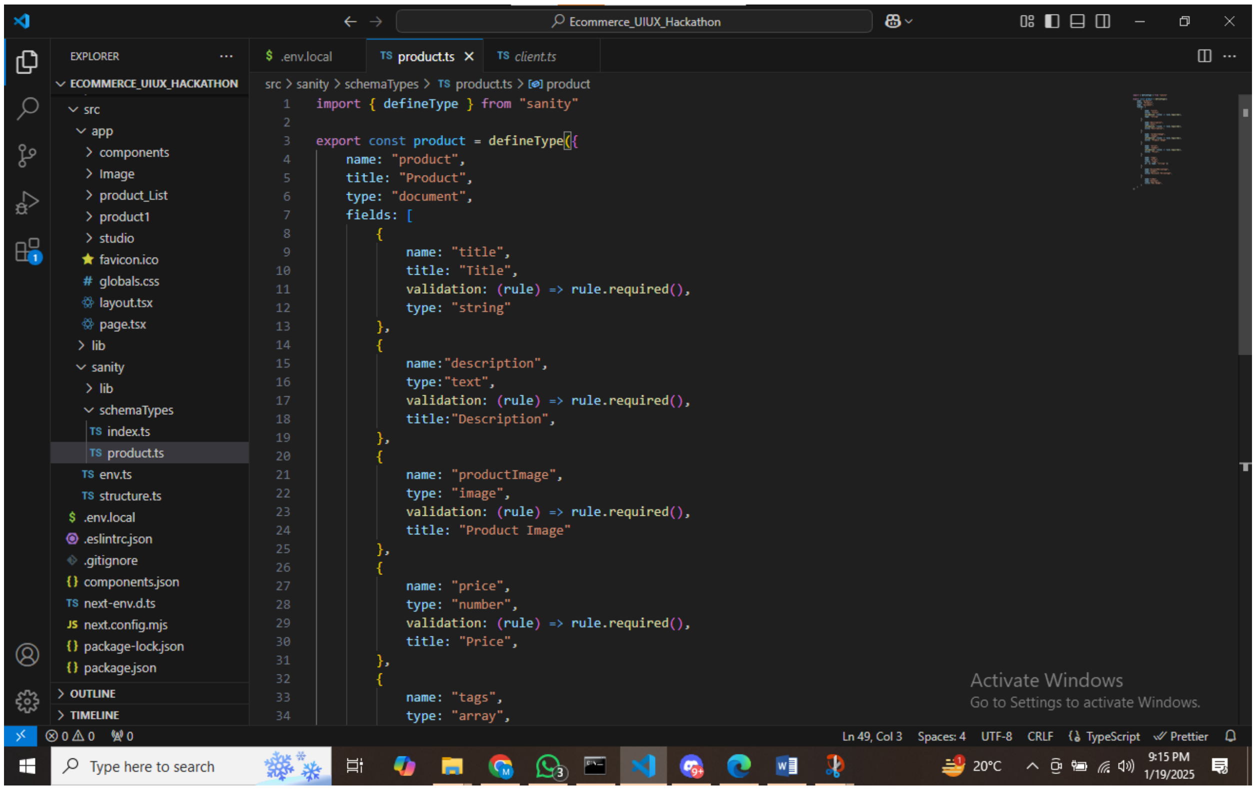
Tokens

Tokens are used to authenticate apps and scripts to access project data.

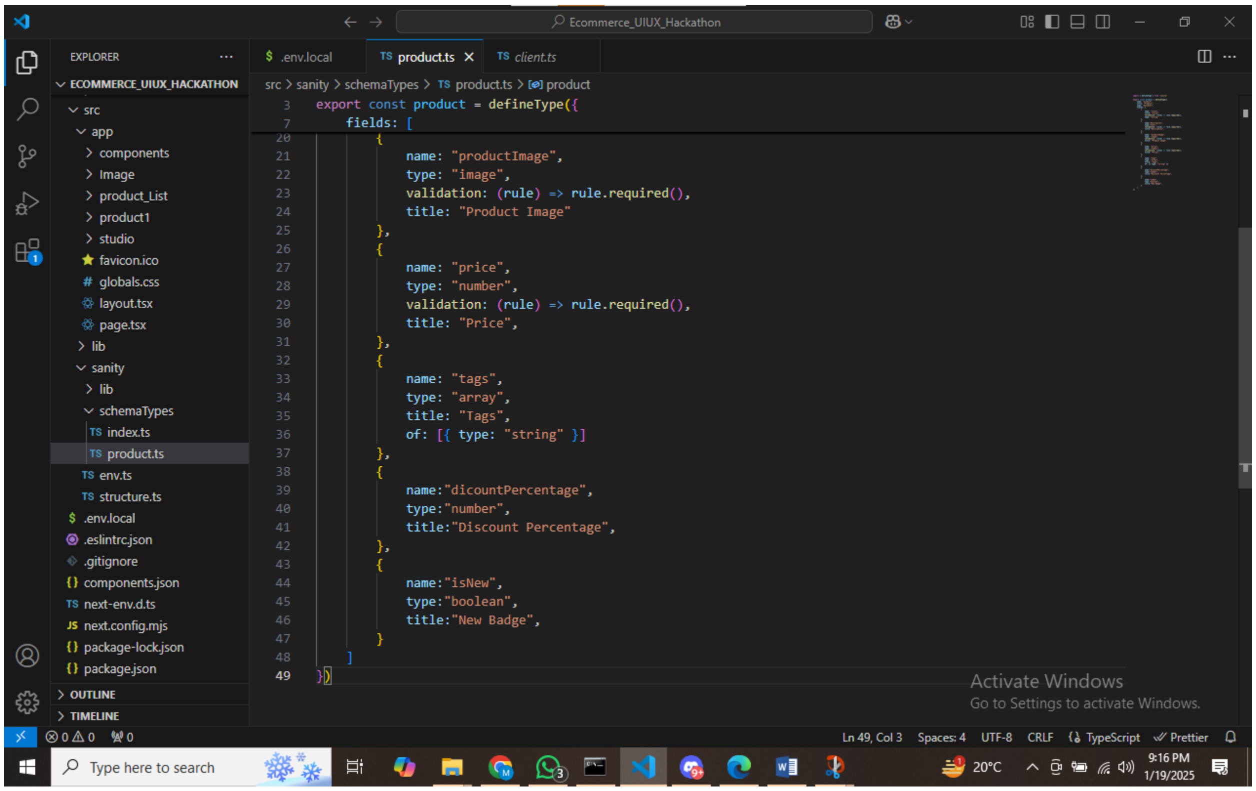
NAME	PERMISSIONS	CREATED
Import3	Developer	0 seconds

Copy the token below – this is your only chance to do so!

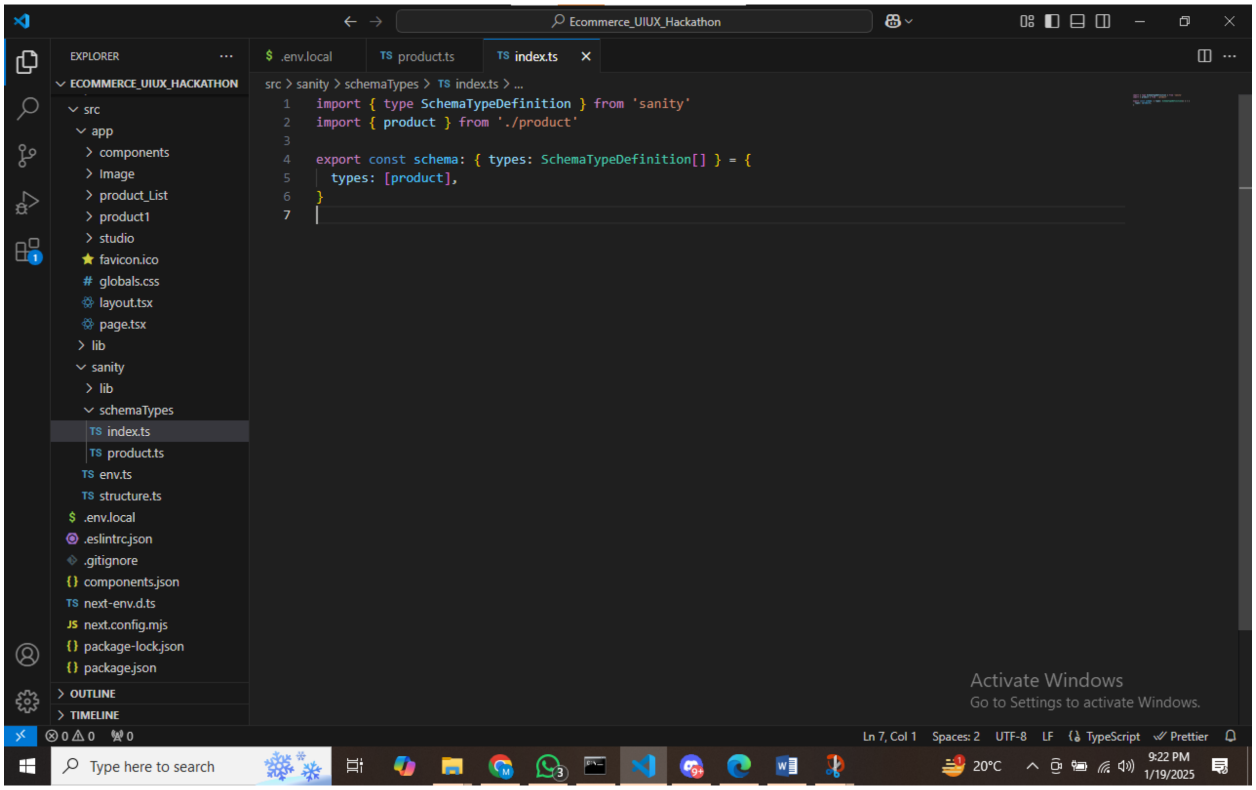
```
skdTXR35z893g7ESheUVSDVsppzG2eghzjGJ0X6Cx1csDG5i8sLA06LPF0MqUvo1cWbZl7SDM6xS0MyrCdzda
xQ2H3M0IwggotJ6EG8GGG7kNW3hcFDAR1Ilsf1XIJzPengcMElLtt2T0yWXkMGRhskpAP83dIwQrH3MAz52zx
jYVqqfX7fc
```


src > sanity > schemaTypes > product.ts > product

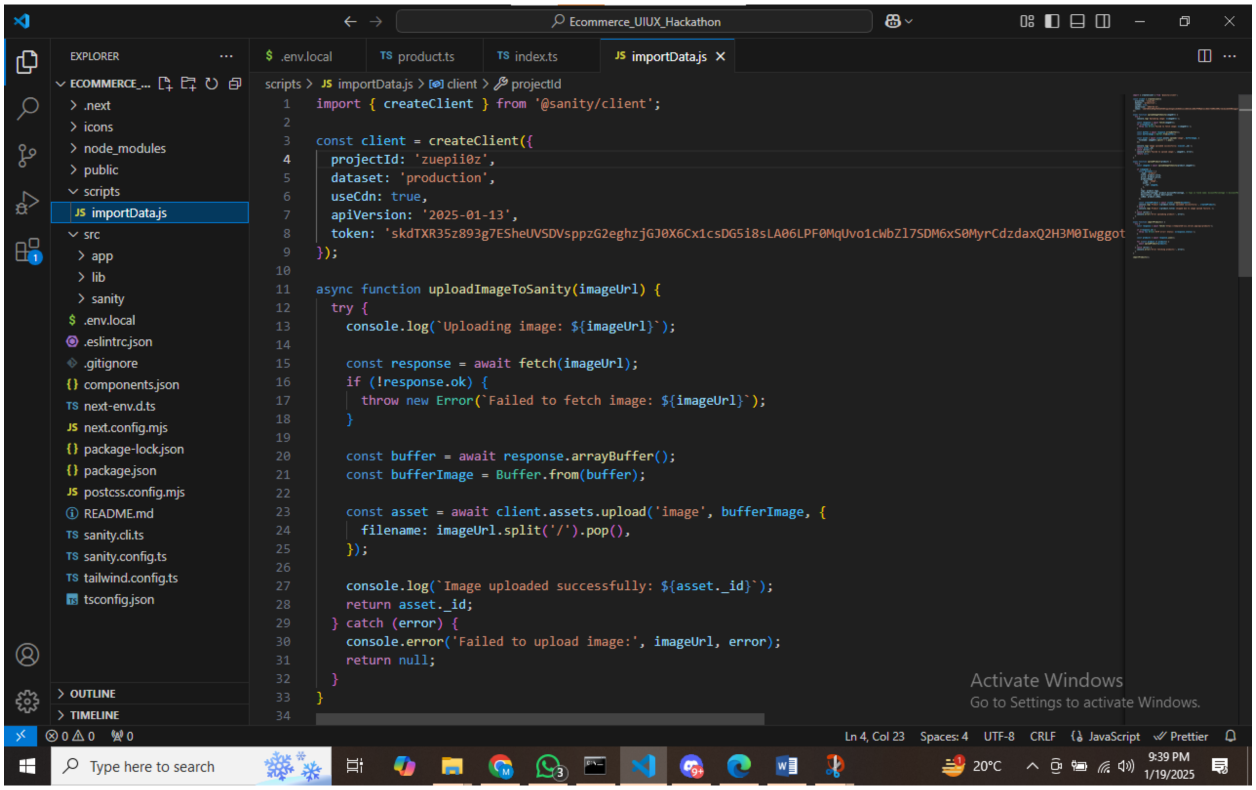
```ts
import { defineType } from "sanity"

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",
```


Ecommerce_UIUX_Hackathon
EXPLORER
ECOMMERCE_UIUX_HACKATHON
.env.local
product.ts
client.ts
src > sanity > schemaTypes > product.ts > product
Ln 49, Col 3 Spaces: 4 UTF-8 CRLF TypeScript Prettier
Activate Windows
Go to Settings to activate Windows.
Type here to search
9:16 PM
1/19/2025

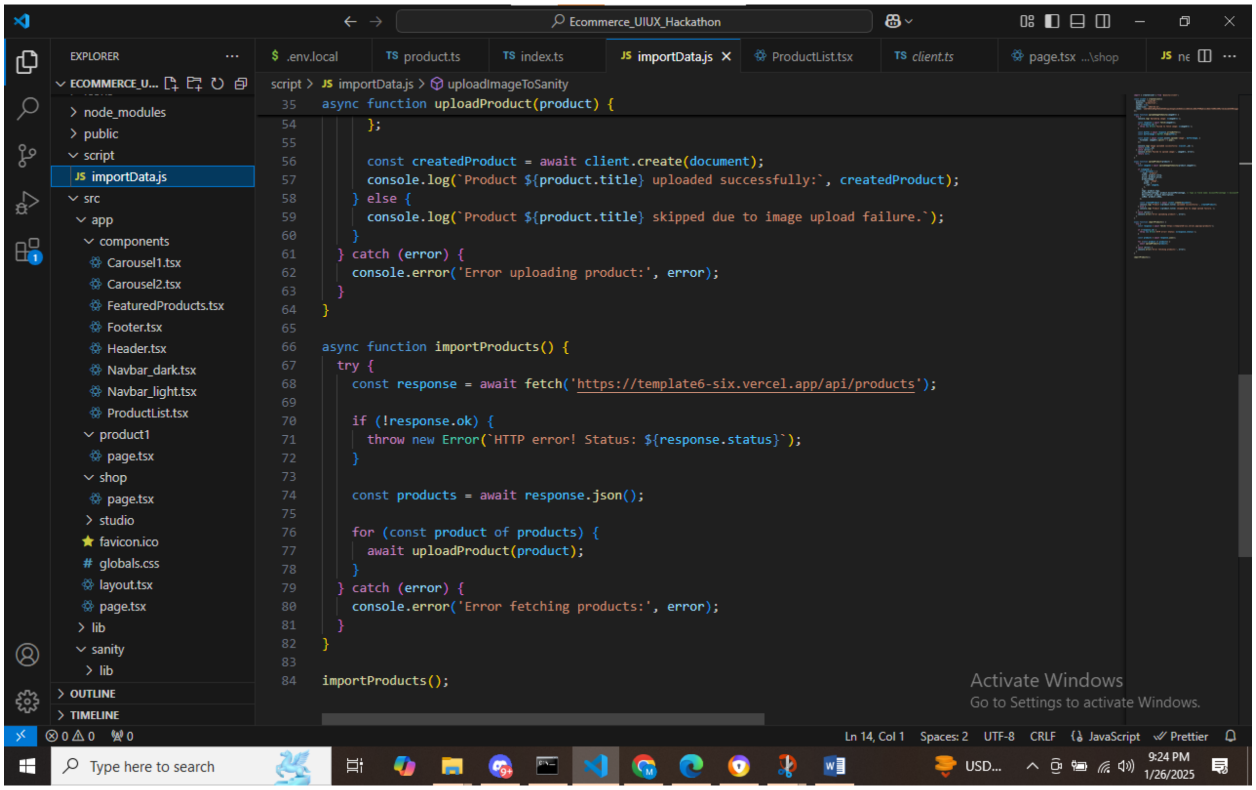
```
export const product = defineType({
    fields: [
            {
                name: "productImage",
                type: "image",
                validation: (rule) => rule.required(),
                title: "Product Image"
            },
            {
                name: "price",
                type: "number",
                validation: (rule) => rule.required(),
                title: "Price",
            },
            {
                name: "tags",
                type: "array",
                title: "Tags",
                of: [{ type: "string" }]
            },
            {
                name:"dicountPercentage",
                type:"number",
                title:"Discount Percentage",
            },
            {
                name:"isNew",
                type:"boolean",
                title:"New Badge",
            }
        ]
})
```



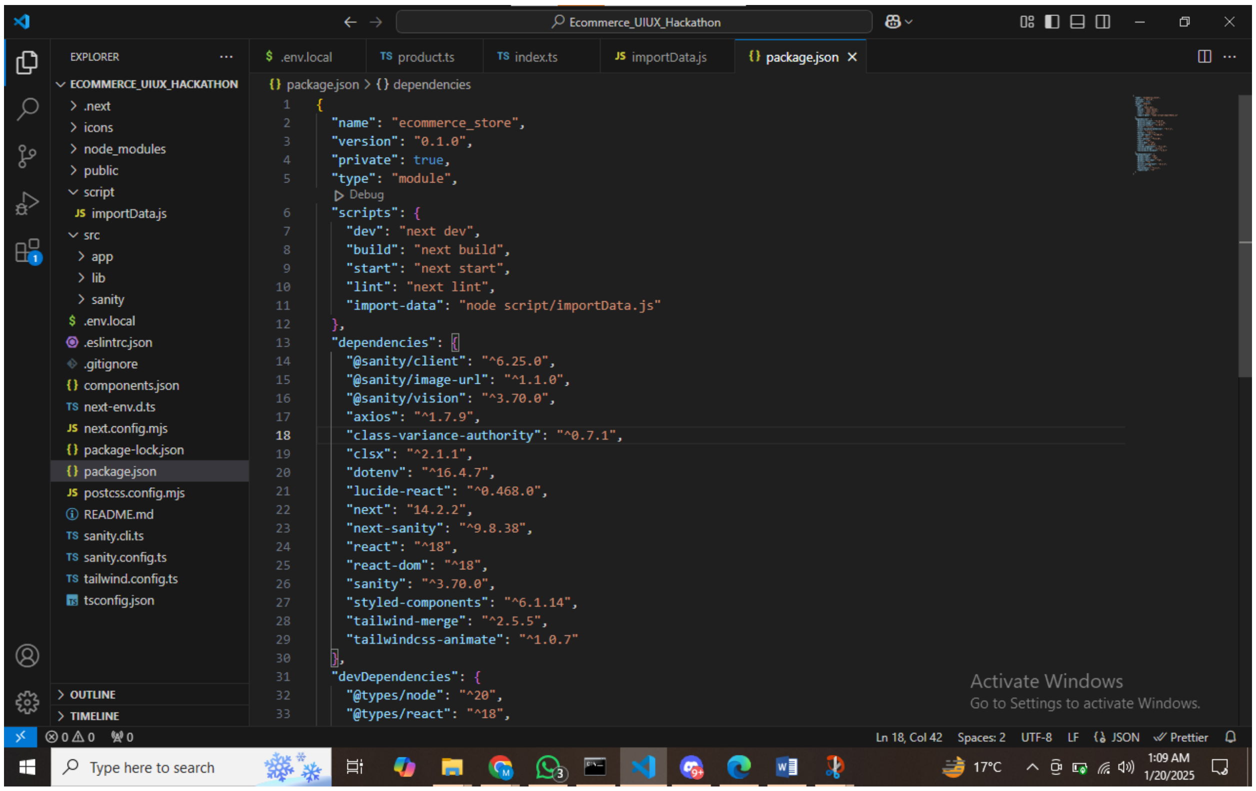
```
import { type SchemaTypeDefinition } from 'sanity'
import { product } from './product'

export const schema: { types: SchemaTypeDefinition[] } = {
  types: [product],
}
```



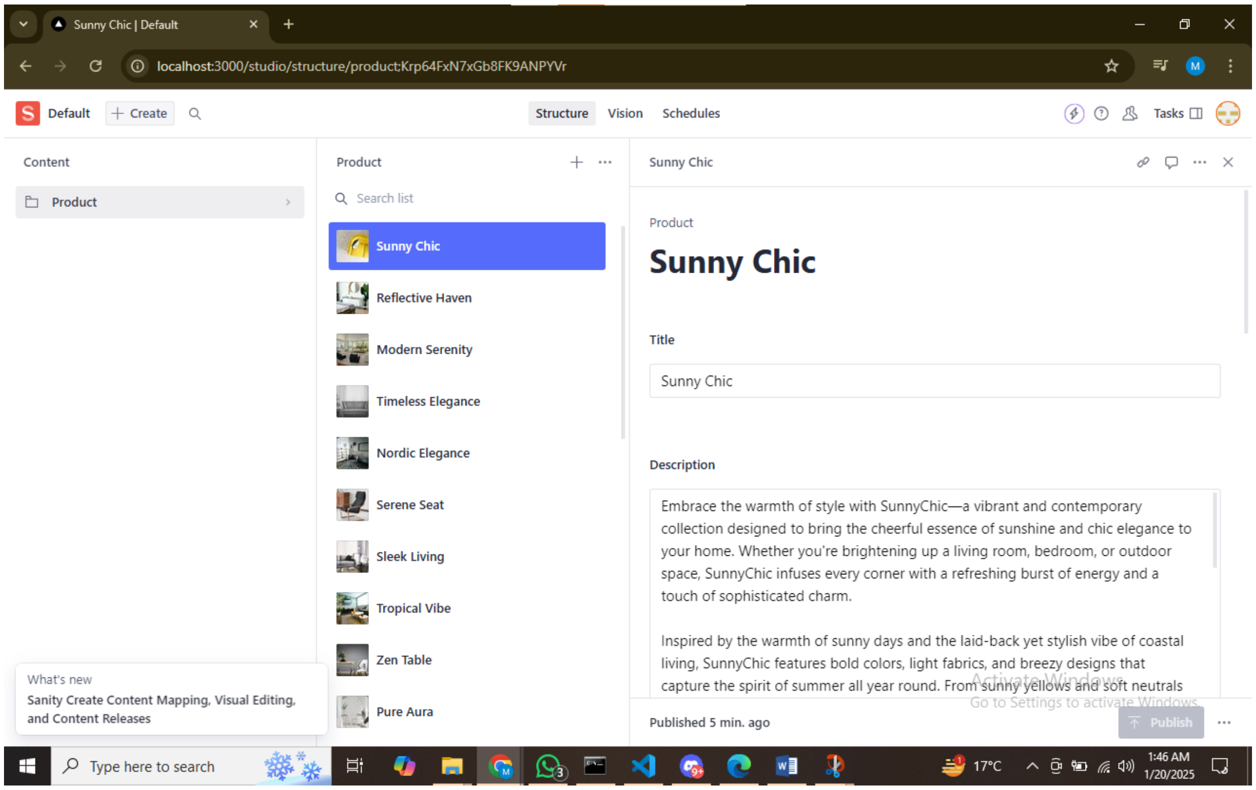
```
import { createClient } from '@sanity/client';

const client = createClient({
  projectId: 'zuepii0z',
  dataset: 'production',
  useCdn: true,
  apiVersion: '2025-01-13',
  token: 'skdTXR35z893g7ESheUVSDVsppzG2eghzjGJ0X6Cx1csDG5i8sLA06LPF0MqUvo1cWbZl7SDM6xS0MyrCdzdaxQ2H3M0Iwggot
});

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) {
      };

      const createdProduct = await client.create(document);
      console.log(`Product ${product.title} uploaded successfully:`, createdProduct);
    } else {
      console.log(`Product ${product.title} skipped due to image upload failure.`);
    }
  } catch (error) {
    console.error('Error uploading product:', error);
  }
}

async function importProducts() {
  try {
    const response = await fetch('https://template6-six.vercel.app/api/products');

    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }

    const products = await response.json();

    for (const product of products) {
      await uploadProduct(product);
    }
  } catch (error) {
    console.error('Error fetching products:', error);
  }
}

importProducts();
```



```json
{
  "name": "ecommerce_store",
  "version": "0.1.0",
  "private": true,
  "type": "module",
  "scripts": {
    "dev": "next dev",
    "build": "next build",
    "start": "next start",
    "lint": "next lint",
    "import-data": "node script/importData.js"
  },
  "dependencies": {
    "@sanity/client": "^6.25.0",
    "@sanity/image-url": "^1.1.0",
    "@sanity/vision": "^3.70.0",
    "axios": "^1.7.9",
    "class-variance-authority": "^0.7.1",
    "clsx": "^2.1.1",
    "dotenv": "^16.4.7",
    "lucide-react": "^0.468.0",
    "next": "14.2.2",
    "next-sanity": "^9.8.38",
    "react": "^18",
    "react-dom": "^18",
    "sanity": "^3.70.0",
    "styled-components": "^6.1.14",
    "tailwind-merge": "^2.5.5",
    "tailwindcss-animate": "^1.0.7"
  },
  "devDependencies": {
    "@types/node": "^20",
    "@types/react": "^18",
```



Product

Sunny Chic

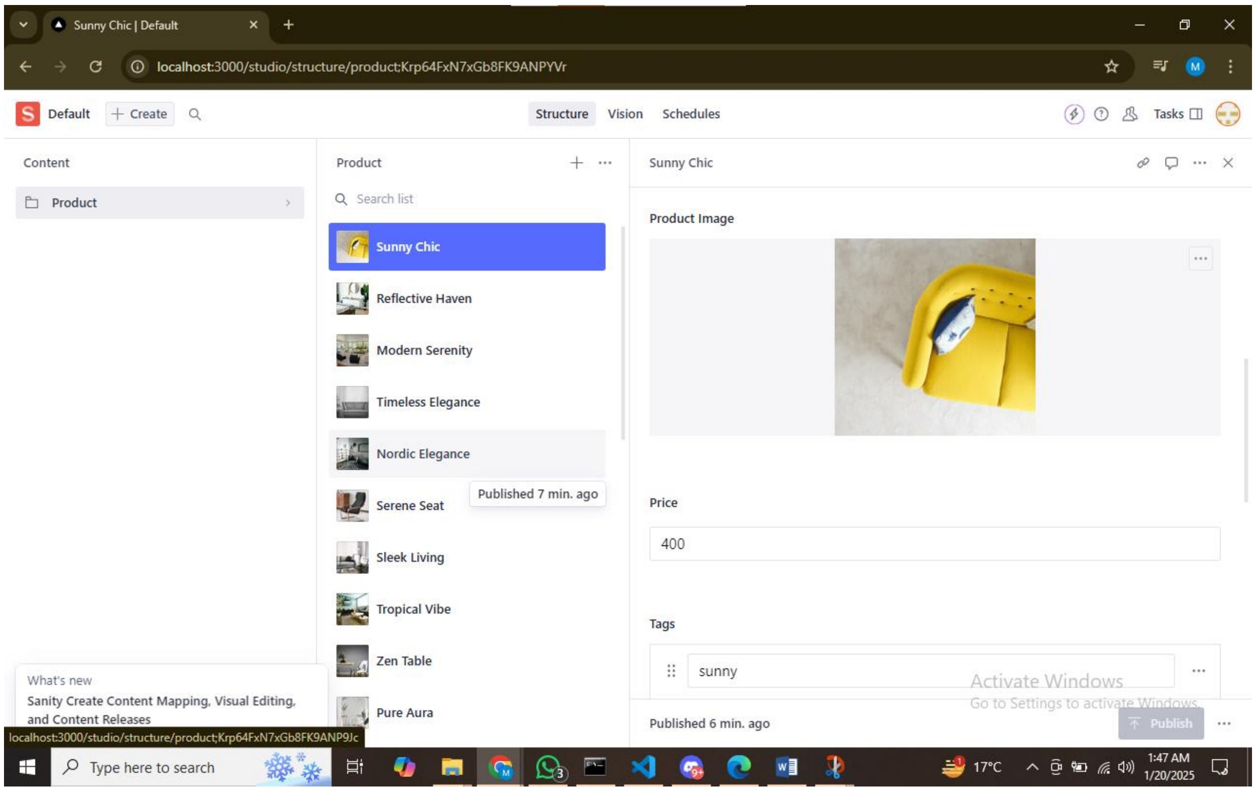
Title

Sunny Chic

Description

Embrace the warmth of style with SunnyChic—a vibrant and contemporary collection designed to bring the cheerful essence of sunshine and chic elegance to your home. Whether you're brightening up a living room, bedroom, or outdoor space, SunnyChic infuses every corner with a refreshing burst of energy and a touch of sophisticated charm.

Inspired by the warmth of sunny days and the laid-back yet stylish vibe of coastal living, SunnyChic features bold colors, light fabrics, and breezy designs that capture the spirit of summer all year round. From sunny yellows and soft neutrals


Sunny Chic | Default
localhost:3000/studio/structure/product;Krp64FxN7xGb8FK9ANPYVr
Default
Create
Structure
Vision
Schedules
Tasks

Content

Product

Product

Search list

- Sunny Chic
- Reflective Haven
- Modern Serenity
- Timeless Elegance
- Nordic Elegance
- Serene Seat
- Sleek Living
- Tropical Vibe
- Zen Table
- Pure Aura

Published 7 min. ago

Sunny Chic

Product Image

Price

400

Tags

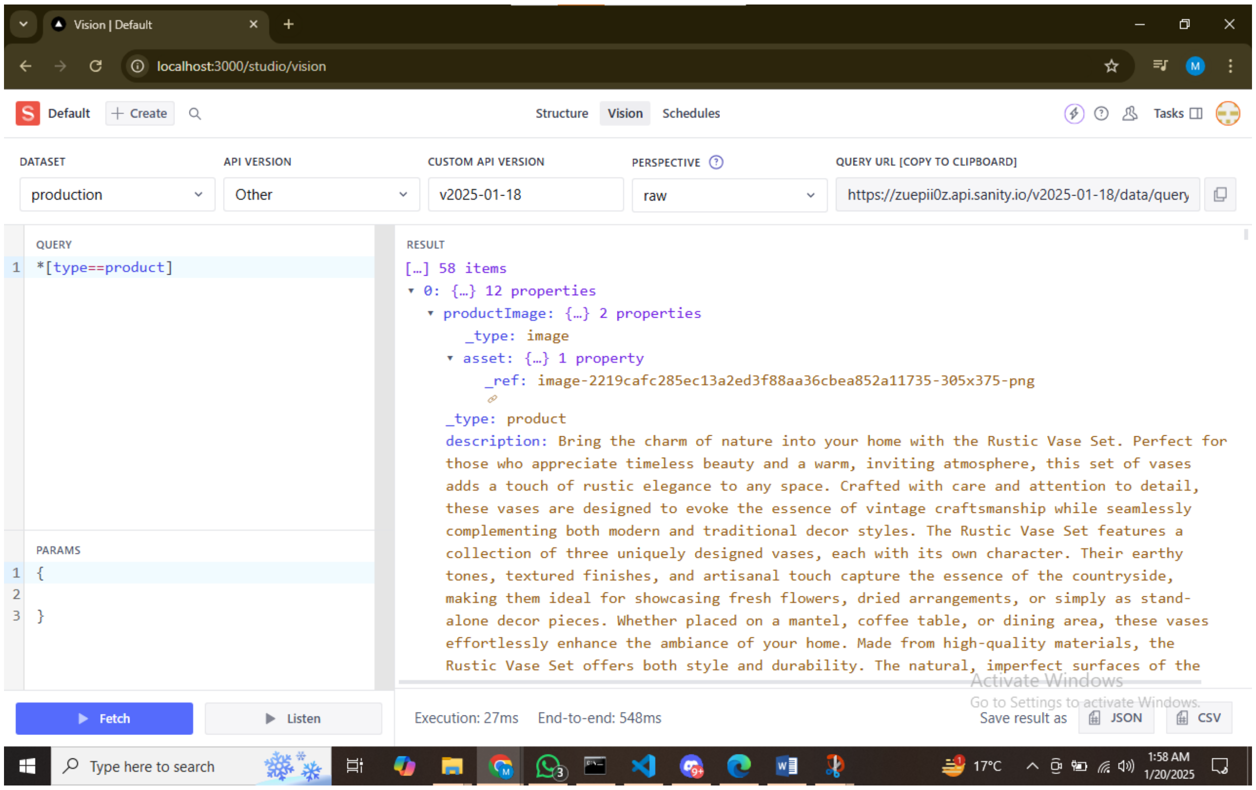
sunny

Published 6 min. ago

Publish

What's new
Sanity Create Content Mapping, Visual Editing, and Content Releases

localhost:3000/studio/structure/product;Krp64FxN7xGb8FK9ANP9Jc



DATASET	API VERSION	CUSTOM API VERSION	PERSPECTIVE	QUERY URL [COPY TO CLIPBOARD]
production	Other	v2025-01-18	raw	https://zuepii0z.api.sanity.io/v2025-01-18/data/query

QUERY

```
*[type==product]
```

PARAMS

```
{

}
```

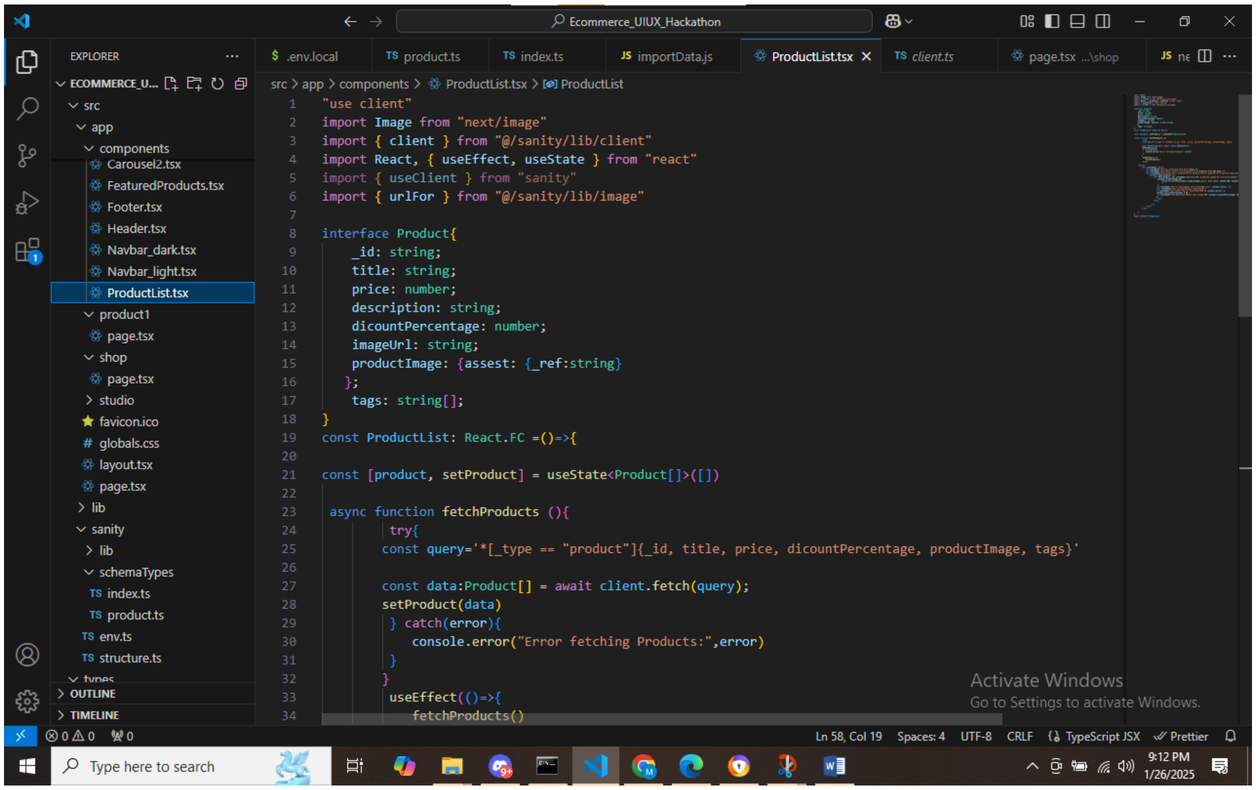
RESULT

```
[…] 58 items
▾ 0: {…} 12 properties
  ▾ productImage: {…} 2 properties
      _type: image
    ▾ asset: {…} 1 property
        _ref: image-2219cafc285ec13a2ed3f88aa36cbea852a11735-305x375-png
    _type: product
    description: Bring the charm of nature into your home with the Rustic Vase Set. Perfect for
    those who appreciate timeless beauty and a warm, inviting atmosphere, this set of vases
    adds a touch of rustic elegance to any space. Crafted with care and attention to detail,
    these vases are designed to evoke the essence of vintage craftsmanship while seamlessly
    complementing both modern and traditional decor styles. The Rustic Vase Set features a
    collection of three uniquely designed vases, each with its own character. Their earthy
    tones, textured finishes, and artisanal touch capture the essence of the countryside,
    making them ideal for showcasing fresh flowers, dried arrangements, or simply as stand-
    alone decor pieces. Whether placed on a mantel, coffee table, or dining area, these vases
    effortlessly enhance the ambiance of your home. Made from high-quality materials, the
    Rustic Vase Set offers both style and durability. The natural, imperfect surfaces of the
```

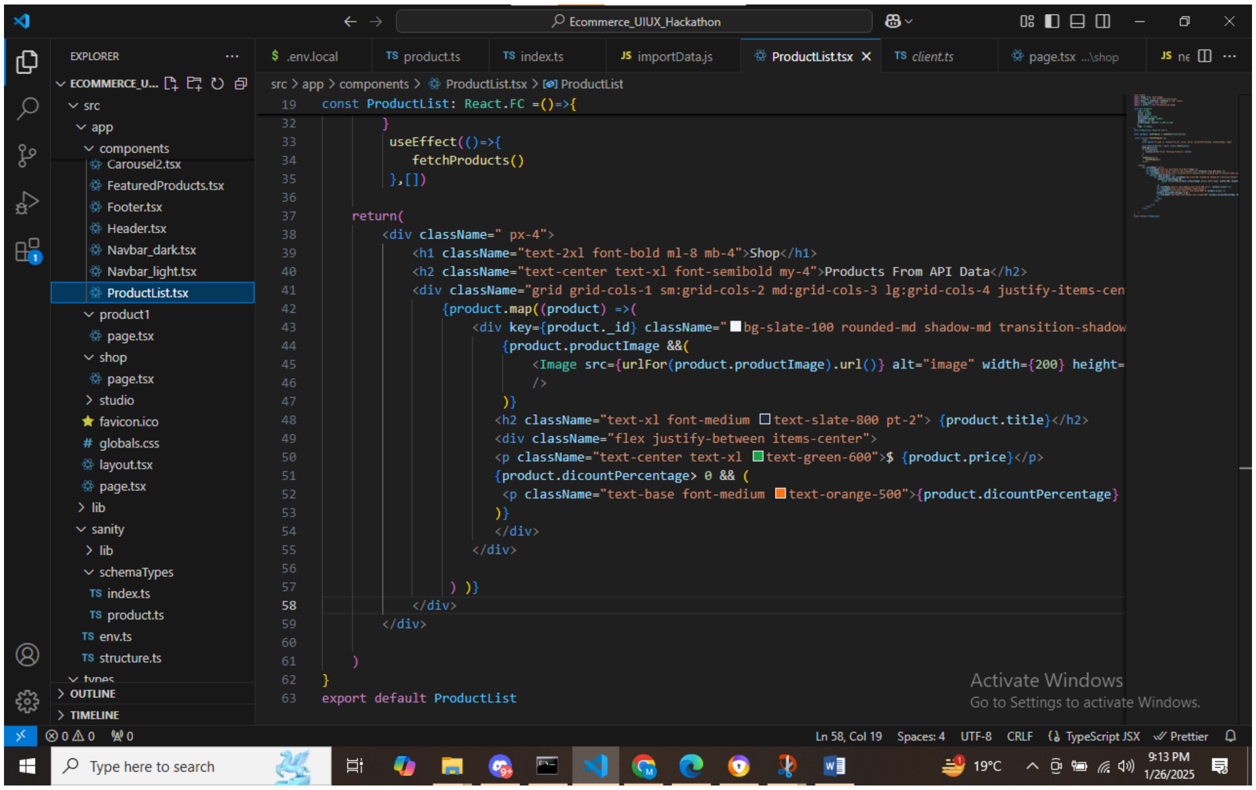
Fetch   Listen

Execution: 27ms   End-to-end: 548ms

Save result as   JSON   CSV

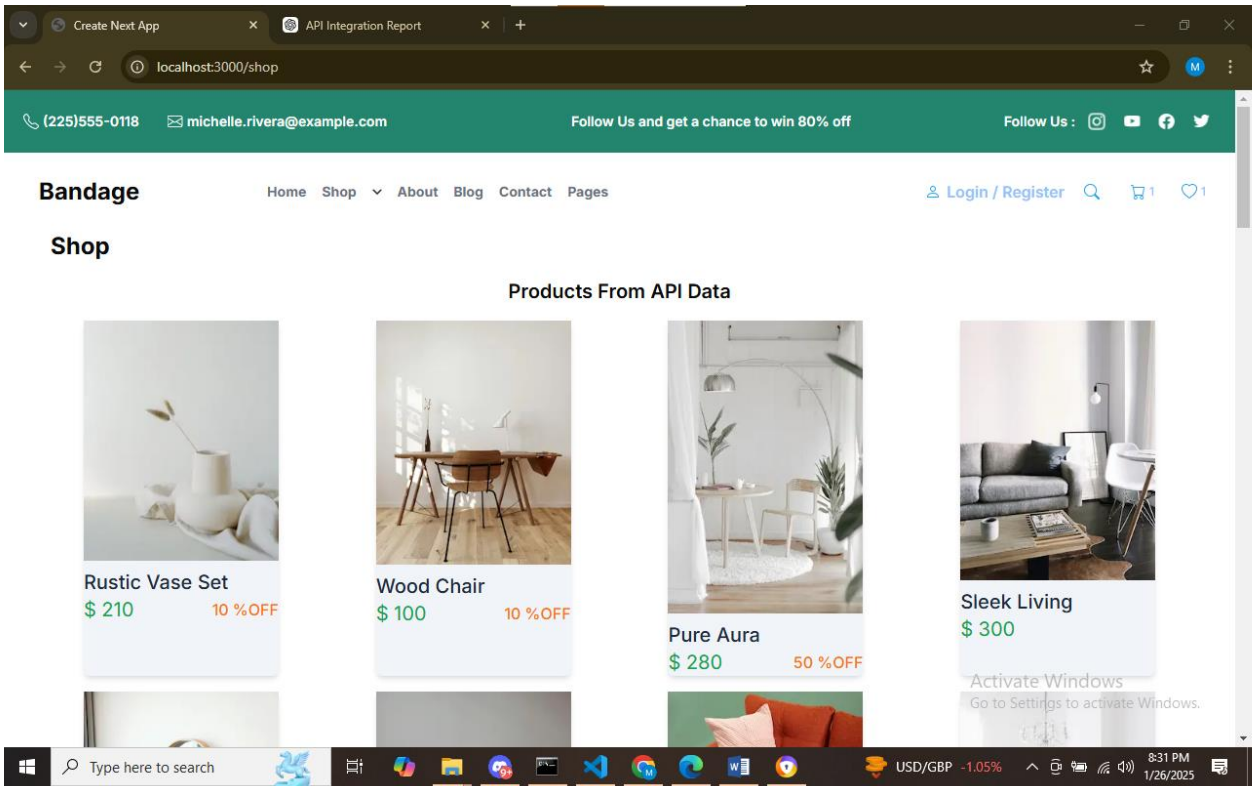


```tsx
"use client"
import Image from "next/image"
import { client } from "@/sanity/lib/client"
import React, { useEffect, useState } from "react"
import { useClient } from "sanity"
import { urlFor } from "@/sanity/lib/image"

interface Product{
    _id: string;
    title: string;
    price: number;
    description: string;
    dicountPercentage: number;
    imageUrl: string;
    productImage: {assest: {_ref:string}
   };
    tags: string[];
}
const ProductList: React.FC =()=>{

const [product, setProduct] = useState<Product[]>([])

 async function fetchProducts (){
        try{
        const query='*[_type == "product"]{_id, title, price, dicountPercentage, productImage, tags}'

        const data:Product[] = await client.fetch(query);
        setProduct(data)
        } catch(error){
            console.error("Error fetching Products:",error)
        }
        }
        useEffect(()=>{
           fetchProducts()
```



```tsx
const ProductList: React.FC =()=>{
        }
        useEffect(()=>{
           fetchProducts()
        },[])

    return(
        <div className=" px-4">
            <h1 className="text-2xl font-bold ml-8 mb-4">Shop</h1>
            <h2 className="text-center text-xl font-semibold my-4">Products From API Data</h2>
            <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 justify-items-cen
                {product.map((product) =>(
                    <div key={product._id} className="bg-slate-100 rounded-md shadow-md transition-shadow
                     {product.productImage &&(
                         <Image src={urlFor(product.productImage).url()} alt="image" width={200} height=
                         />
                     )}
                     <h2 className="text-xl font-medium text-slate-800 pt-2"> {product.title}</h2>
                     <div className="flex justify-between items-center">
                     <p className="text-center text-xl text-green-600">$ {product.price}</p>
                     {product.dicountPercentage> 0 && (
                      <p className="text-base font-medium text-orange-500">{product.dicountPercentage}
                     )}
                     </div>
                  </div>

               ) )}
            </div>
        </div>

    )
}
export default ProductList
```

Data successfully displayed in the frontend



Shop

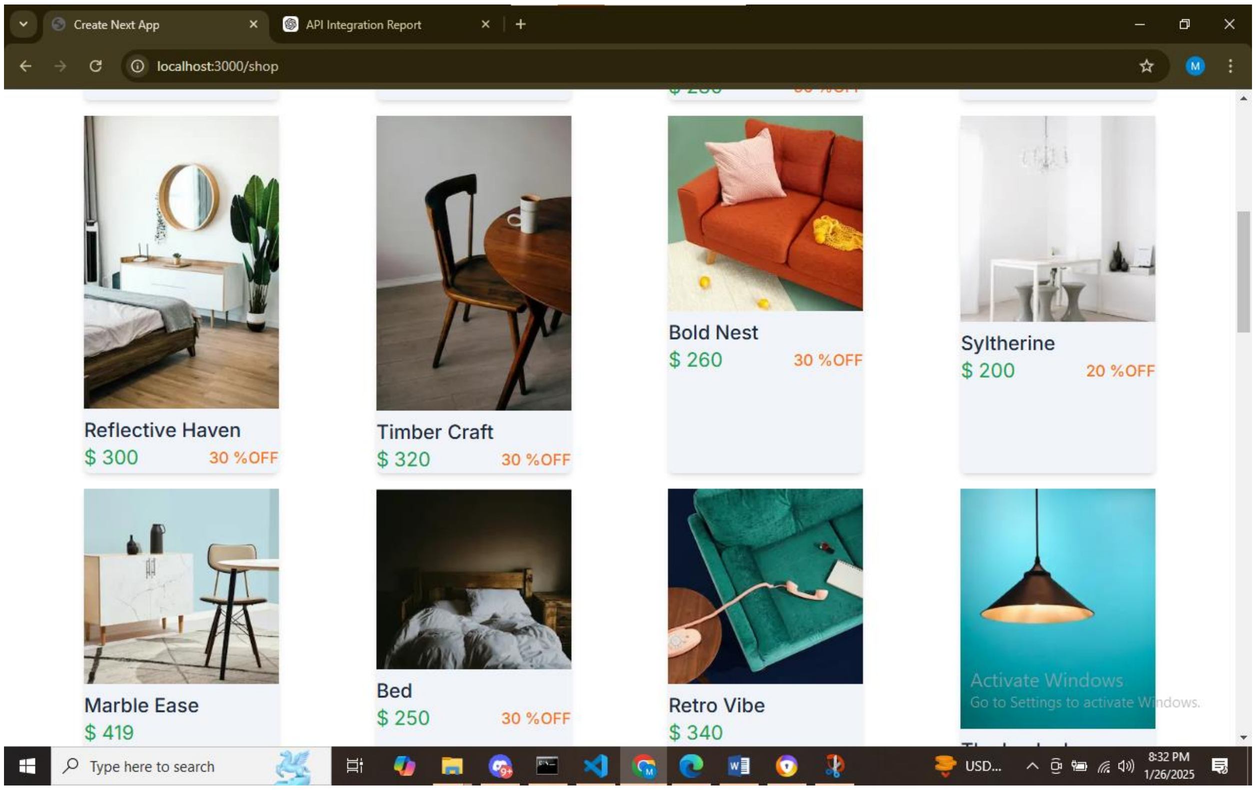
Products From API Data

Rustic Vase Set
$ 210 10 %OFF

Wood Chair
$ 100 10 %OFF

Pure Aura
$ 280 50 %OFF

Sleek Living
$ 300



Reflective Haven
$ 300 30 %OFF

Timber Craft
$ 320 30 %OFF

Bold Nest
$ 260 30 %OFF

Syltherine
$ 200 20 %OFF

Marble Ease
$ 419

Bed
$ 250 30 %OFF

Retro Vibe
$ 340