

# Marketplace Builder Hackathon 2025

## Day 03 – API Integration & Data Migration

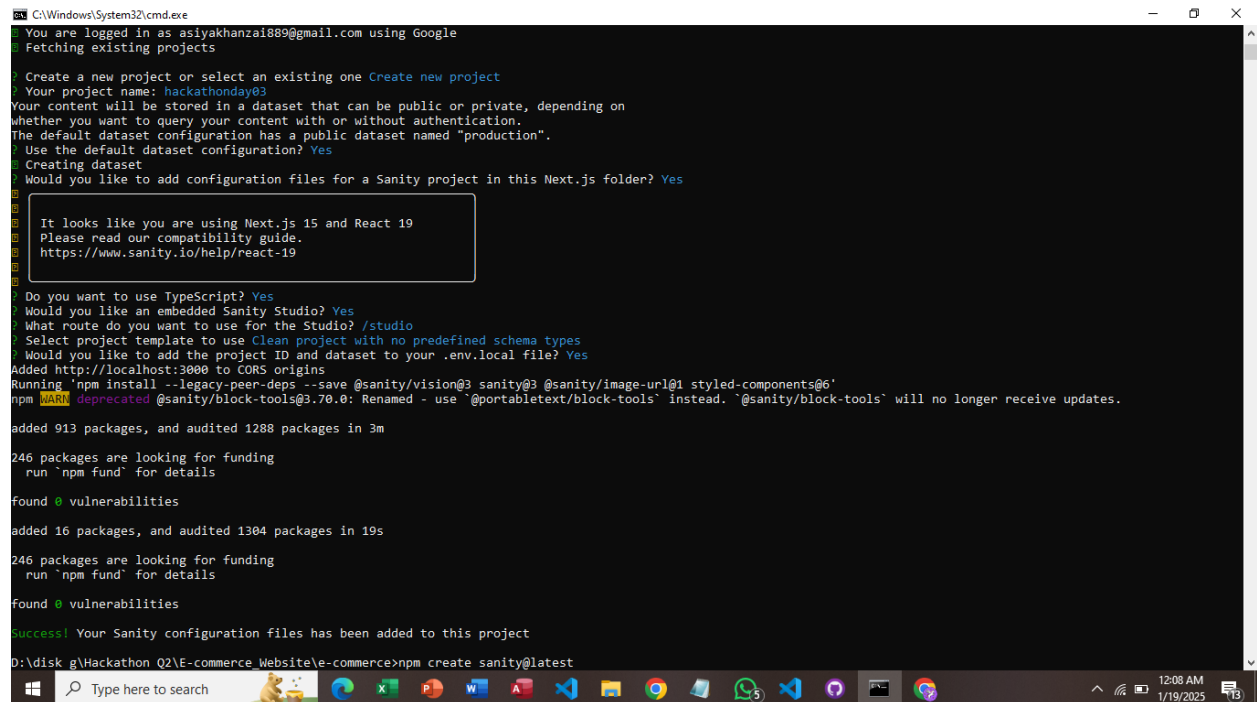
Here's a Report Document:

- Process of API Integrate.
- Adjustments of made to Schema.
- Migration Step to Step.

Provided Screenshots of All Procedure Below:

- Step 1: Make a Next.JS Folder.
- Step 2: Make a Sanity project through CMD. Run a Command:

(npm create sanity@latest)



```
C:\Windows\System32\cmd.exe
You are logged in as asiyakhanzai889@gmail.com using Google
Fetching existing projects

? Create a new project or select an existing one Create new project
? Your project name: hackathonday03
Your content will be stored in a dataset that can be public or private, depending on
whether you want to query your content with or without authentication.
The default dataset configuration has a public dataset named "production".
? Use the default dataset configuration? Yes
? Creating dataset
? Would you like to add configuration files for a Sanity project in this Next.js folder? Yes

It looks like you are using Next.js 15 and React 19
Please read our compatibility guide.
https://www.sanity.io/help/react-19

? Do you want to use TypeScript? Yes
? Would you like an embedded Sanity Studio? Yes
? What route do you want to use for the Studio? /studio
? Select project template to use Clean project with no predefined schema types
? Would you like to add the project ID and dataset to your .env.local file? Yes
Added http://localhost:3000 to CORS origins
Running 'npm install --legacy-peer-deps --save @sanity/vision@3 sanity@3 @sanity/image-url@1 styled-components@6'
npm WARN deprecated @sanity/block-tools@3.70.0: Renamed - use '@portabletext/block-tools' instead. '@sanity/block-tools' will no longer receive updates.

added 913 packages, and audited 1288 packages in 3m

246 packages are looking for funding
  run 'npm fund' for details

found 0 vulnerabilities

added 16 packages, and audited 1304 packages in 19s

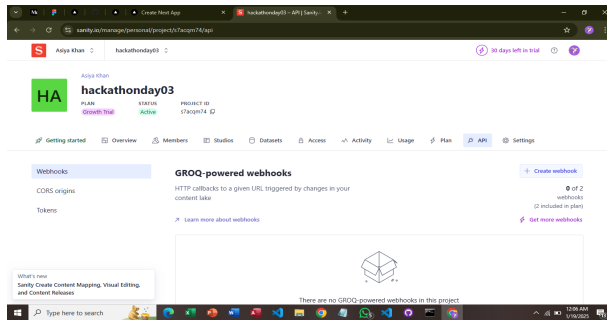
246 packages are looking for funding
  run 'npm fund' for details

found 0 vulnerabilities

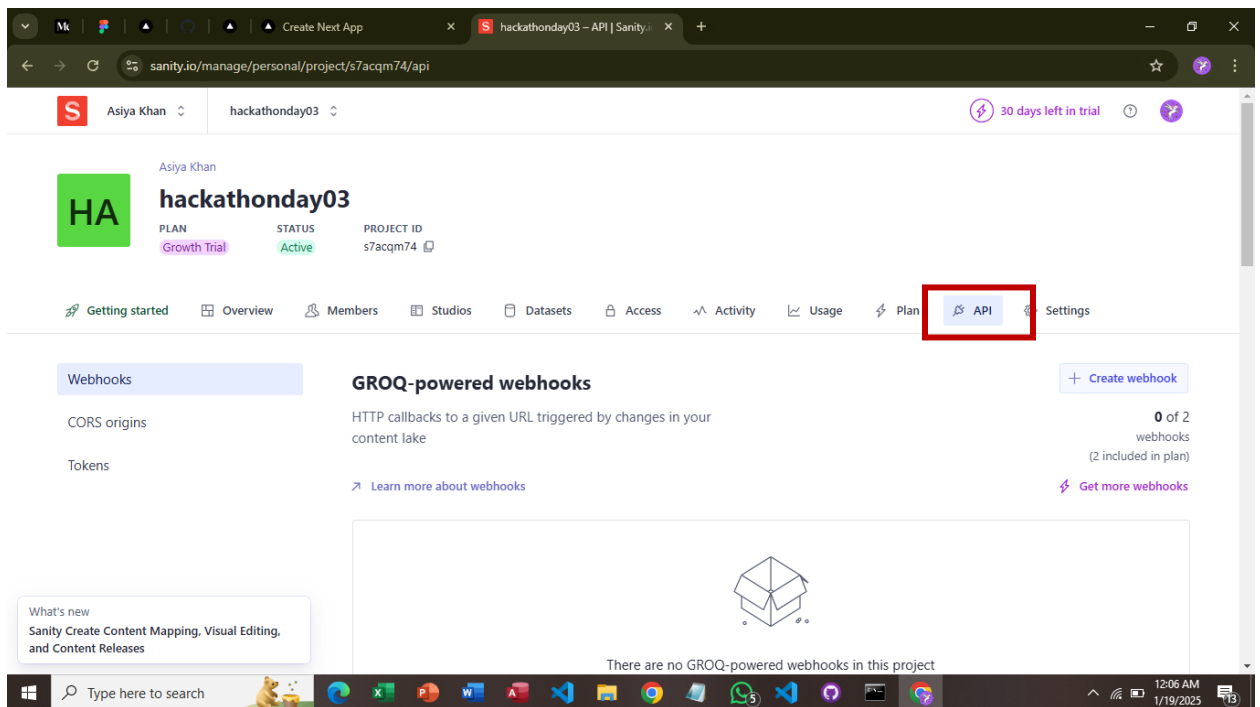
Success! Your Sanity configuration files has been added to this project

D:\disk g\Hackathon Q2\E-commerce Website\E-commerce>npm create sanity@latest
```

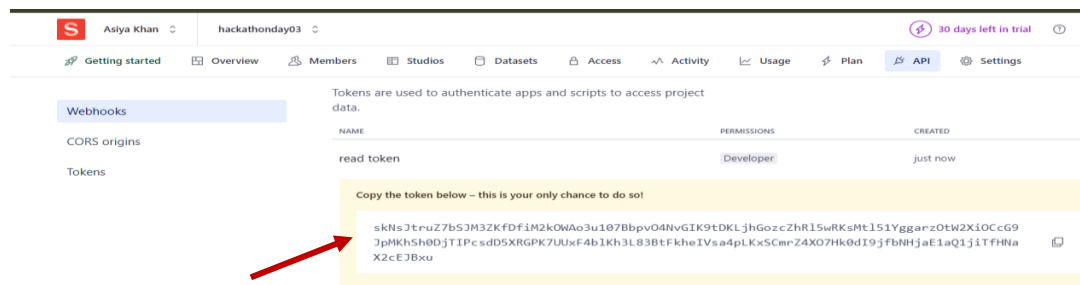
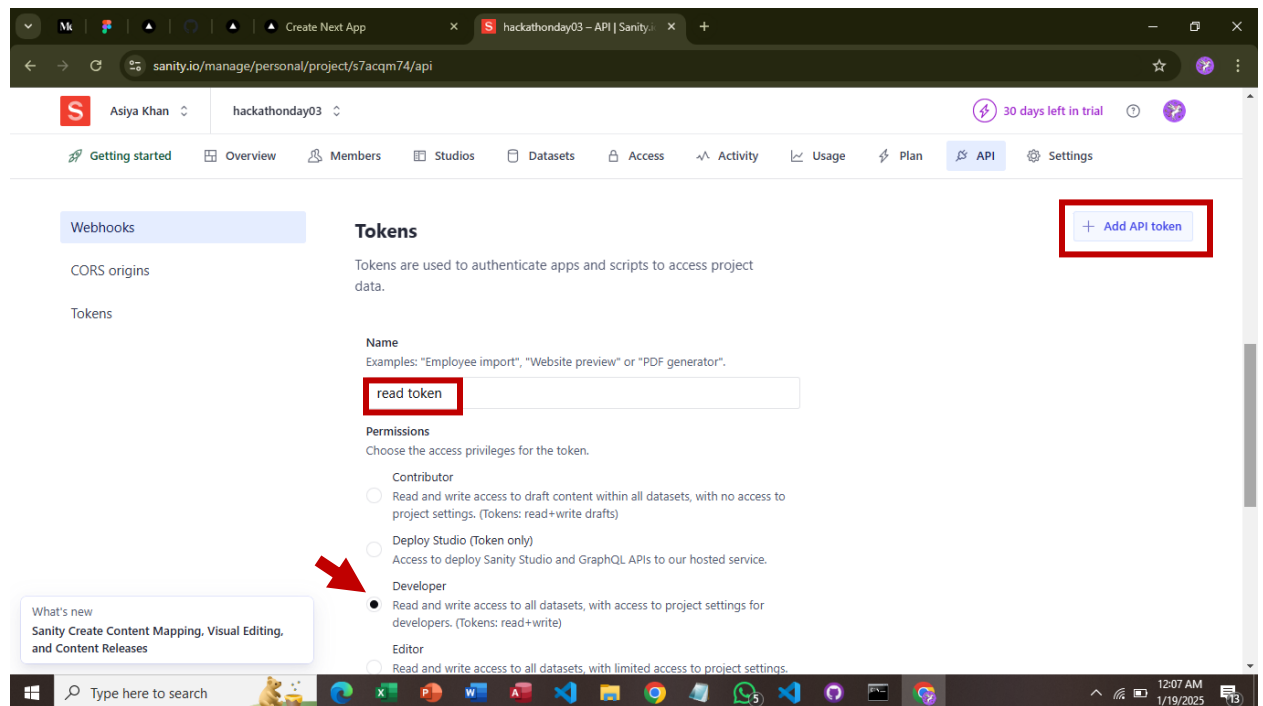
➤ Step 3 : Go to sanity Dashboard here you have your Project.



Step 4: Get you API from sanity to set your env file



- Step 5: Scroll down click on Add API token write read token and select developer and click save



➤ Step 6: Now copy your Project id and data set and API token

The image shows a sequence of three screenshots from the Sanity.io web interface, illustrating the steps to find project and dataset information and generate an API token.

**Top Screenshot:** The Sanity.io dashboard for user 'Asiya Khan' and project 'hackathonday03'. The project is on a 'Growth Trial' plan and is 'Active'. The 'PROJECT ID' is 's7acqm74', which is highlighted with a red box.

**Middle Screenshot:** The 'Datasets' page for the same project. It shows a table with one dataset named 'production', which is highlighted with a red box. The visibility is set to 'public'.

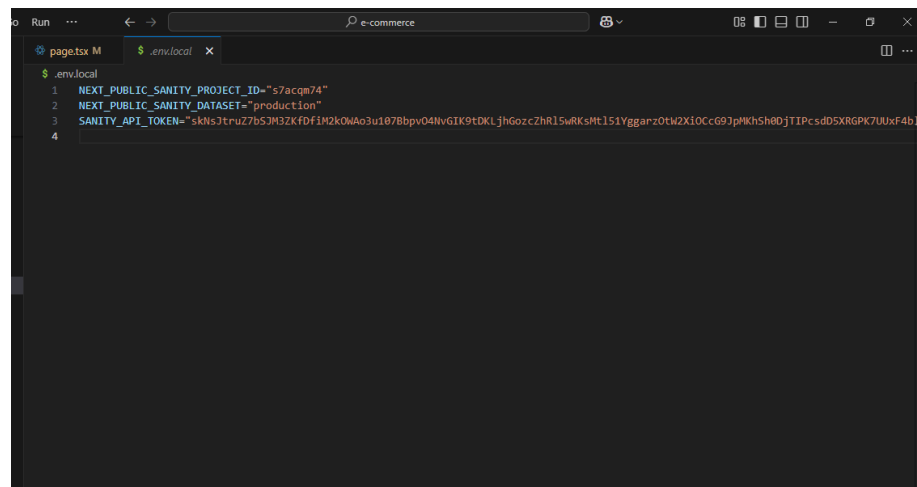
**Bottom Screenshot:** The 'API' page, which provides instructions on using tokens. It shows a table with one token named 'read token' with 'Developer' permissions, created 'just now'. Below the table, a yellow box contains the instruction 'Copy the token below – this is your only chance to do so!'. The token itself is a long alphanumeric string, highlighted with a red box.

NAME	PERMISSIONS	CREATED
read token	Developer	just now

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

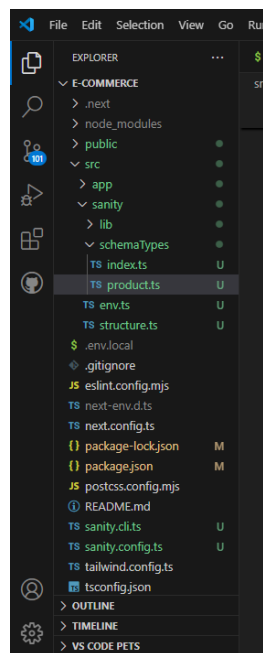
skNsJtruZ7b5JM3ZKfDfiM2kOWAo3u107BbpvO4NvGIK9tDKL-jhGozcZhR15wRKsMt151Yggan-z0tW2X10ccG9JpMKhSh0DjTIPcsdD5XRGPK7UuxF4b1Kh3L83BtFkheIVsa4pLKxSCmrZ4X07Hk0dI9jfbNHjaE1aQ1jiTfHNaX2cE7Bxu

- Step 7: Paste API Token, Project id, Dataset in your env.local file.

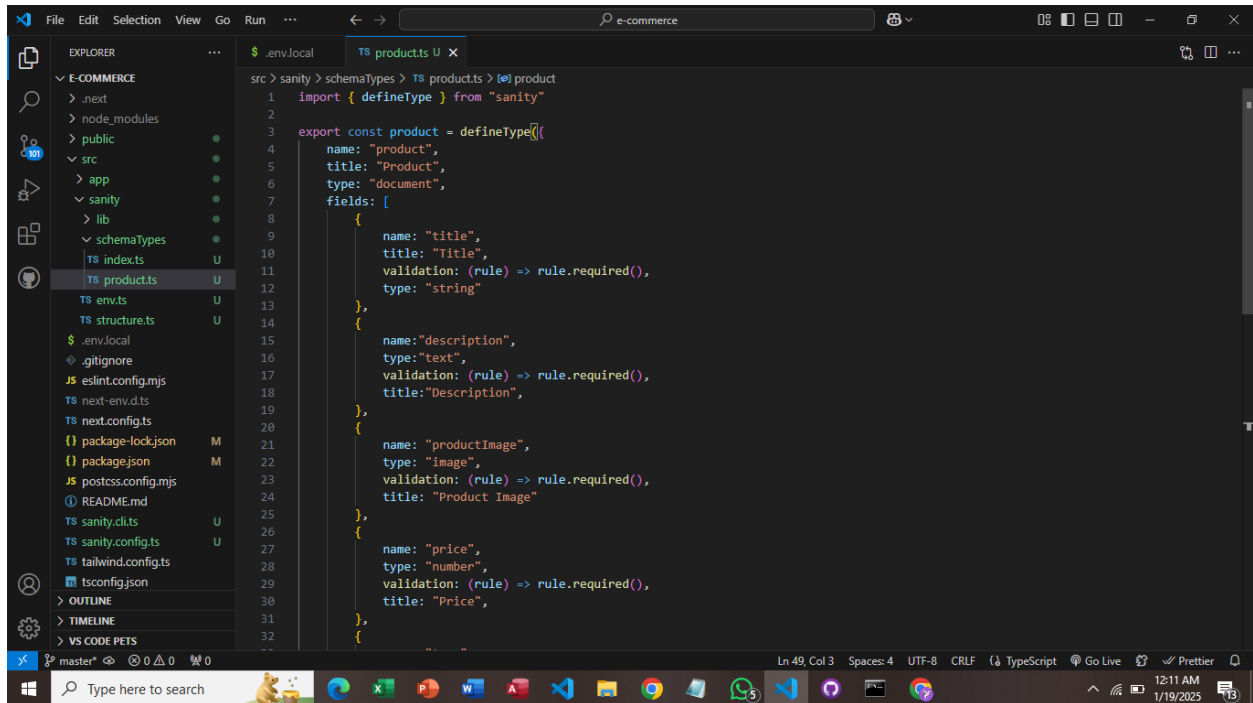


```
$.env.local
1 NEXT_PUBLIC_SANITY_PROJECT_ID="57acq874"
2 NEXT_PUBLIC_SANITY_DATASET="production"
3 SANITY_API_TOKEN="skNs3truZ7b5JH3ZKfDFiM2k0MAo3u1078bpv04NvGtK9tDKLjhg0zc2hR15wRksMt151Yggar20tW2XIOcc69JpMh5Sh80JTIPcsdD5XR6PK7UUXF4b1K"
4
```

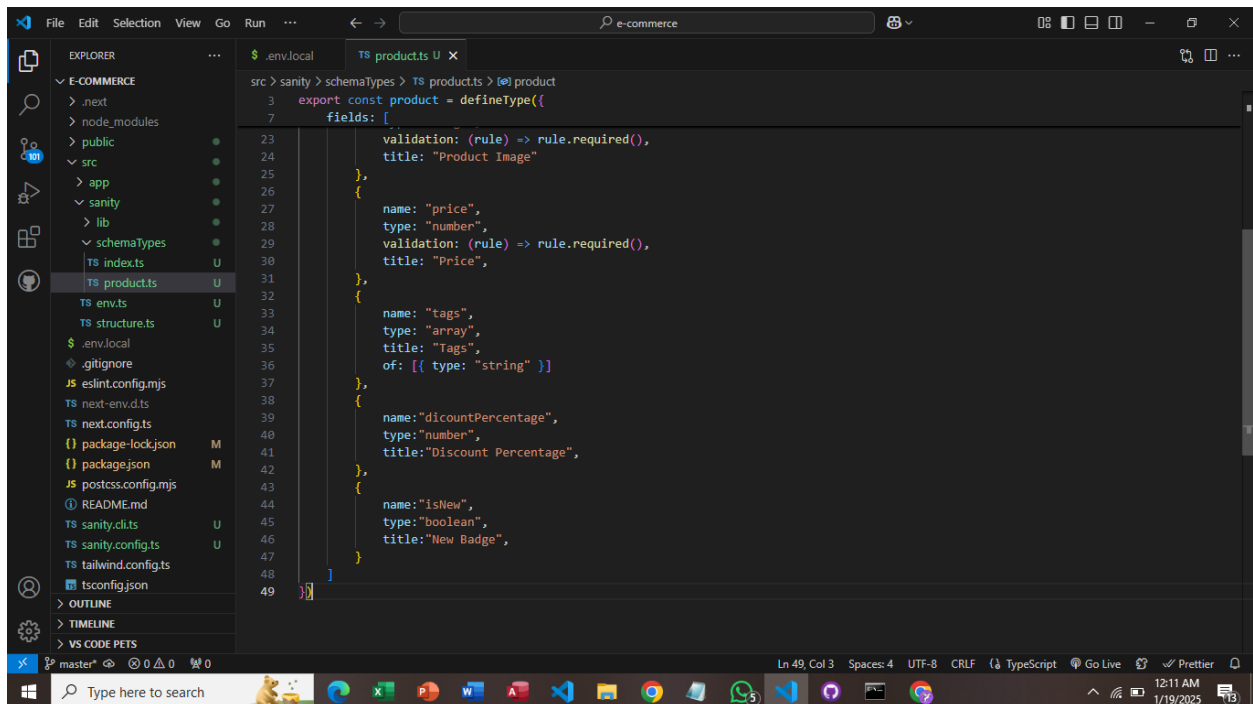
- Step 8: Now make a file product.ts in SchemaTypes folder



➤ Step 9: After making file copy paste code Provided in github repo.

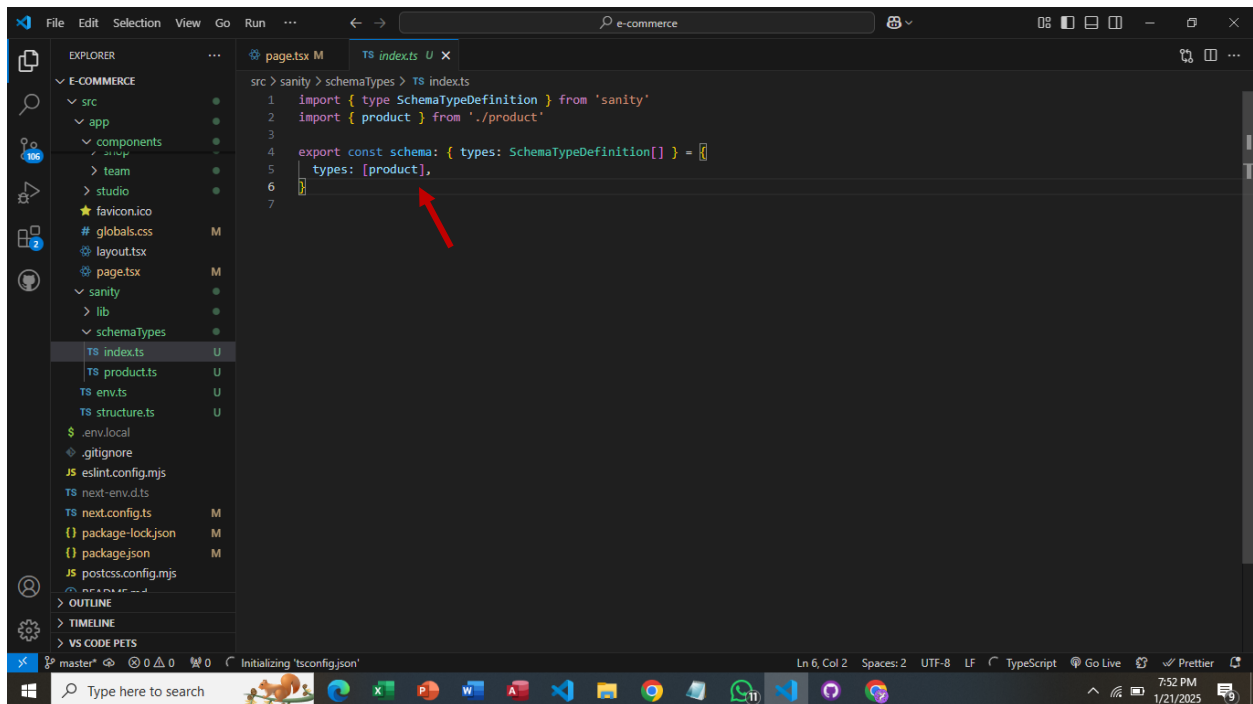


```
src > sanity > schemaTypes > TS products > product
1  import { defineType } from "sanity"
2
3  export const product = defineType({
4    name: "product",
5    title: "Product",
6    type: "document",
7    fields: [
8      {
9        name: "title",
10       title: "Title",
11       validation: (rule) => rule.required(),
12       type: "string"
13     },
14     {
15       name: "description",
16       type: "text",
17       validation: (rule) => rule.required(),
18       title: "Description",
19     },
20     {
21       name: "productImage",
22       type: "image",
23       validation: (rule) => rule.required(),
24       title: "Product Image"
25     },
26     {
27       name: "price",
28       type: "number",
29       validation: (rule) => rule.required(),
30       title: "Price",
31     },
32   ],
33 })
```



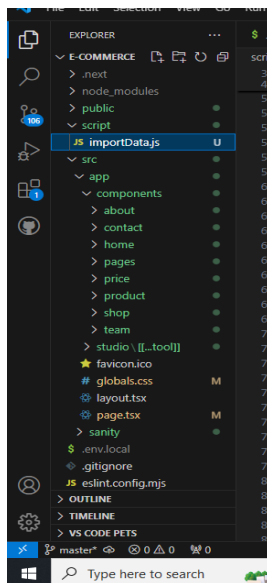
```
src > sanity > schemaTypes > TS products > product
3  export const product = defineType({
7    fields: [
23     validation: (rule) => rule.required(),
24     title: "Product Image"
25   ],
26   {
27     name: "price",
28     type: "number",
29     validation: (rule) => rule.required(),
30     title: "Price",
31   },
32   {
33     name: "tags",
34     type: "array",
35     title: "Tags",
36     of: [{ type: "string" }]
37   },
38   {
39     name: "discountPercentage",
40     type: "number",
41     title: "Discount Percentage",
42   },
43   {
44     name: "isNew",
45     type: "boolean",
46     title: "New Badge",
47   }
48 ]
49 })
```

➤ Step 10: Now go to in index.ts file and call product function



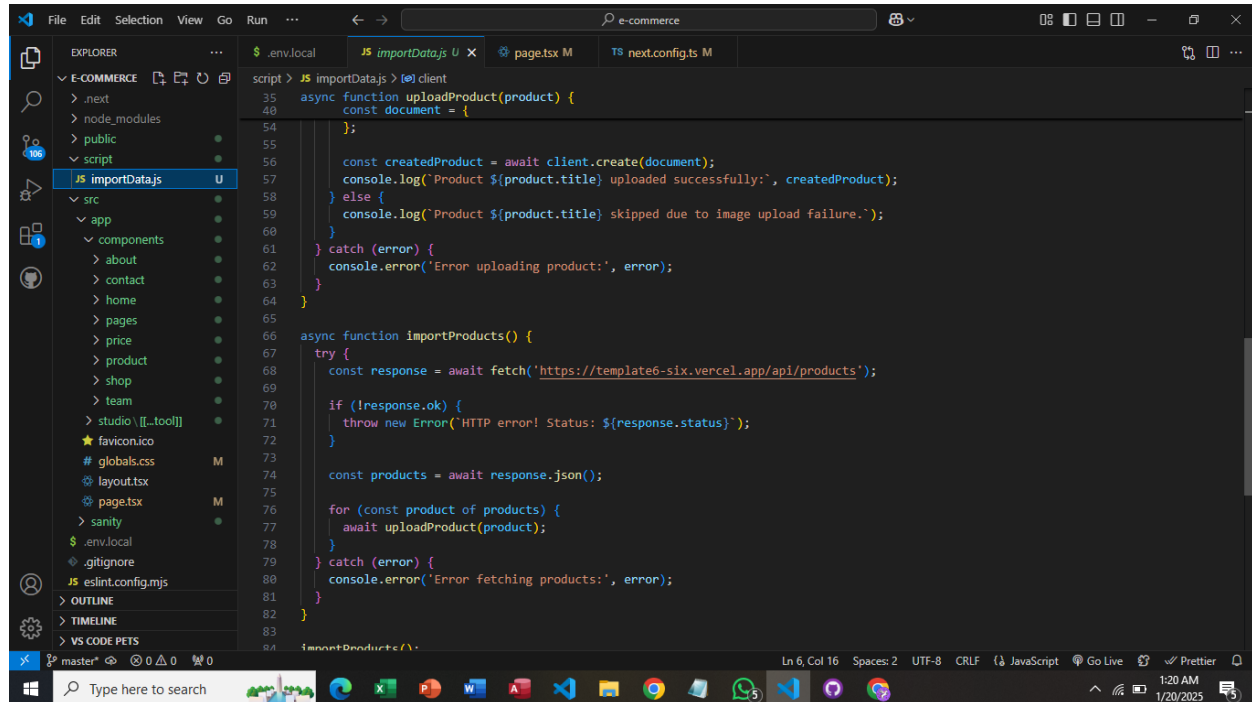
```
src > sanity > schemaTypes > TS index.ts
1 import { type SchemaTypeDefinition } from 'sanity'
2 import { product } from './product'
3
4 export const schema: { types: SchemaTypeDefinition[] } = {
5   types: [product],
6 }
7
```

➤ Step 11: Now make a folder name script and in that folder make a file name importData.js



Step 12: copy paste code provided in github repo for importData.js

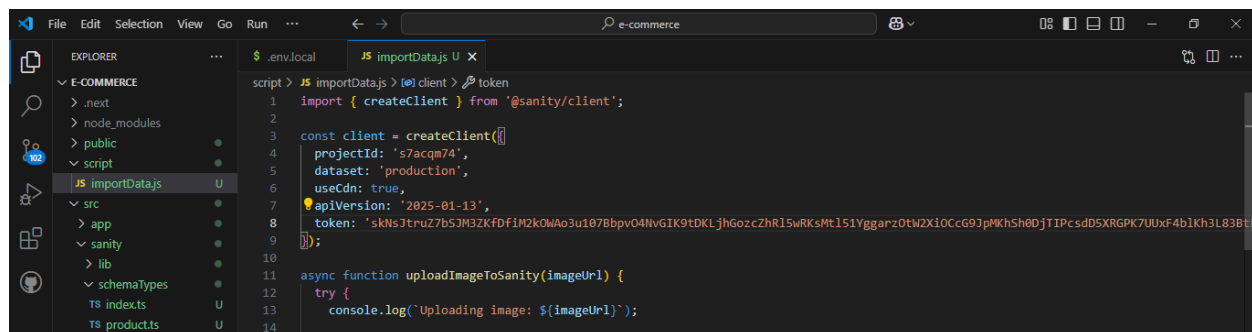
(make sure to cross check that your API is same according to your template)



The screenshot shows the VS Code editor with the 'e-commerce' project open. The Explorer sidebar on the left shows the file structure, with 'importData.js' selected under the 'script' directory. The main editor area displays the code for 'importData.js'. The code includes an 'uploadProduct' function that uses a 'client' to create a product and an 'importProducts' function that fetches products from a Vercel API and uploads them. The code is as follows:

```
script > JS importData.js U X
35 async function uploadProduct(product) {
40   const document = {
54   };
55
56   const createdProduct = await client.create(document);
57   console.log('Product ${product.title} uploaded successfully:', createdProduct);
58 } else {
59   console.log('Product ${product.title} skipped due to image upload failure.');
```

- Step 13: Now make changes in your code of importData file (copy projectId, dataset, Api token from env file enter paste it.)



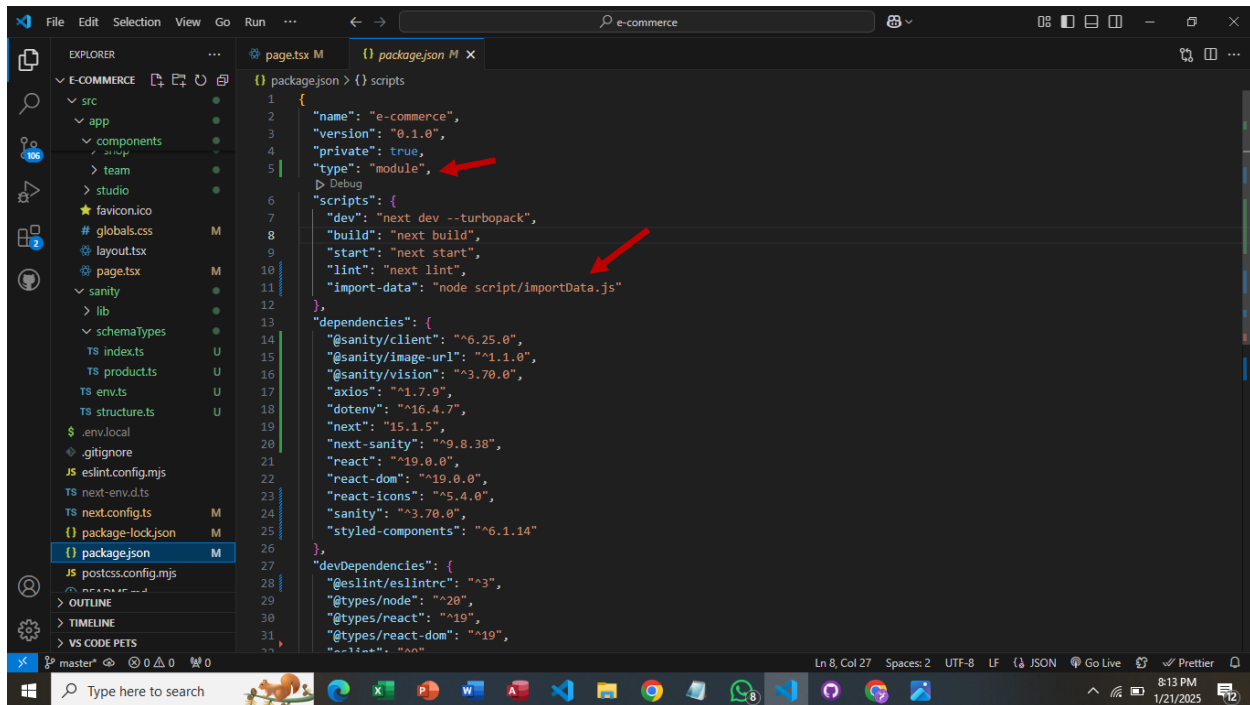
The screenshot shows the VS Code editor with the 'e-commerce' project open. The Explorer sidebar on the left shows the file structure, with 'importData.js' selected under the 'script' directory. The main editor area displays the code for 'importData.js'. The code includes an 'uploadImageToSanity' function that uses a 'client' to upload an image. The code is as follows:

```
script > JS importData.js U X
1 import { createClient } from '@sanity/client';
2
3 const client = createClient({
4   projectId: 's7acqm74',
5   dataset: 'production',
6   useCdn: true,
7   apiVersion: '2025-01-13',
8   token: 'skNsJtruz7b5JM3ZKfDfIM2kOWAo3u1078bpvO4NvGIK9tDKLjhGozcZhr15wRKsMt151Ygganz0tW2X10CcG9JpMKhSh0DjTIPcsdD5XRGPk7UUXf4b1Kh3L83BtF
9 });
10
11 async function uploadImageToSanity(imageUrl) {
12   try {
13     console.log('Uploading image: ${imageUrl}');
```

- Step 12: Now run command in your terminal (npm install @sanity/client axios dotenv)

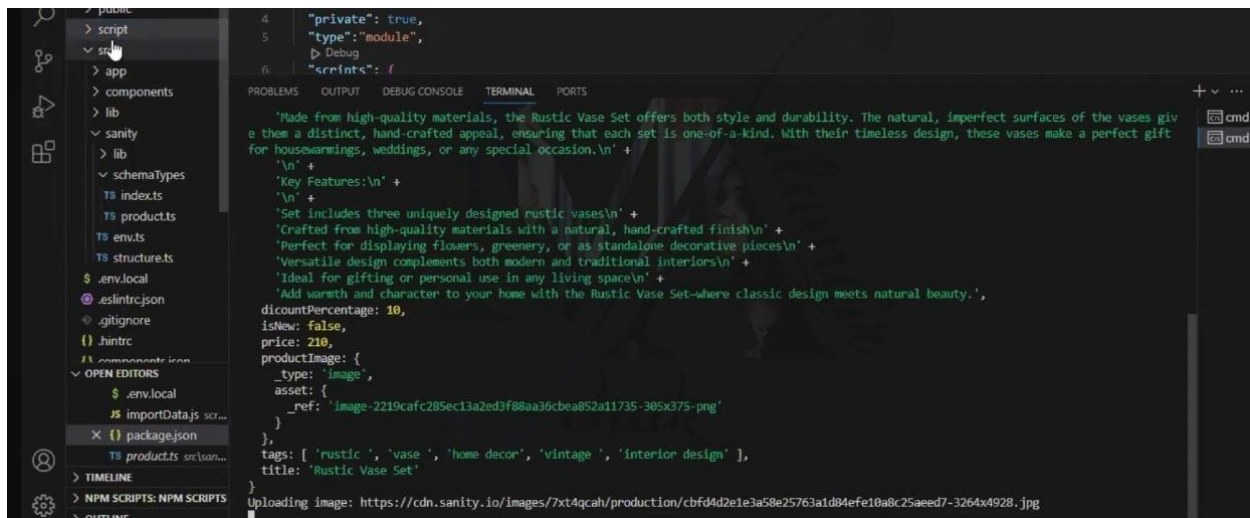


➤ Step 13: Now go to package.json file and add changes



```
{
  "name": "e-commerce",
  "version": "0.1.0",
  "private": true,
  "type": "module",
  "scripts": {
    "dev": "next dev --turbo",
    "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",
    "dotenv": "^16.4.7",
    "next": "15.1.5",
    "next-sanity": "^9.8.38",
    "react": "^19.0.0",
    "react-dom": "^19.0.0",
    "react-icons": "^5.4.0",
    "sanity": "^3.70.0",
    "styled-components": "^6.1.14"
  },
  "devDependencies": {
    "@eslint/eslintrc": "^3",
    "@types/node": "^20",
    "@types/react": "^19",
    "@types/react-dom": "^19",
    "eslint": "^8",
    "eslint-config-next": "15.1.5",
    "eslint-config-prettier": "^9",
    "eslint-plugin-react": "^7",
    "eslint-plugin-react-hooks": "^4",
    "eslint-plugin-styled-components": "^4",
    "prettier": "^3",
    "typescript": "^5"
  }
}
```

Step 14: Now run a command in your terminal  
(npm run import-data)



```
import { client } from '@sanity/client'
import dotenv from 'dotenv'
import fs from 'fs'
import path from 'path'

dotenv.config()

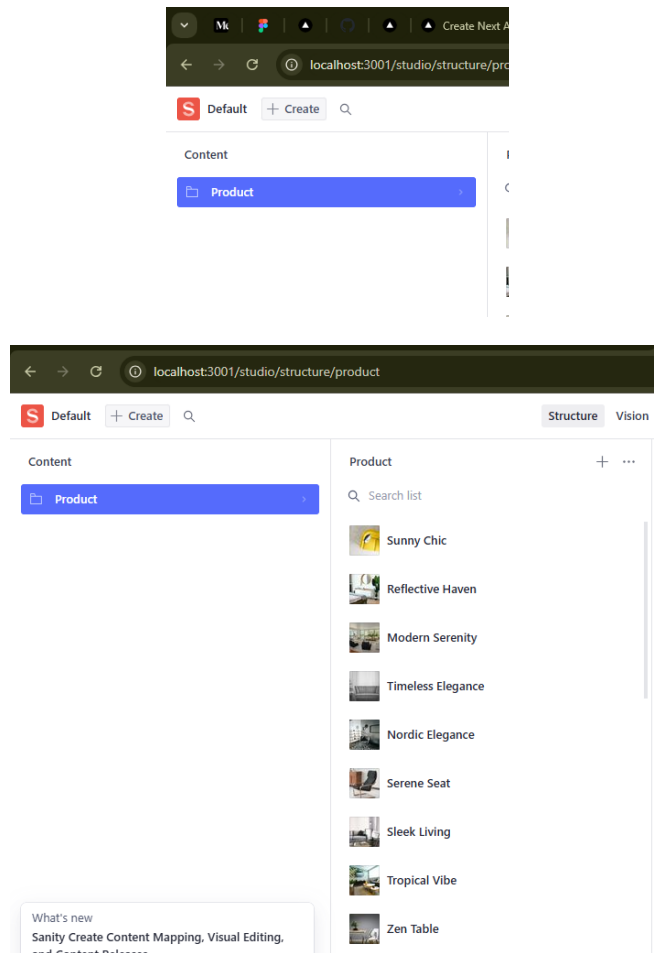
const client = client({
  projectId: process.env.SANITY_PROJECT_ID,
  dataset: process.env.SANITY_DATASET,
  token: process.env.SANITY_TOKEN,
})

const products = [
  {
    title: 'Rustic Vase Set',
    description: 'Made from high-quality materials, the Rustic Vase Set offers both style and durability. The natural, imperfect surfaces of the vases give them a distinct, hand-crafted appeal, ensuring that each set is one-of-a-kind. With their timeless design, these vases make a perfect gift for housewarmings, weddings, or any special occasion.',
    price: 210,
    discountPercentage: 10,
    tags: ['rustic', 'vase', 'home decor', 'vintage', 'interior design'],
    productImage: {
      _type: 'image',
      asset: {
        _ref: 'image-2219cafc285ec13a2ed3f88aa36cbea852a11735-385x375-png'
      }
    }
  }
]

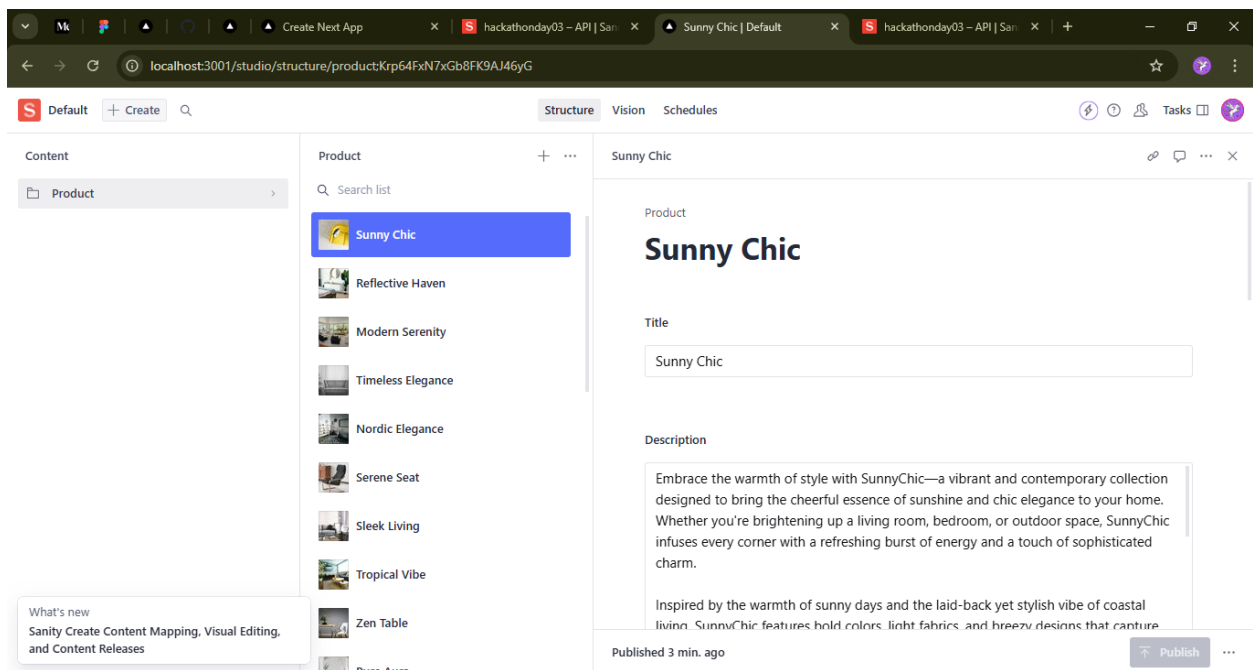
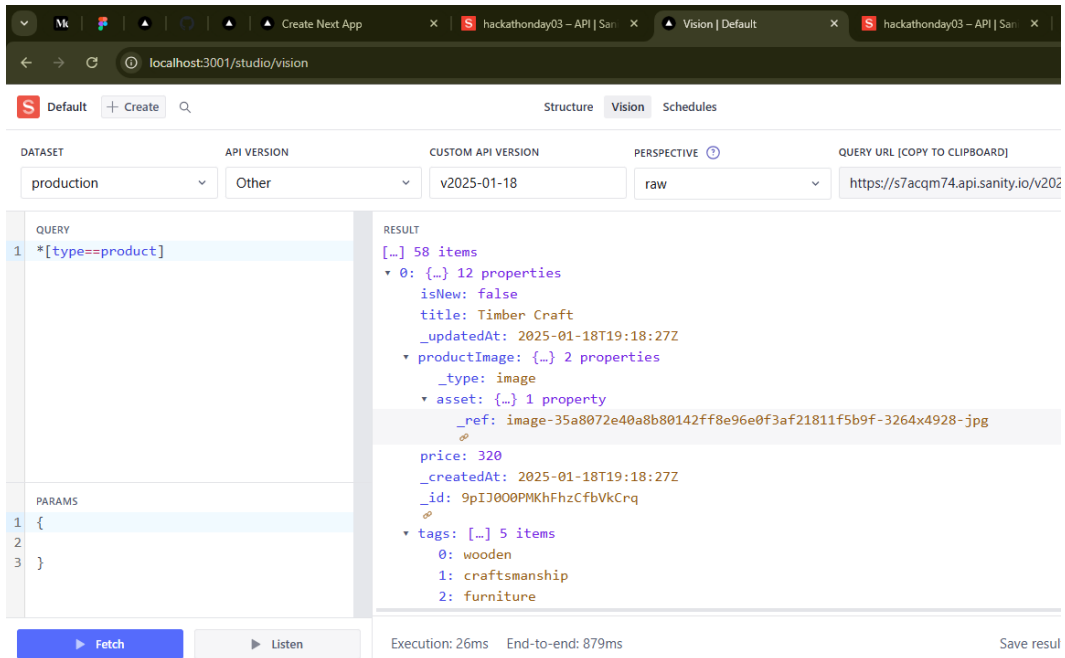
client.createMany({
  data: products
})

console.log('Products imported successfully')
```

Step 15: After all data import run a command (npm run dev) open local host write /studio after local host url open your sanity studio click on products here you have your imported data.



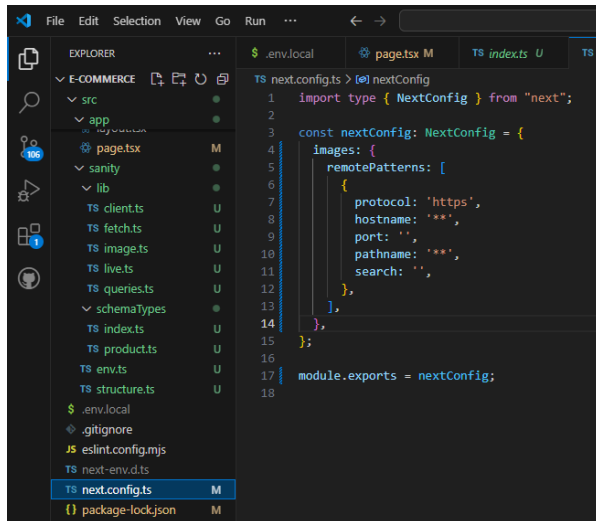
Checking product with qroc query



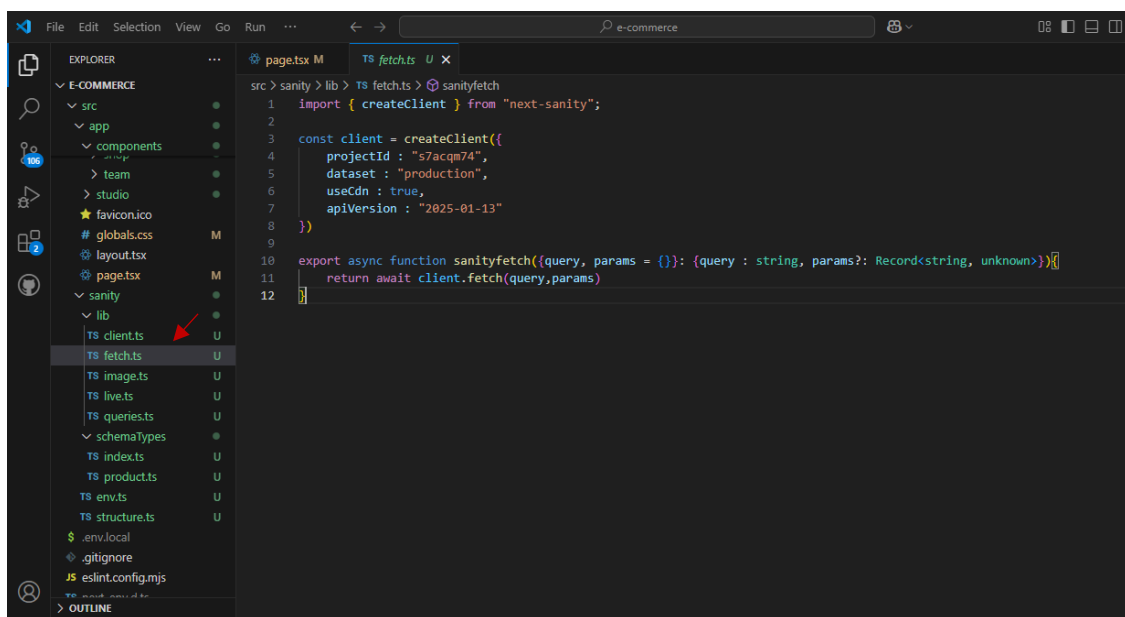
## “Here we Migrated our Data from API to Sanity”

Now we have to display this data to Ui in our local host.

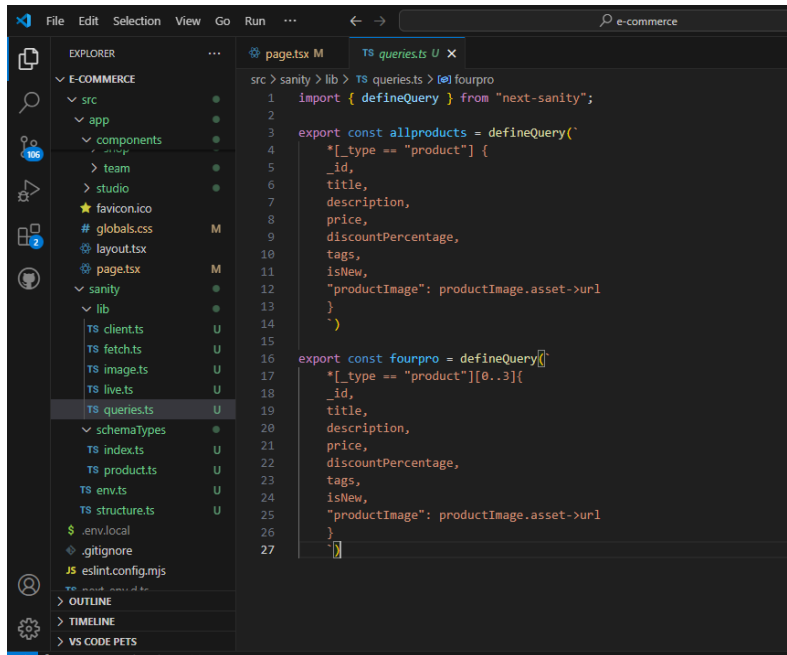
➤ Step 1: Make changes in config.ts file.



➤ Step 2: Make a file in lib folder name fetch.ts



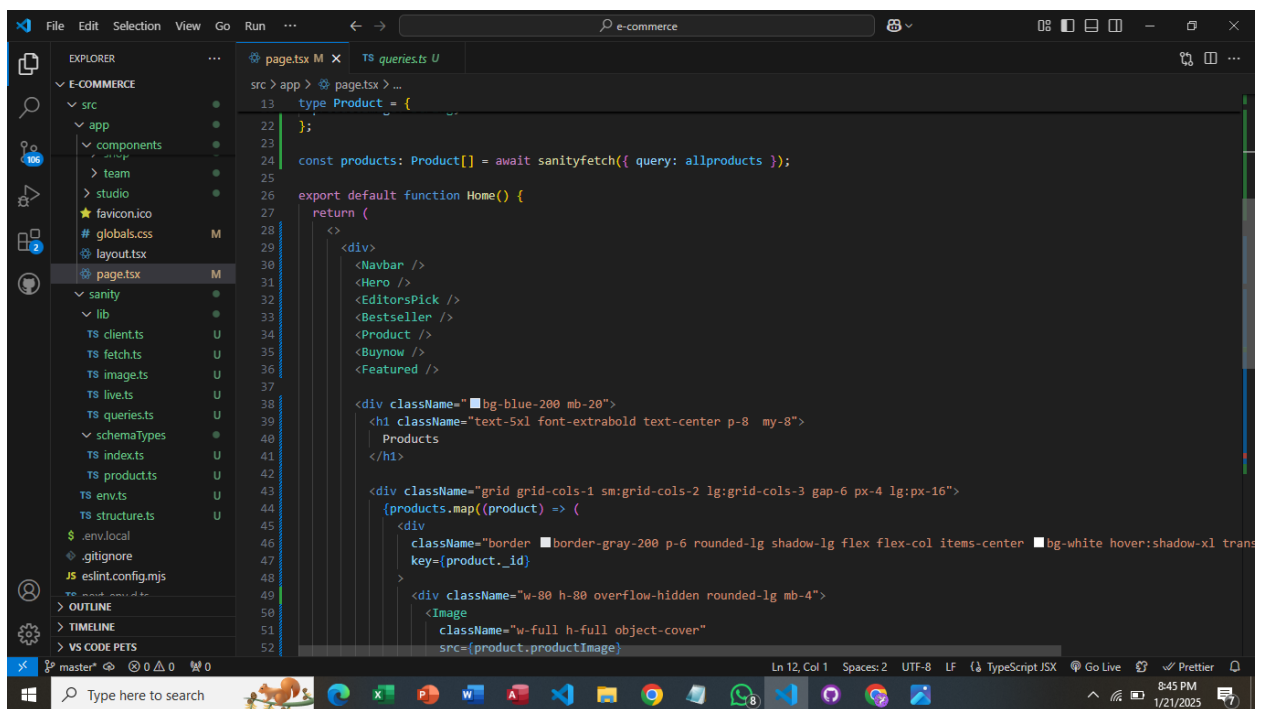
➤ Step 3: Make another file in lib name queries.ts



The screenshot shows the VS Code interface with the Explorer sidebar on the left. The file explorer is expanded to show the 'src' directory, which contains a 'sanity' folder. Inside 'sanity', there is a 'lib' folder containing several TypeScript files: 'client.ts', 'fetch.ts', 'image.ts', 'live.ts', 'queries.ts', 'schemaTypes', 'index.ts', 'products.ts', 'env.ts', 'structure.ts', and 'env.local'. The 'queries.ts' file is selected and its content is displayed in the main editor. The code defines two GraphQL queries: 'allproducts' and 'fourpro'.

```
src > sanity > lib > TS queries > fourpro
1  import { defineQuery } from "next-sanity";
2
3
4  export const allproducts = defineQuery(`
5    *[_type == "product"] {
6      _id,
7      title,
8      description,
9      price,
10     discountPercentage,
11     tags,
12     isNew,
13     "productImage": productImage.asset->url
14   }
15 `)
16
17 export const fourpro = defineQuery(`
18   *[_type == "product"][0..3]{
19     _id,
20     title,
21     description,
22     price,
23     discountPercentage,
24     tags,
25     isNew,
26     "productImage": productImage.asset->url
27   }
28 `)
```

- Step 4: call products in your page.tsx and style products cards.



The screenshot shows the VS Code interface with the Explorer sidebar on the left. The file explorer is expanded to show the 'src' directory, which contains a 'page.tsx' file. The 'page.tsx' file is selected and its content is displayed in the main editor. The code defines a 'Product' type, fetches product data from a Sanity API, and renders a list of product cards in a grid layout.

```
src > app > page.tsx > ...
13  type Product = {
14    _id: string;
15    title: string;
16    description: string;
17    price: number;
18    discountPercentage: number;
19    tags: string[];
20    isNew: boolean;
21    productImage: string;
22  };
23
24  const products: Product[] = await sanityFetch({ query: allproducts });
25
26  export default function Home() {
27    return (
28      <div>
29        <Navbar />
30        <Hero />
31        <EditorsPick />
32        <Bestseller />
33        <Product />
34        <Buynow />
35        <Featured />
36
37        <div className="bg-blue-200 mb-20">
38          <h1 className="text-5xl font-extrabold text-center p-8 my-8">
39            Products
40          </h1>
41
42          <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-6 px-4 lg:px-16">
43            {products.map((product) => (
44              <div
45                className="border border-gray-200 p-6 rounded-lg shadow-lg flex flex-col items-center bg-white hover:shadow-xl transition"
46                key={product._id}
47              >
48                <div className="w-80 h-80 overflow-hidden rounded-lg mb-4">
49                  <Image
50                    className="w-full h-full object-cover"
51                    src={product.productImage}
52                  />
53                <div>
54                  <h3>{product.title}</h3>
55                  <p>{product.description}</p>
56                  <p>{product.price}</p>
57                  <p>{product.discountPercentage}</p>
58                  <p>{product.tags}</p>
59                </div>
60              </div>
61            ))}
62          </div>
63        </div>
64      </div>
65    );
66  }
67}
```




➤ Step 5: Refresh your local Host.


Create Next App

localhost:3000


# Products




**Timber Craft**  
Price: 320 Rs.




**Cloud Haven Chair**  
Price: 230 Rs.



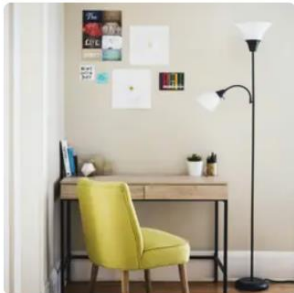
**The Dandy chair**  
Price: 150 Rs.




**Sleek Living**  
Price: 300 Rs.




**Serene Seat**  
Price: 350 Rs.



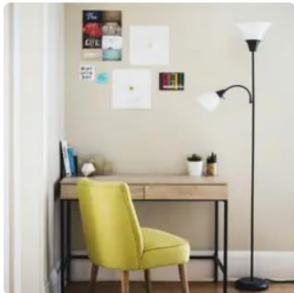
**Nordic Elegance**  
Price: 280 Rs.



**Reflective Haven**  
Price: 300 Rs.




**Sunny Chic**  
Price: 400 Rs.



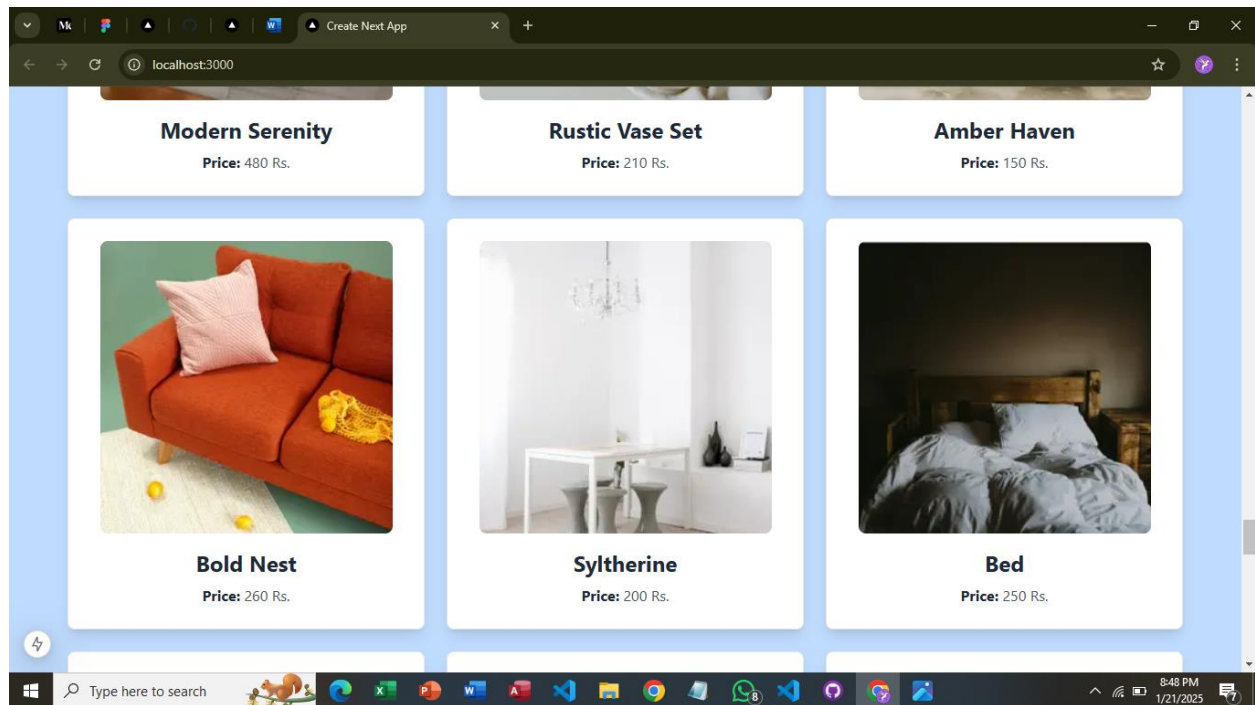
**Bright Space**  
Price: 180 Rs.

Type here to search



8:48 PM  
1/21/2025





“Here is products Fetch & Displayed on Ui local host.”

Documents by Asiya Khan