

Task- 3 Api Integration and Data Migration

website Name: comforty

date: 18-jan-2025

author:Samia javed

overveiw:

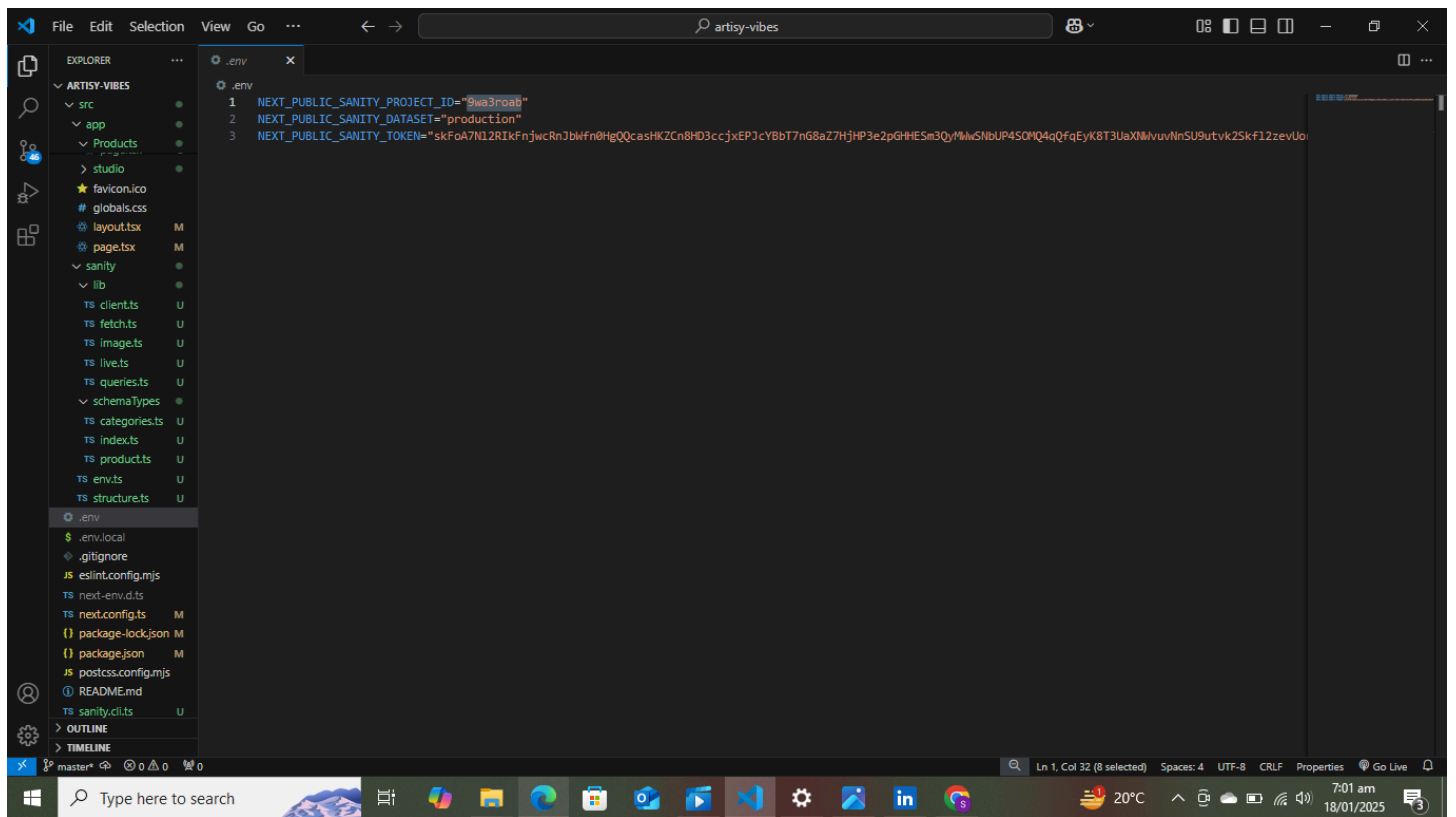
Day-3 focusing on API Integration and data migration get data from external api and display on UI and creating dynamic data. here's a summary.

Summary:

- >setup sanity with nextjs.
- >create product and catagory schema
- >migrate data from external api to sanity CMS.
- >test and validate the integration to ensure that the data is smoothly working.

Setup sanity:

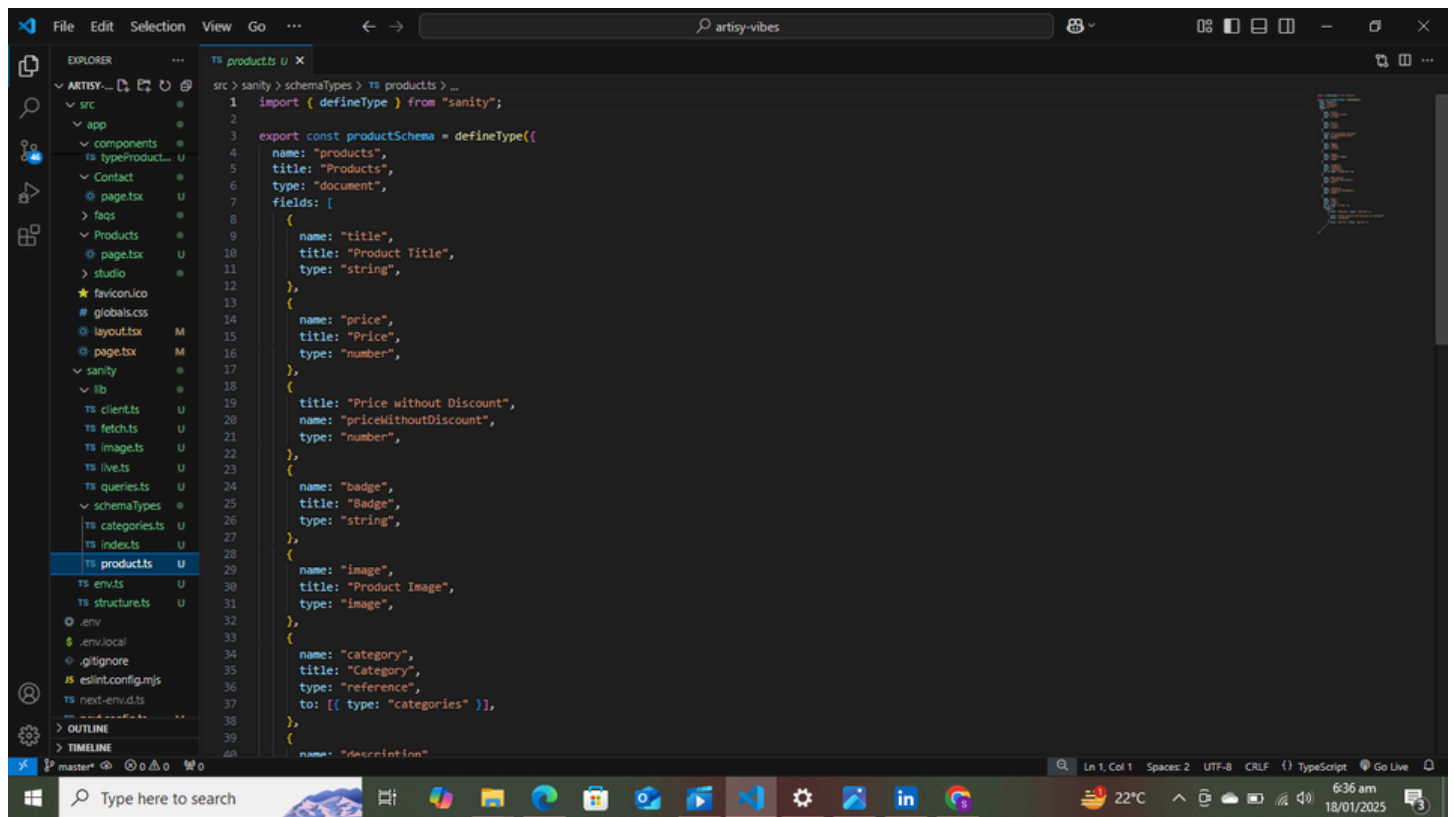
- >the code can be connected with sanity api by configured data set.
- >add project Id, Dataset and token.
- >use .env file to store sensitive data.



Migration:

migrate data from given api and make schema of categories and products.

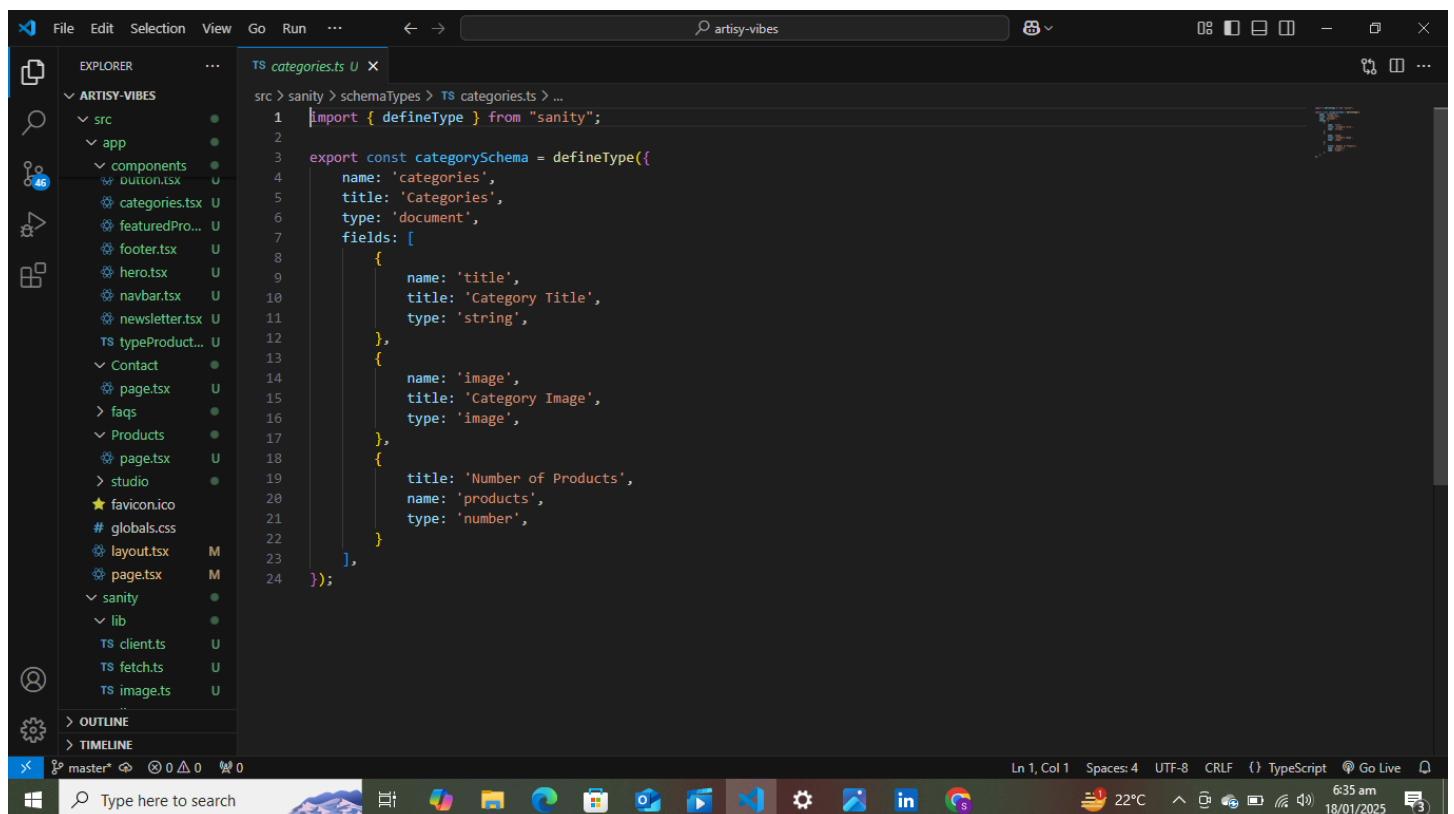
schema:



This screenshot shows the Visual Studio Code editor with the file `products.ts` open. The Explorer sidebar on the left shows the project structure for `ARTISY-VIBES`, with `src > sanity > schemasTypes > TS products.ts` selected. The main editor displays the following TypeScript code:

```
1 import { defineType } from "sanity";
2
3 export const productSchema = defineType({
4   name: "products",
5   title: "Products",
6   type: "document",
7   fields: [
8     {
9       name: "title",
10      title: "Product Title",
11      type: "string",
12    },
13    {
14      name: "price",
15      title: "Price",
16      type: "number",
17    },
18    {
19      title: "Price without Discount",
20      name: "priceWithoutDiscount",
21      type: "number",
22    },
23    {
24      name: "badge",
25      title: "Badge",
26      type: "string",
27    },
28    {
29      name: "image",
30      title: "Product Image",
31      type: "image",
32    },
33    {
34      name: "category",
35      title: "Category",
36      type: "reference",
37      to: [{ type: "categories" }],
38    },
39  ],
40  name: "description"
41});
```

The status bar at the bottom indicates the cursor is at line 1, column 1, with 2 spaces, in UTF-8 encoding with CRLF line endings, using the TypeScript language.

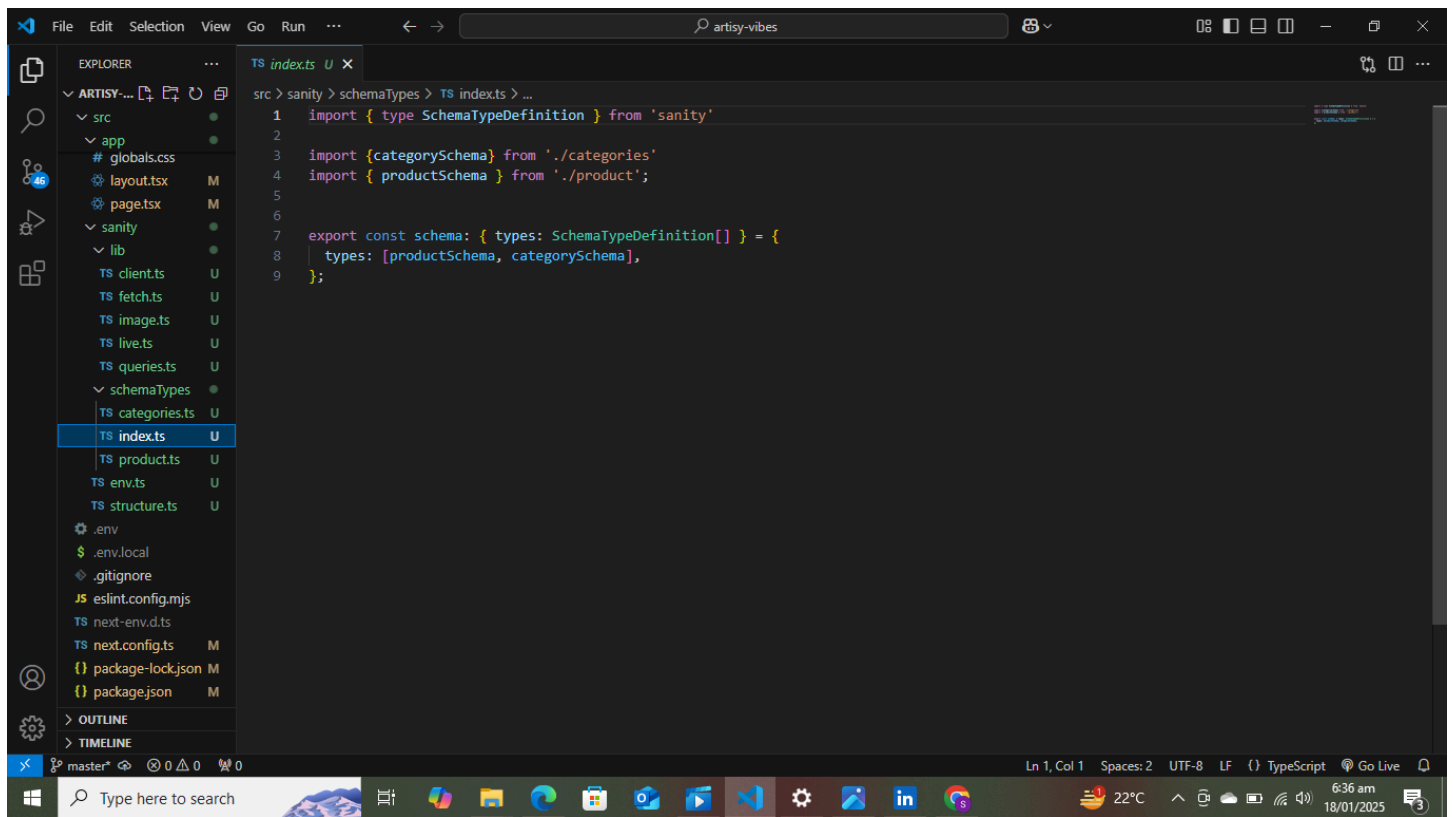


This screenshot shows the Visual Studio Code editor with the file `categories.ts` open. The Explorer sidebar on the left shows the project structure for `ARTISY-VIBES`, with `src > sanity > schemasTypes > TS categories.ts` selected. The main editor displays the following TypeScript code:

```
1 import { defineType } from "sanity";
2
3 export const categorySchema = defineType({
4   name: 'categories',
5   title: 'Categories',
6   type: 'document',
7   fields: [
8     {
9       name: 'title',
10      title: 'Category Title',
11      type: 'string',
12    },
13    {
14      name: 'image',
15      title: 'Category Image',
16      type: 'image',
17    },
18    {
19      title: 'Number of Products',
20      name: 'products',
21      type: 'number',
22    },
23  ],
24 });
```

The status bar at the bottom indicates the cursor is at line 1, column 1, with 4 spaces, in UTF-8 encoding with CRLF line endings, using the TypeScript language.

import schemas in index.ts



Fetch data:

>fetch data by using groq queries

Core Concepts

1. Selection (*):

Start queries with * to select all documents or specify document types.

Filters ([]):

Use filters to narrow down the dataset based on conditions.

Projection ({}):

Shape the returned data by selecting specific fields.

References (->):

Fetch referenced documents with the arrow operator.

Functions:

Use built-in functions for advanced operations, like sorting, slicing, or manipulating strings.

Conditionals (if...else):

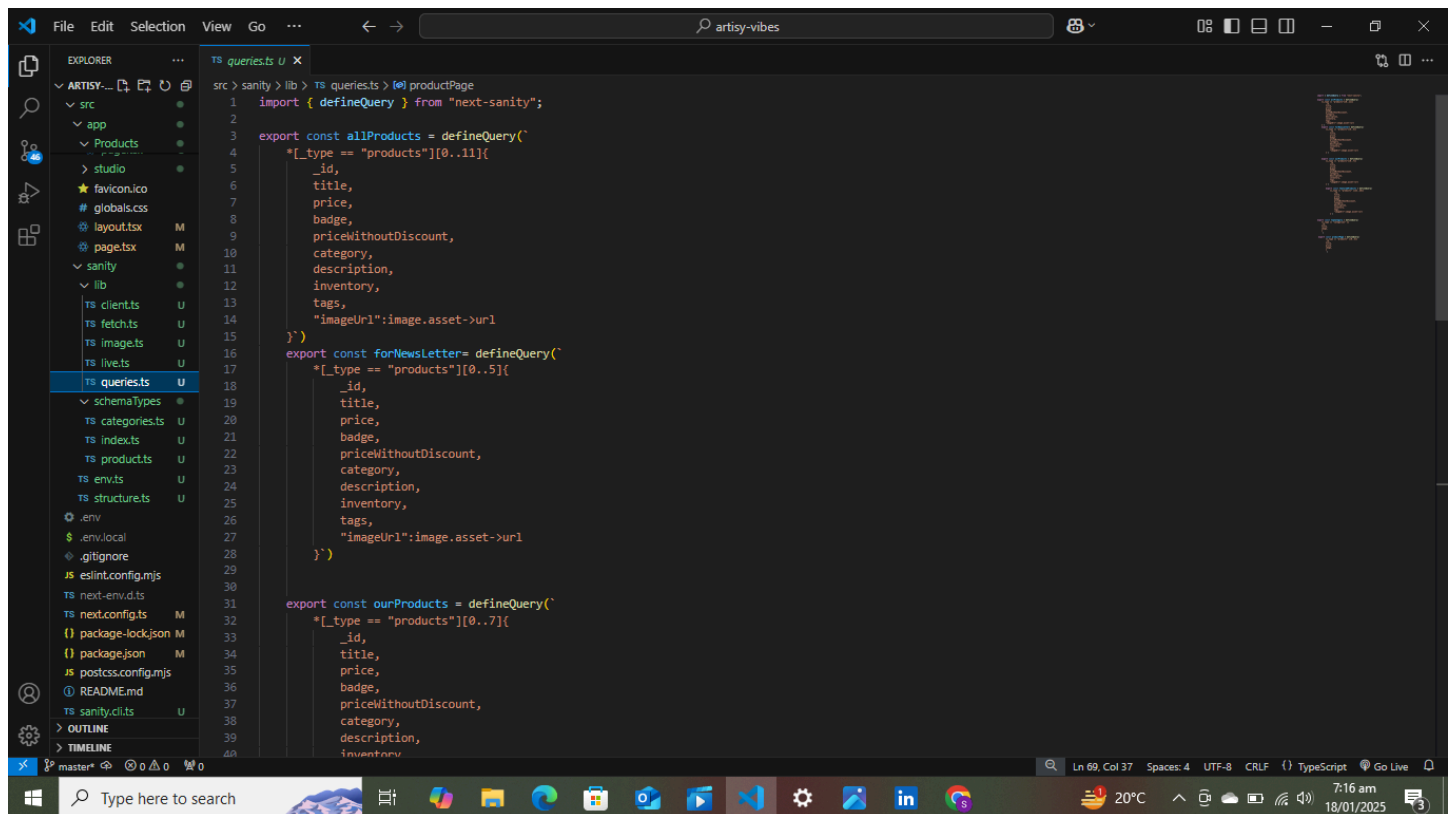
Add conditional logic to return specific values.

Aggregations:

Perform calculations like count or sum.

Chaining (!):

Pipe data through multiple operations.



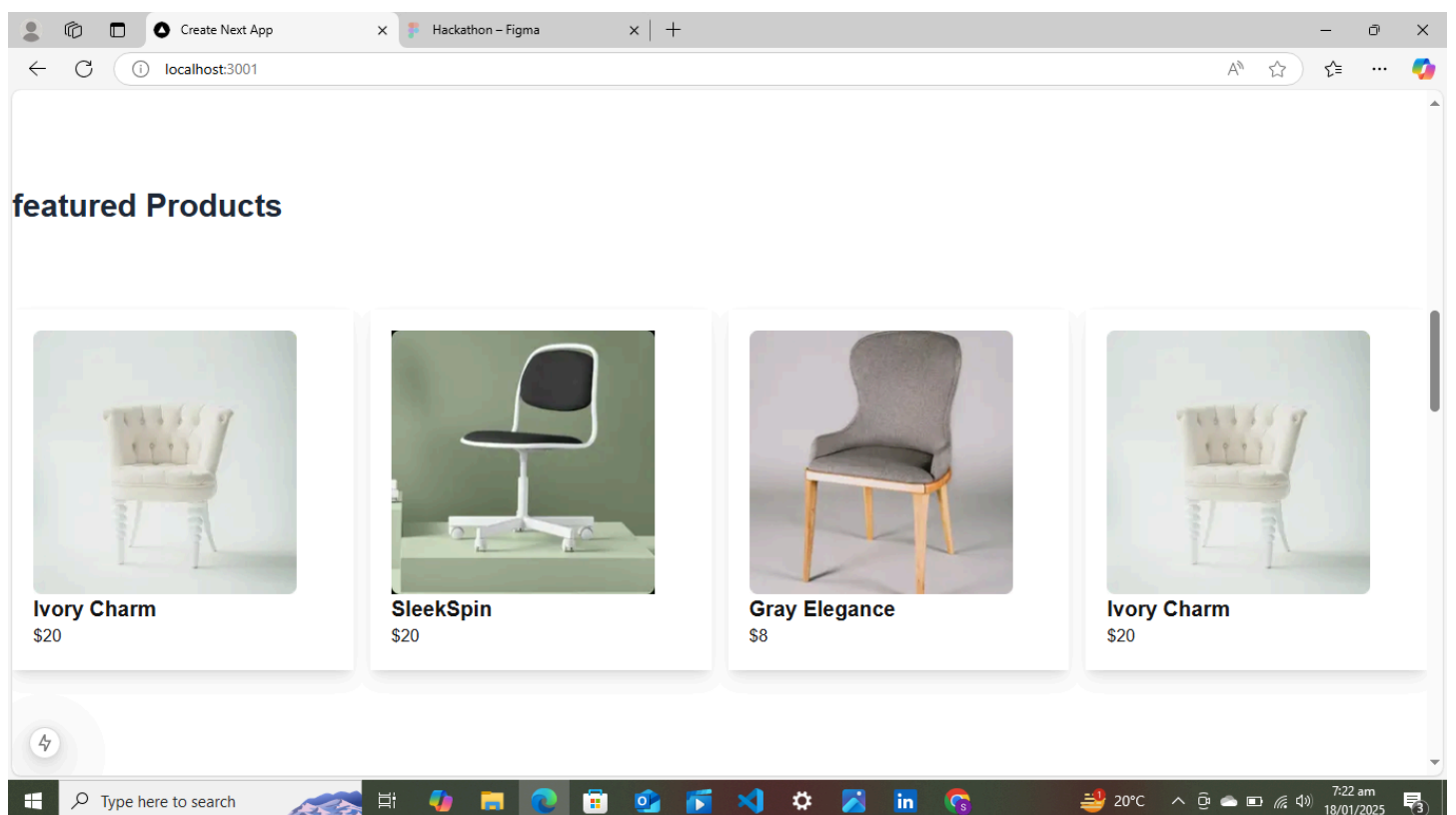
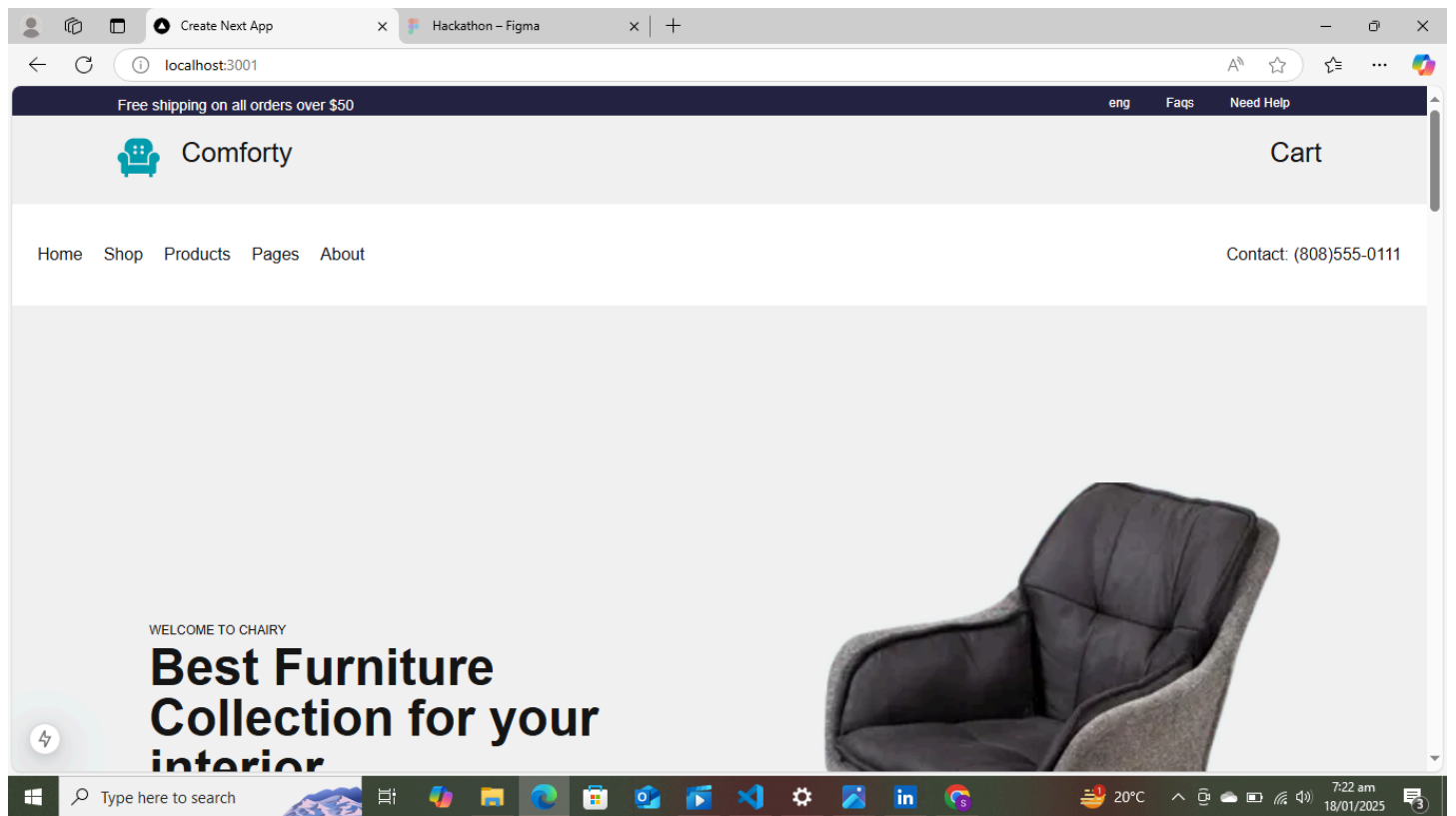
Display on UI:

components: components make our code reuseable.

style and design: for styling tailwind css is best option.

sensitive data: for storing sensitive data we use .env file.

dynamic: we build dynamic code with sanity:



conclusion : task - 3 build dynamic data > display on UI by using external api and sanity;