

The screenshot displays the Foodify application interface, which is divided into three main sections: a left sidebar, a top navigation bar, and a main content area.

**Top Navigation Bar:** Features a 'Default' tab, a '+ Create' button, and a search icon. On the right, there are links for 'Structure', 'Vision', and 'Schedules', along with a 'Tasks' section and a user profile icon.

**Left Sidebar:** Contains a 'Content' section with a 'Food' category selected. Below it, a list of food items is shown, with 'Chicken Chup' highlighted in blue. Other items include Pizza, Country Burger, Burger, Chocolate Muffin, Fresh Lime, and Chicken Chup (repeated).

**Main Content Area:** Displays the details for the 'Chicken Chup' item. The title 'Chicken Chup' is prominently shown. Below it, there are input fields for 'Food Name' (filled with 'Chicken Chup'), 'Category' (filled with 'Appetizer'), and 'Current Price' (filled with '12'). A 'Published 59 min. ago' timestamp is visible at the bottom left of the main content area.

**Bottom Section:** This section shows the 'Chicken Chup' item details again, but with additional fields. The 'Current Price' is '12', and the 'Original Price' is '15'. There is a 'Tags' section with 'Sell' and 'Crispy' tags. The 'Food Image' section shows a placeholder image of a pizza. The timestamp at the bottom is 'Published 1 hr. ago'.

Default

+ Create

StructureVisionSchedules

Content

Food

Chef

Chef

+ ...

Tahmina Rumi

Search list

William Rumi

Bisnu Devgon

Munna Kathy

M. Mohammad

Jorina Begum

Tahmina Rumi

William Rumi

Bisnu Devgon

Munna Kathy

M. Mohammad

Jorina Begum

Tahmina Rumi

Chef

Tahmina Rumi

Chef Name

Tahmina Rumi

Position

Role or title of the chef (e.g., Head Chef, Sous Chef)

Head Chef

Years of Experience

Number of years the chef has worked in the culinary field

12

Published 1 hr. ago

Publish

Default

+ Create

StructureVisionSchedules

Content

Food

Chef

Chef

+ ...

Tahmina Rumi

Search list

William Rumi

Bisnu Devgon

Munna Kathy

M. Mohammad

Jorina Begum

Tahmina Rumi

William Rumi

Bisnu Devgon

Munna Kathy

M. Mohammad

Jorina Begum


Tahmina Rumi

Specialty

Specialization of the chef (e.g., Italian Cuisine, Pastry)

Italian Cuisine

Chef Image



Description

Short bio or introduction about the chef

Expert in crafting authentic Italian dishes and pastries.

Published 1 hr. ago

Publish

Default

+ Create

StructureVisionSchedules

DATASET

production

API VERSION

Other

CUSTOM API VERSION

v2025-01-19

PERSPECTIVE

raw

QUERY URL [COPY TO CLIPBOARD]

https://zn3e0jov.api.sanity.io/v2025-01-19/data/query/productic

1

QUERY

\*[type==chef]

PARAMS

{

}

RESULT

[...] 46 items

0: {...} 13 properties

originalPrice: 50

available: true

tags: [...] 2 items

0: Cheesy

1: Vegetarian

\_id: CpL9FWqkp36faTVG3cvTY

category: Main Course

updatedAt: 2025-01-21T18:44:29Z

image: [...] 2 properties

\_type: image

asset: {...} 2 properties

\_type: reference

\_ref: image-59499bc14b7749f088d0017b8871e06a47a9deb1-1248x1068-png

\_rev: CpL9FWqkp36faTVG3cvPM

\_type: food

description: Delicious vegetarian pizza topped with fresh vegetables and cheese.

price: 43

\_createdAt: 2025-01-21T18:44:29Z

name: Pizza

1: {...} 13 properties

Fetch

Listen

Execution: 16ms End-to-end: 477ms

Save result as

JSON

CSV

Default + Create Structure Vision Schedules Tasks

DATASET: production API VERSION: Other CUSTOM API VERSION: v2025-01-19 PERSPECTIVE: raw QUERY URL [COPY TO CLIPBOARD]: https://zn3e0jov.api.sanity.io/v2025-01-19/data/query/productic

QUERY: 1 \*{type==food}

PARAMS: 1 { 2 } 3 }

RESULT: [-] 46 items

- 0: { } 13 properties
  - available: true
  - description: Delicious vegetarian pizza topped with fresh vegetables and cheese.
  - price: 43
  - \_id: Cpl9FWqkp36faTVGJcvTY
  - \_updatedAt: 2025-01-21T18:44:29Z
  - originalPrice: 50
  - \_rev: Cpl9FWqkp36faTVGJcvPM
  - \_type: food
  - name: Pizza
  - category: Main Course
  - image: { } 2 properties
    - \_type: image
    - asset: { } 2 properties
      - \_ref: image-59499bc14b7749f088d0017b8871e06a47a9deb1-1248x1068-png
      - \_type: reference
  - tags: [-] 2 items
    - 0: Cheesy
    - 1: Vegetarian
  - \_createdAt: 2025-01-21T18:44:29Z
- 1: { } 13 properties

Execution: 22ms End-to-end: 212ms

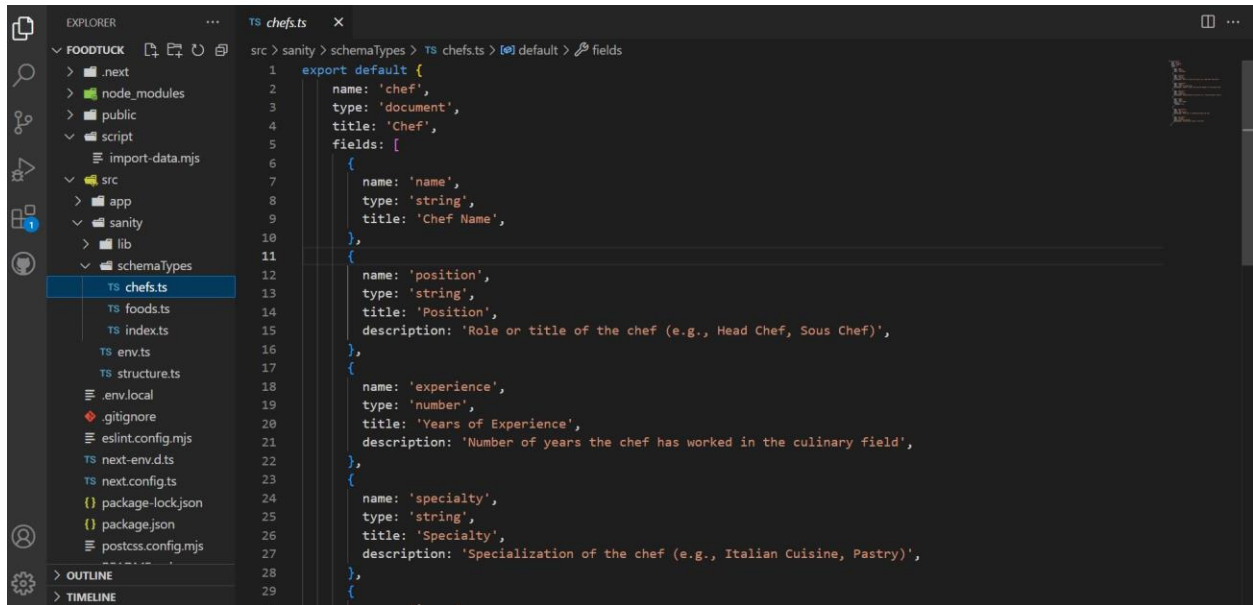
Save result as JSON CSV

EXPLORER: FOODTUCK, node\_modules, public, script, import-data.mjs, src, app, sanity, lib, schemaTypes, TS chefs.ts, TS foods.ts, TS index.ts, TS env.ts, TS structure.ts, .env.local, .gitignore, eslint.config.mjs, next-env.d.ts, next.config.ts, package-lock.json, package.json, postcss.config.mjs, OUTLINE, TIMELINE

TS foods.ts

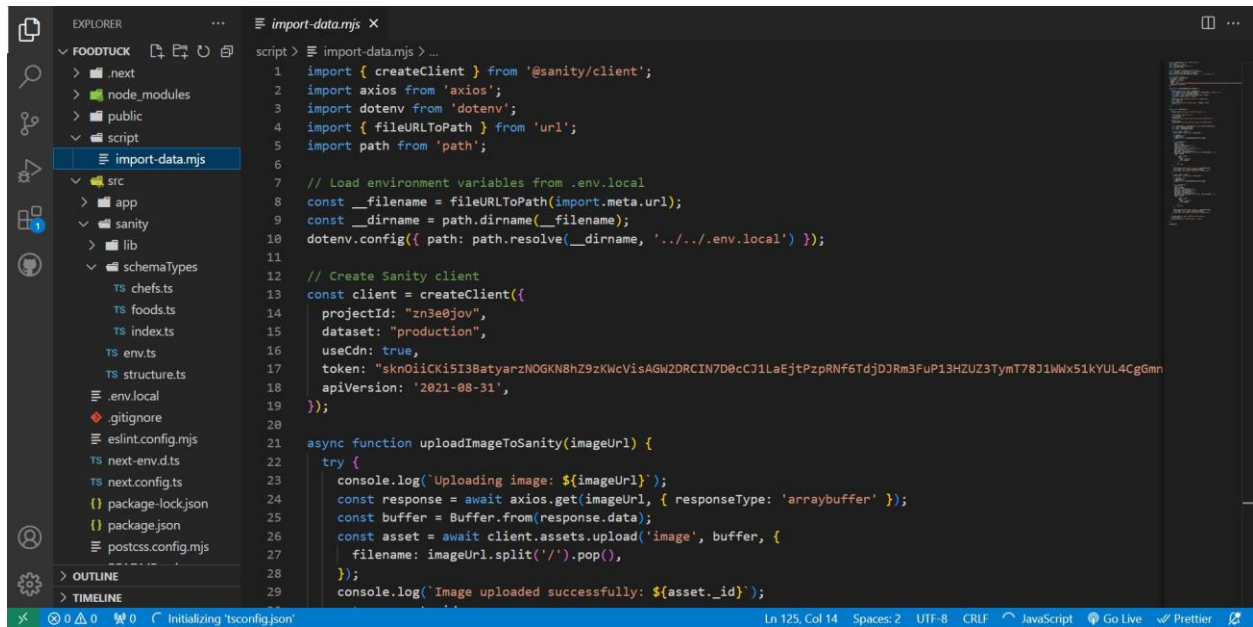
```
src > sanity > schemaTypes > TS foods.ts > default
1 export default {
2   name: 'food',
3   type: 'document',
4   title: 'Food',
5   fields: [
6     {
7       name: 'name',
8       type: 'string',
9       title: 'Food Name',
10    },
11    {
12      name: 'category',
13      type: 'string',
14      title: 'Category',
15      description:
16        'Category of the food item (e.g., Burger, Sandwich, Drink, etc.)',
17    },
18    {
19      name: 'price',
20      type: 'number',
21      title: 'Current Price',
22    },
23    {
24      name: 'originalPrice',
25      type: 'number',
26      title: 'Original Price',
27      description: 'Price before discount (if any)',
28    },
29  ],
30 }
```

Ln 60, Col 5 Spaces: 2 UTF-8 CRLF TypeScript Go Live Prettier



The screenshot shows the VS Code editor with the Explorer sidebar on the left. The 'FOODTUCK' project is expanded, showing the 'src' directory. Inside 'src', the 'schemaTypes' directory is selected, and the 'TS chefs.ts' file is open in the editor. The breadcrumb navigation at the top reads: 'src > sanity > schemaTypes > TS chefs.ts > default > fields'. The code in the editor is as follows:

```
1 export default {
2   name: 'chef',
3   type: 'document',
4   title: 'Chef',
5   fields: [
6     {
7       name: 'name',
8       type: 'string',
9       title: 'Chef Name',
10    },
11    {
12      name: 'position',
13      type: 'string',
14      title: 'Position',
15      description: 'Role or title of the chef (e.g., Head Chef, Sous Chef)',
16    },
17    {
18      name: 'experience',
19      type: 'number',
20      title: 'Years of Experience',
21      description: 'Number of years the chef has worked in the culinary field',
22    },
23    {
24      name: 'specialty',
25      type: 'string',
26      title: 'Specialty',
27      description: 'Specialization of the chef (e.g., Italian Cuisine, Pastry)',
28    },
29  ],
30 }
```



The screenshot shows the VS Code editor with the Explorer sidebar on the left. The 'FOODTUCK' project is expanded, showing the 'script' directory. The 'import-data.mjs' file is selected in the Explorer and open in the editor. The breadcrumb navigation at the top reads: 'script > import-data.mjs > ...'. The code in the editor is as follows:

```
1 import { createClient } from '@sanity/client';
2 import axios from 'axios';
3 import dotenv from 'dotenv';
4 import { fileURLToPath } from 'url';
5 import path from 'path';
6
7 // Load environment variables from .env.local
8 const __filename = fileURLToPath(import.meta.url);
9 const __dirname = path.dirname(__filename);
10 dotenv.config({ path: path.resolve(__dirname, '../.env.local') });
11
12 // Create Sanity client
13 const client = createClient({
14   projectId: "zn3e0jov",
15   dataset: "production",
16   useCdn: true,
17   token: "skn0iCKi5I38atyarznOGKN8hZ9zKWcVisAGW2DRCIN7D0cCJ1LaEjtPzPRNf6TdJdJRm3FuP13HZUZ3TymT78J1WwX51kYUL4Cg6mn",
18   apiVersion: '2021-08-31',
19 });
20
21 async function uploadImageToSanity(imageUrl) {
22   try {
23     console.log(`Uploading image: ${imageUrl}`);
24     const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
25     const buffer = Buffer.from(response.data);
26     const asset = await client.assets.upload('image', buffer, {
27       filename: imageUrl.split('/').pop(),
28     });
29     console.log(`Image uploaded successfully: ${asset._id}`);
30   } catch (error) {
31     console.error('Error uploading image:', error);
32   }
33 }
```

The status bar at the bottom indicates the file is at 'Ln 125, Col 14' with 'Spaces: 2', 'UTF-8' encoding, 'CRLF' line endings, and 'JavaScript' language. It also shows 'Go Live' and 'Prettier' icons.

```
import-data.mjs .env.local x
.env.local
1 NEXT PUBLIC SANITY PROJECT ID="zn3e0iov"
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Microsoft Windows [Version 10.0.26100.2894]
(c) Microsoft Corporation. All rights reserved.

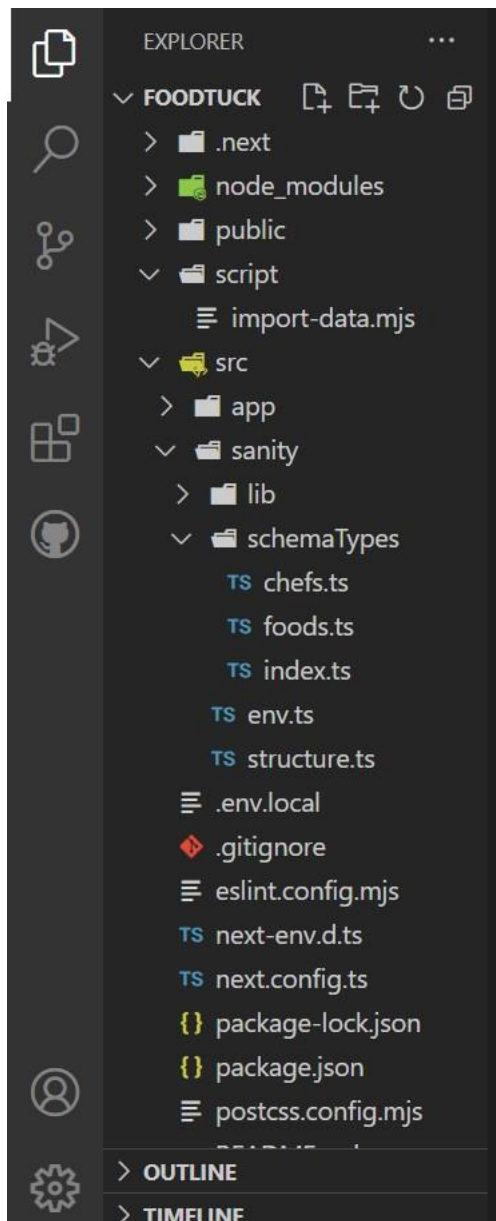
C:\Users\PMLS\Desktop\Day_3\foodtuck>npm run import-data

> foodtuck@0.1.0 import-data
> node script/import-data.mjs

Fetching food, chef data from API...
Processing food: Fresh Lime
Uploading image: https://sanity-nextjs-rouge.vercel.app/food/food-1.png
Image uploaded successfully: image-e155a50cd8bcd0e82ca81649945bd30a2fc111276-1248x1068-png
Uploading food to Sanity: Fresh Lime
Food uploaded successfully: FoP6QGkAI9c8N6wTb7jqY1
Processing food: Chocolate Muffin
Uploading image: https://sanity-nextjs-rouge.vercel.app/food/food-2.png
Image uploaded successfully: image-2392122ad0508b5c890aebd8ca36436c214fd7db-1248x1068-png
Uploading food to Sanity: Chocolate Muffin
Food uploaded successfully: FoP6QGkAI9c8N6wTb7jqTF
Processing food: Burger
Uploading image: https://sanity-nextjs-rouge.vercel.app/food/food-3.png
Image uploaded successfully: image-95a970acfaa0bc5e7df93be9527c2d8a1bc93562-1248x1068-png
Uploading food to Sanity: Burger
Food uploaded successfully: Cpl9FWqkqp36faTVGKDfoi
Processing food: Country Burger
Uploading image: https://sanity-nextjs-rouge.vercel.app/food/food-4.png
Image uploaded successfully: image-ea501a87e980fc084fb7deacca2c3538b167a477-1248x1068-png
Uploading food to Sanity: Country Burger
Food uploaded successfully: FoP6QGkAI9c8N6wTb7jr8P
```

```
import-data.mjs .env.local x
.env.local
1 NEXT PUBLIC SANITY PROJECT ID="zn3e0iov"
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Chef uploaded successfully: FoP6QGkAI9c8N6wTb7jsG9
Processing chef: Jorina Begum
Uploading image: https://sanity-nextjs-rouge.vercel.app/chef/chef-2.png
Image uploaded successfully: image-a8a4535b34a230733d2ef6eb5c0a4169f65226d5-1248x1517-png
Uploading chef to Sanity: Jorina Begum
Chef uploaded successfully: Cpl9FWqkqp36faTVGKDhQw
Processing chef: M. Mohammad
Uploading image: https://sanity-nextjs-rouge.vercel.app/chef/chef-3.png
Image uploaded successfully: image-9b9161acc32440ad4a6b853851b9c232b9c9c53e-1248x1517-png
Uploading chef to Sanity: M. Mohammad
Chef uploaded successfully: FoP6QGkAI9c8N6wTb7jsqX
Processing chef: Munna Kathy
Uploading image: https://sanity-nextjs-rouge.vercel.app/chef/chef-4.png
Image uploaded successfully: image-03eb4eacebd8ca11b707cfc569b87894b4e9bd57-1248x1517-png
Uploading chef to Sanity: Munna Kathy
Chef uploaded successfully: Cpl9FWqkqp36faTVGKDIZo
Processing chef: Bisnu Devgon
Uploading image: https://sanity-nextjs-rouge.vercel.app/chef/chef-5.png
Image uploaded successfully: image-7576fb850ddb0f7d4cefab457f848c09a816186d-1248x1517-png
Uploading chef to Sanity: Bisnu Devgon
Chef uploaded successfully: Cpl9FWqkqp36faTVGKDjJw
Processing chef: William Rumi
Uploading image: https://sanity-nextjs-rouge.vercel.app/chef/chef-6.png
Image uploaded successfully: image-ef1c3b9ecfd9bc1aad0a931c6b4c564d6939e4f8-1248x1517-png
Uploading chef to Sanity: William Rumi
Chef uploaded successfully: FoP6QGkAI9c8N6wTb7jtQv
Data import completed successfully!

C:\Users\PMLS\Desktop\Day_3\foodtuck>
```





```
FOODTUCK
src
  app
  api
  blog
  blogdetails
  checkout
  chefs
  components
  contact
  faq
  menu
  pages
  products
    page.tsx
  shop
  shoppingcart
  signin
  signup
  studio\[[...tool]]
    page.tsx
  favicon.ico
  globals.css
  layout.tsx
  name.tsx
OUTLINE
TIMELINE

src > app > products > page.tsx > ...
1  "use client";
2  import React, { useEffect, useState } from "react";
3  import sanity from "@sanity"; // Import from sanity.ts
4  import Image from "next/image";
5
6  interface Product {
7    _id: string;
8    title: string;
9    price: number;
10   category: string;
11   description: string;
12   imageUrl: string;
13   tags: string[];
14 }
15
16 const ProductCards: React.FC = () => {
17   const [products, setProducts] = useState<Product[]>([]);
18   const [cart, setCart] = useState<Product[]>([]);
19   const [error, setError] = useState<string | null>(null);
20
21   const fetchProducts = async () => {
22     try {
23       const query = `
24         *[_type == "product"] {
25           _id,
26           title,
27           price,
28           category,
29           description,
30         }
31       `;
32
33       const data = await sanity.fetch(query);
34       console.log("Fetched Products:", data); // Debugging
35       setProducts(data);
36       setError(null);
37     } catch (err: any) {
38       console.error("Error Fetching Products:", err);
39       setError("Failed to load products. Please try again later.");
40     }
41   };
42
43   useEffect(() => {
44     fetchProducts();
45   }, []);
46
47   return (
48     <div className="p-4">
49       <h2 className="text-center text-slate-800 mt-4 mb-4">
50         Products From API's Data
51       </h2>
52
53       {error && <p className="text-red-500 text-center">{error}</p>}
54
55       {products.length === 0 && !error && (
56         <p className="text-center text-gray-600">Loading products...</p>
57       )}
58
59       <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-4">
60         {products.map((product) => (
61           <div
62             key={product._id}
63             className="bg-white shadow-md rounded-lg p-4 hover:shadow-lg transition-shadow duration-300"
64           >
65             <img alt={product.imageUrl} className="w-100px h-100px object-cover mb-4" />
66             <p>{product.title}</p>
67             <p>{product.price}</p>
68             <p>{product.category}</p>
69             <p>{product.description}</p>
70             <p>{product.tags.join(", ")}</p>
71           </div>
72         ))}
73       </div>
74     </div>
75   );
76 }
```

```
FOODTUCK
src
  app
  api
  blog
  blogdetails
  checkout
  chefs
  components
  contact
  faq
  menu
  pages
  products
    page.tsx
  shop
  shoppingcart
  signin
  signup
  studio\[[...tool]]
    page.tsx
  favicon.ico
  globals.css
  layout.tsx
  name.tsx
OUTLINE
TIMELINE

src > app > products > page.tsx > ...
16 const ProductCards: React.FC = () => {
21   const fetchProducts = async () => {
22     try {
23       const query = `
24         *[_type == "product"] {
25           _id,
26           title,
27           price,
28           category,
29           description,
30           imageUrl: productImage.asset->url,
31           tags
32         }
33       `;
34       const data = await sanity.fetch(query);
35       console.log("Fetched Products:", data); // Debugging
36       setProducts(data);
37       setError(null);
38     } catch (err: any) {
39       console.error("Error Fetching Products:", err);
40       setError("Failed to load products. Please try again later.");
41     }
42   };
43 }
```

```
FOODTUCK
src
  app
  api
  blog
  blogdetails
  checkout
  chefs
  components
  contact
  faq
  menu
  pages
  products
    page.tsx
  shop
  shoppingcart
  signin
  signup
  studio\[[...tool]]
    page.tsx
  favicon.ico
  globals.css
  layout.tsx
  name.tsx
OUTLINE
TIMELINE

src > app > products > page.tsx > ...
16 const ProductCards: React.FC = () => {
43
44   useEffect(() => {
45     fetchProducts();
46   }, []);
47
48   return (
49     <div className="p-4">
50       <h2 className="text-center text-slate-800 mt-4 mb-4">
51         Products From API's Data
52       </h2>
53
54       {error && <p className="text-red-500 text-center">{error}</p>}
55
56       {products.length === 0 && !error && (
57         <p className="text-center text-gray-600">Loading products...</p>
58       )}
59
60       <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-4">
61         {products.map((product) => (
62           <div
63             key={product._id}
64             className="bg-white shadow-md rounded-lg p-4 hover:shadow-lg transition-shadow duration-300"
65           >
66             <img alt={product.imageUrl} className="w-100px h-100px object-cover mb-4" />
67             <p>{product.title}</p>
68             <p>{product.price}</p>
69             <p>{product.category}</p>
70             <p>{product.description}</p>
71             <p>{product.tags.join(", ")}</p>
72           </div>
73         ))}
74       </div>
75     </div>
76   );
77 }
```





```
107 <h2 className="text-lg font-black text-red-400">Cart Summary</h2>
108 {cart.length > 0 ? (
109   <ul className="space-y-4">
110     {cart.map((item, index) => (
111       <li key={index} className="flex justify-between items-center bg-white shadow-sm p-4 rounded-md">
112         <div>
113           <p className="font-medium text-slate-900">{item.title}</p>
114           <p className="text-sm text-blue-600">${item.price.toFixed(2)}</p>
115         </div>
116         <Image src={item.imageUrl} alt={item.title} width={50} height={50} className="rounded-md" />
117       </li>
118     )
119   )}
120 ) : (
121   <p className="text-center text-slate-600">Your Cart Is Empty.</p>
122 )}
123 </div>
124 </div>
125 );
126 };
127
128 export default ProductCards;
129
```

Products From API's Data

Loading products...

**Cart Summary**

Your Cart Is Empty.