Day 3 - API Integration Report - Furniture Store

API Integration Process

The following steps were taken to integrate the API `https://template-0-beta.vercel.app/api/product` with Sanity.io and display the data in a Next.js frontend:

1. **Set Up Sanity.io**
   - Create A new sanity project.
   - Created a dataset (`production`) and selected the E-commerce schema template.
   - Started the Sanity Studio locally..
2. **Defined Product Schema**
   - Created a schema file `product.ts` to define fields for `id`, `name`, `imagePath`, `price`, `description`, `discountPercentage`, `isFeaturedProduct`, `stockLevel`, and `category`.
   - Registered the schema in `schema.ts` and deployed it using `sanity deploy`.
3. **API Integration**
   - Created a `sanityClient.ts` file to configure the Sanity client using project ID, dataset, and API token.
   - Built an API route in `pages/api/fetch-and-insert.ts` to fetch product data from the external API and insert it into Sanity CMS.
   - Used `axios` to fetch data and `client.create` to insert the data into the `product` schema in Sanity.
4. **Display Data in Frontend**
   - Fetched data from Sanity in `home.tsx` using `client.fetch` with a GROQ query (`*[_type == 'product']`).
   - Rendered the product data dynamically in the frontend.

Adjustments Made to Schemas

The following adjustments were made to the default Sanity schema:

- Added a custom schema `product.ts` to represent API data.
- Defined the following fields:

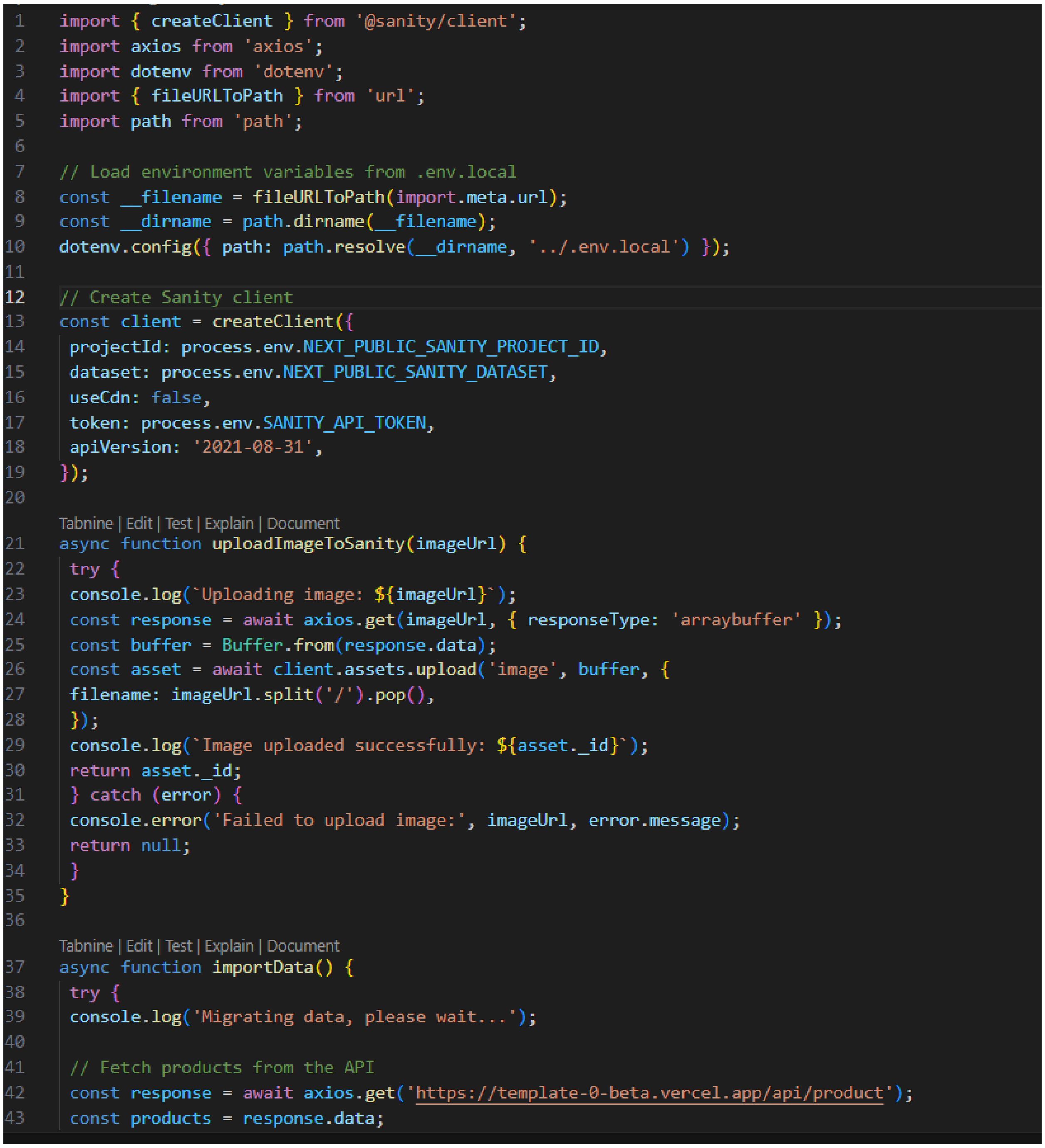
- `id` (string): Unique identifier for the product.
- `name` (string): Product name.
- `imagePath` (url): URL for the product image.
- `price` (number): Product price.
- `description` (text): Product description.
- `discountPercentage` (number): Discount percentage.
- `isFeaturedProduct` (boolean): Flag for featured products.
- `stockLevel` (number): Available stock.
- `category` (string): Product category.

Migration Steps and Tools Used

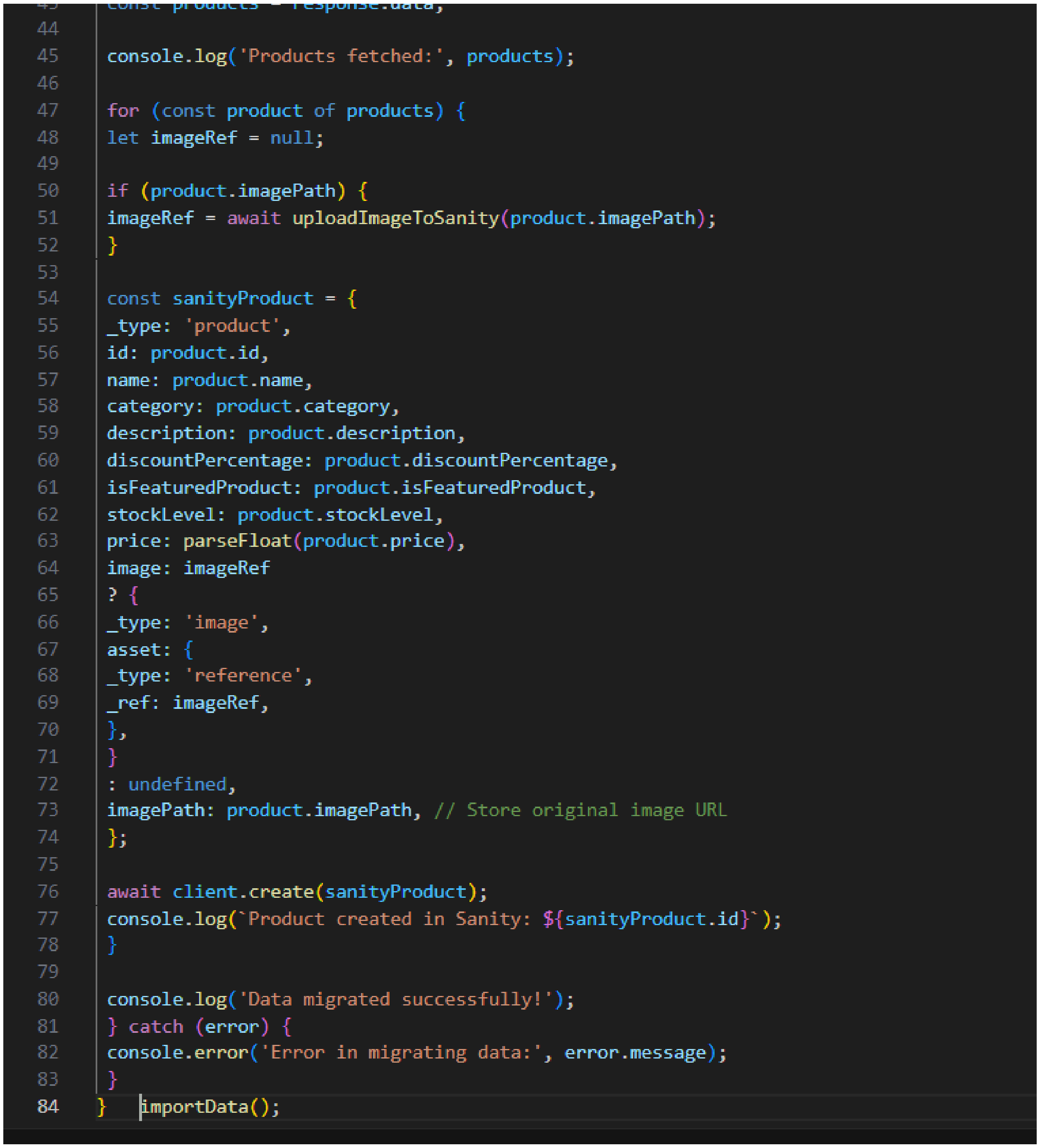
1. **Tools Used**
   - `axios`: To fetch product data from the external API.
   - `@sanity/client`: To interact with Sanity CMS.
   - `sanity deploy`: To deploy schema updates to the Sanity backend.
2. **Steps for Migration**
   - Created a migration script in `scripts/data-migration.mjs`.
   - Fetched product data from the API.
   - Transformed and validated the data to match the Sanity schema.
   - Inserted the data into Sanity using the `client.create` method.

Screenshots

1. **API Calls**

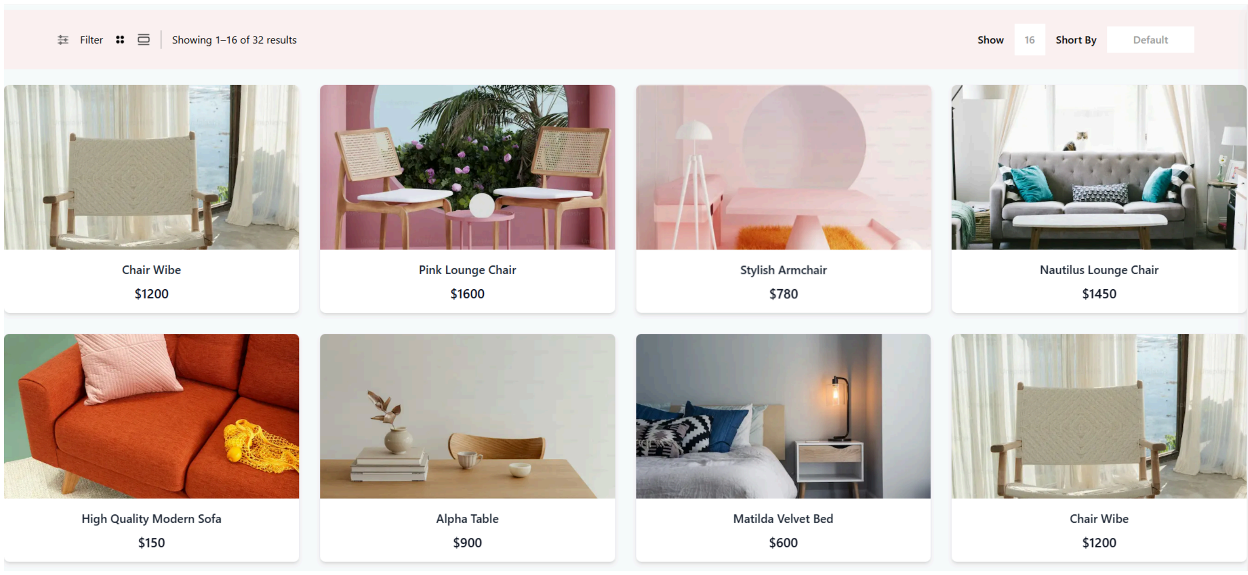

Tabnine | Edit | Test | Explain | Document
Tabnine | Edit | Test | Explain | Document

```javascript
import { createClient } from '@sanity/client';
import axios from 'axios';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';
import path from 'path';

// Load environment variables from .env.local
const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);
dotenv.config({ path: path.resolve(__dirname, '../.env.local') });

// Create Sanity client
const client = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
  useCdn: false,
  token: process.env.SANITY_API_TOKEN,
  apiVersion: '2021-08-31',
});

async function uploadImageToSanity(imageUrl) {
  try {
  console.log(`Uploading image: ${imageUrl}`);
  const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
  const buffer = Buffer.from(response.data);
  const asset = await client.assets.upload('image', buffer, {
  filename: imageUrl.split('/').pop(),
  });
  console.log(`Image uploaded successfully: ${asset._id}`);
  return asset._id;
  } catch (error) {
  console.error('Failed to upload image:', imageUrl, error.message);
  return null;
  }
}

async function importData() {
  try {
  console.log('Migrating data, please wait...');

  // Fetch products from the API
  const response = await axios.get('https://template-0-beta.vercel.app/api/product');
  const products = response.data;
```



```
const products = response.data;

console.log('Products fetched:', products);

for (const product of products) {
let imageRef = null;

if (product.imagePath) {
imageRef = await uploadImageToSanity(product.imagePath);
}

const sanityProduct = {
_type: 'product',
id: product.id,
name: product.name,
category: product.category,
description: product.description,
discountPercentage: product.discountPercentage,
isFeaturedProduct: product.isFeaturedProduct,
stockLevel: product.stockLevel,
price: parseFloat(product.price),
image: imageRef
? {
_type: 'image',
asset: {
_type: 'reference',
_ref: imageRef,
},
}
: undefined,
imagePath: product.imagePath, // Store original image URL
};

await client.create(sanityProduct);
console.log(`Product created in Sanity: ${sanityProduct.id}`);
}

console.log('Data migrated successfully!');
} catch (error) {
console.error('Error in migrating data:', error.message);
}
}   importData();
```

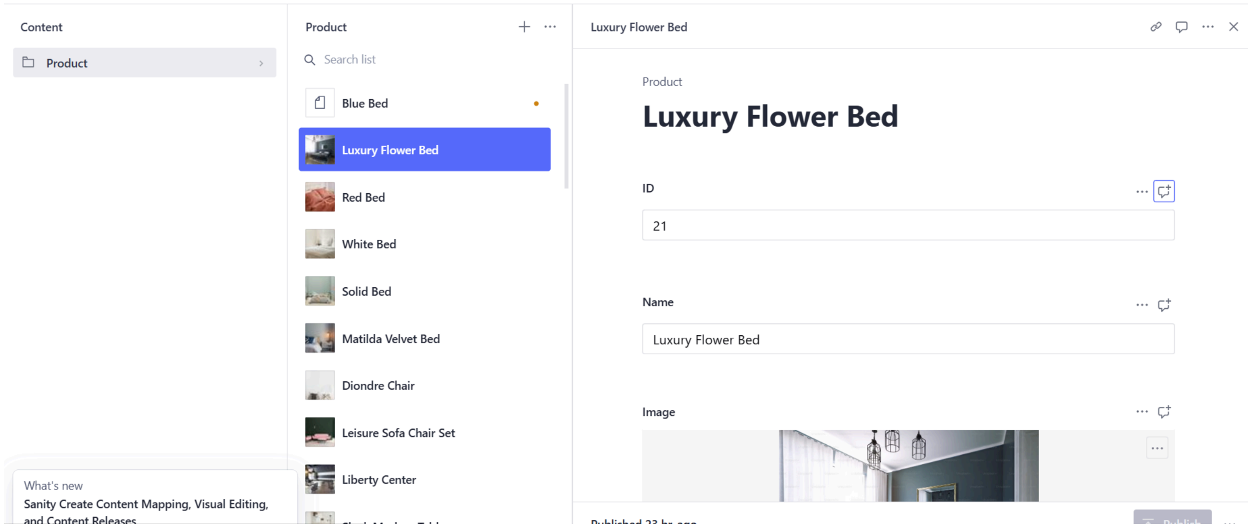
2. **Frontend Display**


Filter
Showing 1–16 of 32 results
Show 16 Short By Default

Chair Wibe	Pink Lounge Chair	Stylish Armchair	Nautilus Lounge Chair
$1200	$1600	$780	$1450
High Quality Modern Sofa	Alpha Table	Matilda Velvet Bed	Chair Wibe
$150	$900	$600	$1200

.

3. **Sanity CMS Fields**



Product
Luxury Flower Bed
ID
21
Name
Luxury Flower Bed
Image

Code Snippets

```
import { createClient } from '@sanity/client';
import axios from 'axios';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';
import path from 'path';

// Load environment variables from .env.local
const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);
dotenv.config({ path: path.resolve(__dirname, '../.env.local') });

// Create Sanity client
const client = createClient({
 projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
 dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
 useCdn: false,
 token: process.env.SANITY_API_TOKEN,
 apiVersion: '2021-08-31',
});

async function uploadImageToSanity(imageUrl) {
 try {
 console.log(`Uploading image: ${imageUrl}`);
 const response = await axios.get(imageUrl, { responseType: 'arraybuffer'
});
 const buffer = Buffer.from(response.data);
 const asset = await client.assets.upload('image', buffer, {
 filename: imageUrl.split('/').pop(),
 });
 console.log(`Image uploaded successfully: ${asset._id}`);
 return asset._id;
 } catch (error) {
 console.error('Failed to upload image:', imageUrl, error.message);
 return null;
 }
}

async function importData() {
 try {
 console.log('Migrating data, please wait...');

 // Fetch products from the API
 const response = await
axios.get('https://template-0-beta.vercel.app/api/product');
 const products = response.data;

 console.log('Products fetched:', products);
 for (const product of products) {
 let imageRef = null;

 if (product.imagePath) {
 imageRef = await uploadImageToSanity(product.imagePath);
 }

 const sanityProduct = {
 _type: 'product',
 id: product.id,
 name: product.name,
 category: product.category,
 description: product.description,
 discountPercentage: product.discountPercentage,
 isFeaturedProduct: product.isFeaturedProduct,
 stockLevel: product.stockLevel,
 price: parseFloat(product.price),
 image: imageRef
 ? {
 _type: 'image',
 asset: {
 _type: 'reference',
 _ref: imageRef,
 },
 }
 : undefined,
 imagePath: product.imagePath, // Store original image URL
 };

 await client.create(sanityProduct);
 console.log(`Product created in Sanity: ${sanityProduct.id}`);
 }

 console.log('Data migrated successfully!');
 } catch (error) {
 console.error('Error in migrating data:', error.message);
 }
}    importData();
```

Fetching Data in Frontend

```
import React from 'react';
import Image from 'next/image';
import Navbar from './navbar';

import { client } from '@/sanity/lib/client';
import { urlFor } from '@/sanity/lib/image';
import Filter from './filter';
import Delivery from './delivery';
import Hero from './hero';

interface Product {
  category: string;
  id: string;
  price: number;
  description: string;
  stockLevel: number;
  imagePath: string;
  discountPercentage: number;
  isFeaturedProduct: number;
  name: string;
  image: any;
}

// Fetch products from Sanity
async function fetchProducts(): Promise<Product[]> {
  const query = `*[_type == "product"]{
    category,
    _id,
    price,
    description,
    stockLevel,
    imagePath,
    discountPercentage,
    isFeaturedProduct,
    name,
    "image":image.asset._ref
  }`;
  const products = await client.fetch(query);
  return products;
```

```
}

const Shop = async () => {
  const products = await fetchProducts();

  return (
    <div className="bg-gray-50">
      <div className="max-w-screen-2xl container mx-auto pb-8 px-4">
        {/* Header Section */}
        <div className="bg-[#faf4f4]">
          <Navbar />
        </div>

            <Hero />

          {/* Breadcrumb Section */}
          <div className="absolute top-1/2 left-1/2 transform
-translate-x-1/2 -translate-y-1/2 mt-14">

          </div>
        </div>

        {/* Shop Line Section */}
        <div className="my-6">
          <Filter />
        </div>

        {/* Product List */}
        <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3
lg:grid-cols-4 gap-8">
          {products.map((product: Product, index) => (
            <div
              key={index}
              className="flex flex-col items-center bg-white shadow-md
rounded-lg overflow-hidden transition-transform transform hover:scale-105"
            >
              {/* Product Image */}
              <Image
                src={urlFor(product.image).url()} // Convert
ImageUrlBuilder to string URL
```

```
                alt={product.name}
                height={300}
                width={300}
                className="h-[250px] w-full object-cover"
              />
              <div className="p-4 text-center">
                {/* Product Name */}
                <p className="text-lg font-medium
text-gray-800">{product.name}</p>
                {/* Product Price */}
                <h3 className="text-xl font-semibold text-gray-900
mt-2">${product.price}</h3>
              </div>
            </div>
          ))}
        </div>

        {/* Additional Sections */}
        <div className="justify-center mx-auto">

        </div>
        <Delivery />
      </div>

  );
};

export default Shop;
```