DAY 4 HACKATHON: DYNAMIC FRONTEND COMPONENTS

Report on E-Commerce MarketPlace Website Development

Introduction:

The developed e-commerce website is a fully functional and responsive marketplace built with modern technologies, providing a seamless user experience for browsing, searching, and purchasing products. The project incorporates advanced features like dynamic product listing, category-based filtering, payment gateway integration, and an intuitive cart system, ensuring high performance and scalability.

Technologies Used

1. Next.js Framework:

Description: Next.js, a React-based framework, was chosen for its server-side rendering (SSR) capabilities, static site generation (SSG), and API route handling. These features ensure optimal performance and SEO-friendly architecture.

Implementation: Pages and components are modularized, leveraging Next.js' dynamic routing to enhance navigation between categories, products, and checkout pages.

2. Tailwind CSS:

Description: Tailwind CSS, a utility-first CSS framework, was employed for designing responsive and mobile-friendly layouts with minimal CSS code.

Usage: Custom classes provided flexibility in implementing reusable and adaptive design components, such as product cards, navigation bars, and grid layouts.

3. Sanity CMS

Description: Sanity, a headless content management system (CMS), was integrated to manage product data dynamically. It provides a robust platform for data management and real-time updates.

Updates Implemented:

Price Field Update: The product schema was updated to include a `price` field for dynamically setting and updating product prices.

- Ensures accurate and consistent product pricing across the website.

Hero Section Fields: A new schema for the Hero Section was created to include fields like `title`, `subtitle`, and `image`.

This allows easy customization of the Hero Section's content directly from Sanity CMS without requiring code updates.

Implementation:

- API queries fetched data from Sanity for products, categories, and hero sections.
- Sanity's asset pipeline efficiently handled images, ensuring fast loading.

4. API Integration:

Description: APIs were utilized to fetch external data and sync it with the CMS.

Implementation: A custom API was developed to fetch categorized product data.
- Client-side rendering with fallback mechanisms ensured seamless data fetching and error handling.

5. Stripe Payment Gateway:

Description: Stripe, a secure and reliable payment gateway, was integrated for managing transactions.

Usage: The checkout flow includes secure token generation and server-side processing for payment handling.
- Stripe's APIs facilitated real-time updates and notifications for successful transactions.

6. Shopping Cart with `use-shopping-cart`:

Description: The `use-shopping-cart` library provided a pre-built solution for managing cart functionality.

Features:
- Add, remove, and update items in the cart.
- Persistent cart state across sessions for enhanced user experience.

7. Shadcn Library:

Description: Shadcn was used to integrate pre-designed and accessible UI components, expediting development without compromising design consistency.

Implementation: Buttons, modals, and form inputs were styled and customized to align with the project theme.

Website Features:

1.Responsive Design:

Description:

- The website is fully responsive and adapts seamlessly across devices of various screen sizes, including desktops, tablets, and mobile phones.
- Tailwind CSS utilities (`grid`, `flex`, and `media queries`) were used to create a fluid layout that adjusts components dynamically based on screen width.
- Product grids, navigation bars, and hero sections are optimized for mobile-first design, providing an excellent user experience regardless of the device.

2. Dynamic Product Listings:

Description: Products are dynamically displayed using data fetched from Sanity CMS, categorized based on user preferences.

Implementation:

- The data fetching function fetches products by category and displays them using a grid layout styled with Tailwind CSS.
- Products are updated in real-time with category filtering and search functionality for enhanced navigation.

3. Dynamic Category Page:

Description:

- A dedicated dynamic category page allows users to view products based on their selected category.
- The `getProductsByCategory` function fetches data from Sanity CMS by category parameter and dynamically renders products in the selected category.
- Next.js' dynamic routing (`[category]`) enables the creation of a single reusable page component for all categories.

Features:

- Displays category-specific products in a visually appealing grid layout.
- Handles cases where no products are available in a category with user-friendly messaging.
- The page is responsive, ensuring a consistent experience on all devices.

4. Product Detail Page:

Description: Each product includes a detailed page displaying attributes such as name, price, discount percentage, description, and images.

Implementation:

- Dynamic routing in Next.js enables efficient rendering of individual product details.
- Users can easily navigate from product listings to detail pages.

5. Search and Filter Functionality:

Description: Users can search for products using a keyword-based search bar and filter them by categories for a streamlined shopping experience.

6. Hero Section:

Description:

- A dynamic Hero Section was built using data fetched from Sanity CMS.
- The section displays visually captivating content, including a title, subtitle, and banner image, which can be updated directly through the CMS.
- Automatic transitions were implemented for a slideshow effect using `useState` and `useEffect` hooks.

7. Payment and Checkout:

Description:

- A robust checkout page facilitates secure payment processing via Stripe, enhancing user trust and security.
- User cart items are synchronized during the checkout process.

Challenges and Solutions:

Challenge 1: Managing Real-Time Product Updates:

Solution: Sanity's GROQ queries and webhooks allowed real-time data synchronization.

Challenge 2: Complex Cart State Management:

Solution: The `use-shopping-cart` library simplified state management for cart items and user interactions.

Challenge 3: Secure Payment Integration:

Solution:Stripe APIs were implemented with server-side token validation and error handling to ensure transaction security.

Challenge 4: Dynamic Hero Section Integration:

Solution: A separate schema for the Hero Section in Sanity enabled easy updates and allowed flexible control over its content.

Challenge 5: Responsive Design Complexity:

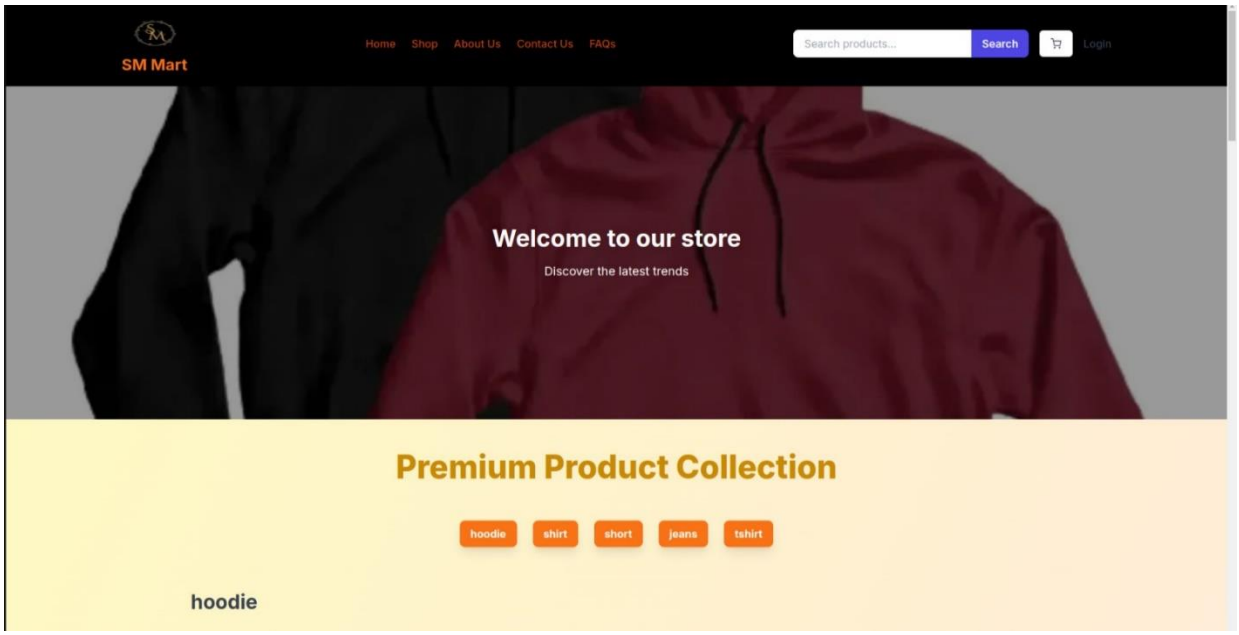
Solution: Tailwind CSS's utility classes and media queries simplified the implementation of responsive designs for grids and navigation.

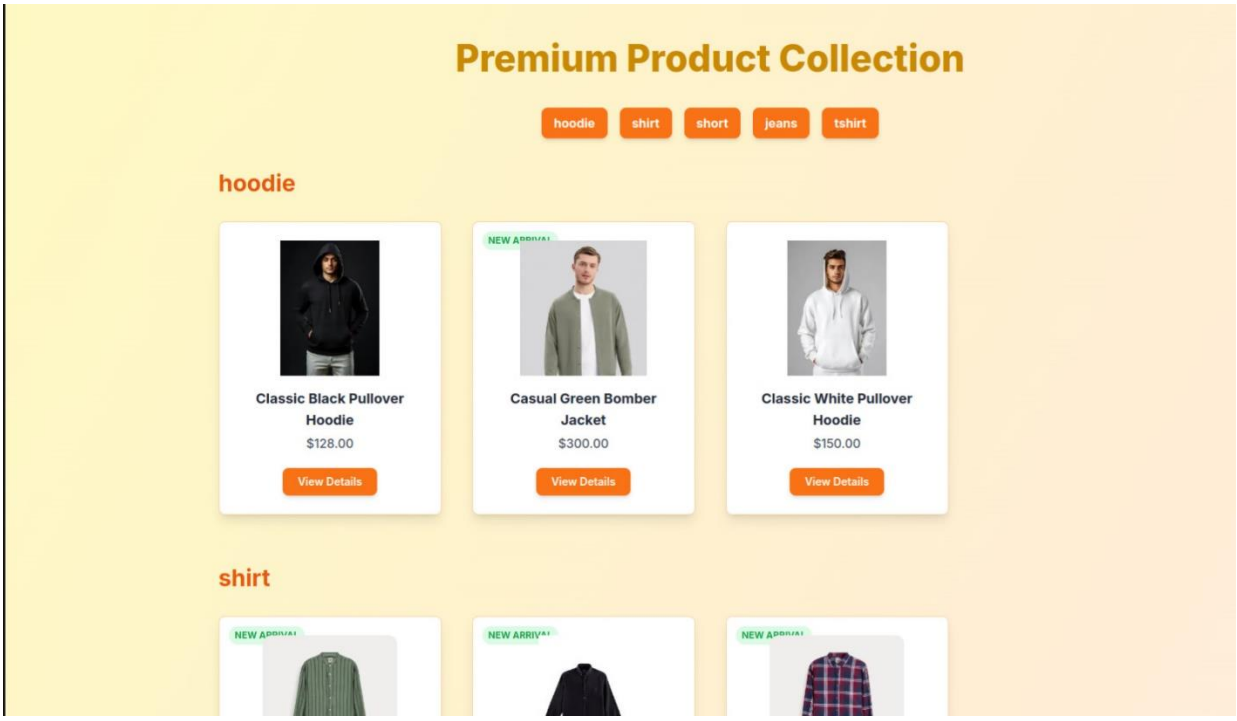
Conclusion

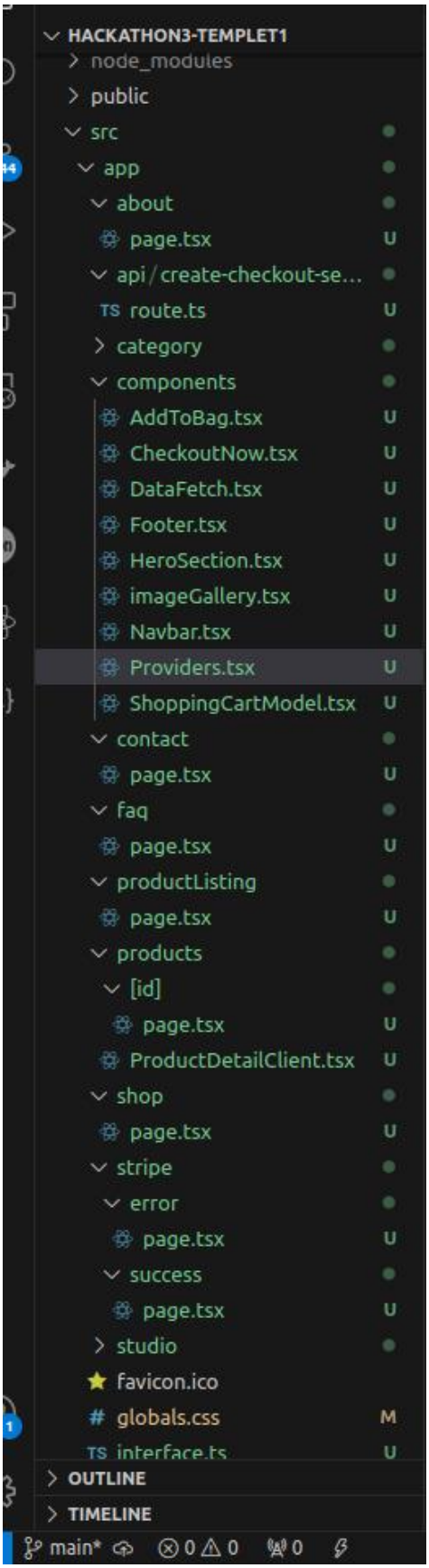
This e-commerce website leverages modern web development frameworks and tools, providing a robust, scalable, and user-friendly platform for online shopping. Its dynamic category page, responsive design, secure payment processing, and real-time data updates make it suitable for marketplace-based applications. The addition of a dynamic Hero Section and price field updates

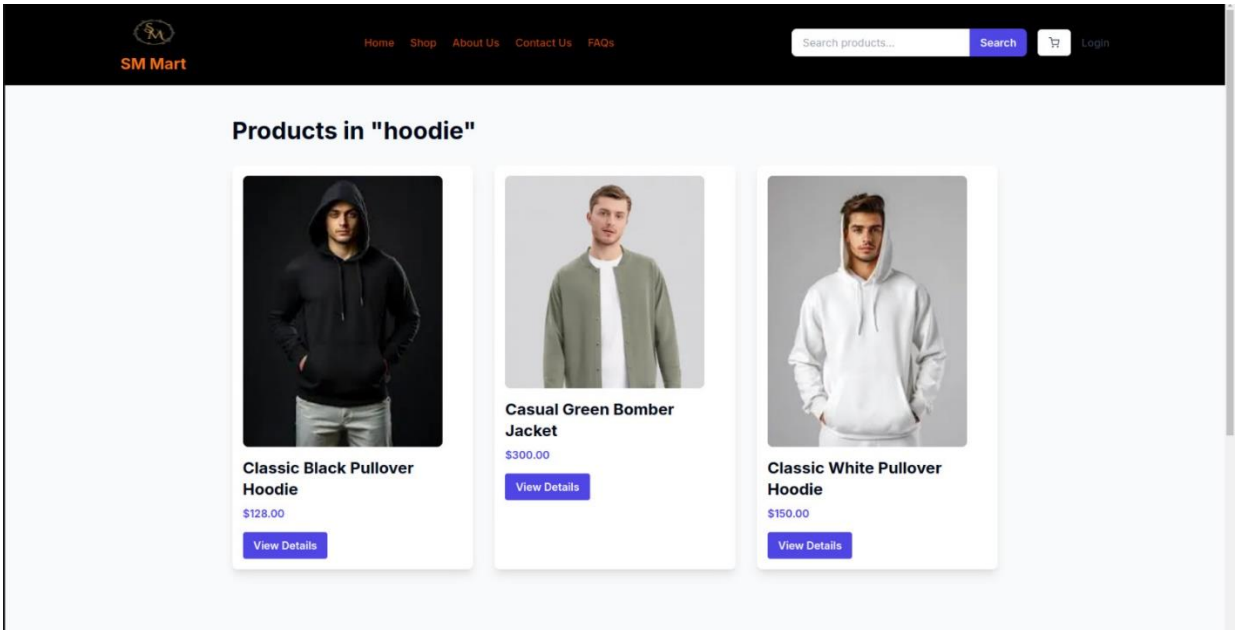
enhances the website's customization capabilities. Future improvements may include user authentication, multi-currency support, and enhanced analytics.

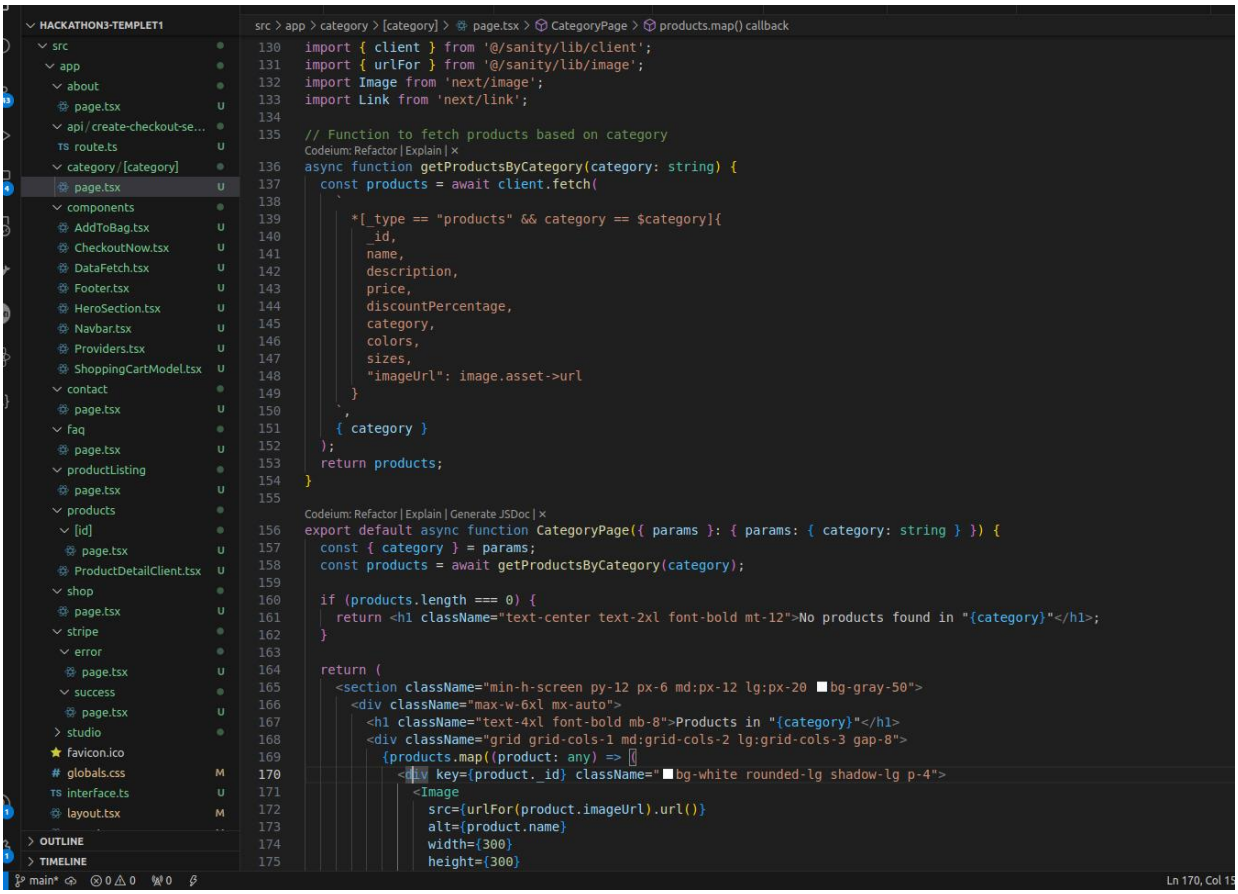
2. Functional Deliverable






HACKATHON3-TEMPLET1
node_modules
public
src
app
about
page.tsx
api / create-checkout-se...
route.ts
category
components
AddToBag.tsx
CheckoutNow.tsx
DataFetch.tsx
Footer.tsx
HeroSection.tsx
imageGallery.tsx
Navbar.tsx
Providers.tsx
ShoppingCartModel.tsx
contact
page.tsx
faq
page.tsx
productListing
page.tsx
products
[id]
page.tsx
ProductDetailClient.tsx
shop
page.tsx
stripe
error
page.tsx
success
page.tsx
studio
favicon.ico
globals.css
interface.ts
OUTLINE
TIMELINE
main*


SM Mart
Home
Shop
About Us
Contact Us
FAQs
Search products...
Search
Login
Products in "hoodie"
Classic Black Pullover Hoodie
$128.00
View Details
Casual Green Bomber Jacket
$300.00
View Details
Classic White Pullover Hoodie
$150.00
View Details


HACKATHON3-TEMPLET1
src > app > category > [category] > page.tsx > CategoryPage > products.map() callback
src
app
about
page.tsx
api/create-checkout-se...
route.ts
category/[category]
page.tsx
components
AddToBag.tsx
CheckoutNow.tsx
DataFetch.tsx
Footer.tsx
HeroSection.tsx
Navbar.tsx
Providers.tsx
ShoppingCartModel.tsx
contact
page.tsx
faq
page.tsx
productListing
page.tsx
products
[id]
page.tsx
ProductDetailClient.tsx
shop
page.tsx
stripe
error
page.tsx
success
page.tsx
studio
favicon.ico
globals.css
interface.ts
layout.tsx
OUTLINE
TIMELINE
import { client } from '@/sanity/lib/client';
import { urlFor } from '@/sanity/lib/image';
import Image from 'next/image';
import Link from 'next/link';
// Function to fetch products based on category
Codeium: Refactor | Explain | ×
async function getProductsByCategory(category: string) {
const products = await client.fetch(
`
*[_type == "products" && category == $category]{
_id,
name,
description,
price,
discountPercentage,
category,
colors,
sizes,
"imageUrl": image.asset->url
}
`,
{ category }
);
return products;
}
Codeium: Refactor | Explain | Generate JSDoc | ×
export default async function CategoryPage({ params }: { params: { category: string } }) {
const { category } = params;
const products = await getProductsByCategory(category);
if (products.length === 0) {
return <h1 className="text-center text-2xl font-bold mt-12">No products found in "{category}"</h1>;
}
return (
<section className="min-h-screen py-12 px-6 md:px-12 lg:px-20 bg-gray-50">

<h1 className="text-4xl font-bold mb-8">Products in "{category}"</h1>

{products.map((product: any) => (

<Image
src={urlFor(product.imageUrl).url()}
alt={product.name}
width={300}
height={300}
main*
Ln 170, Col 15

Casual Green Bomber Jacket

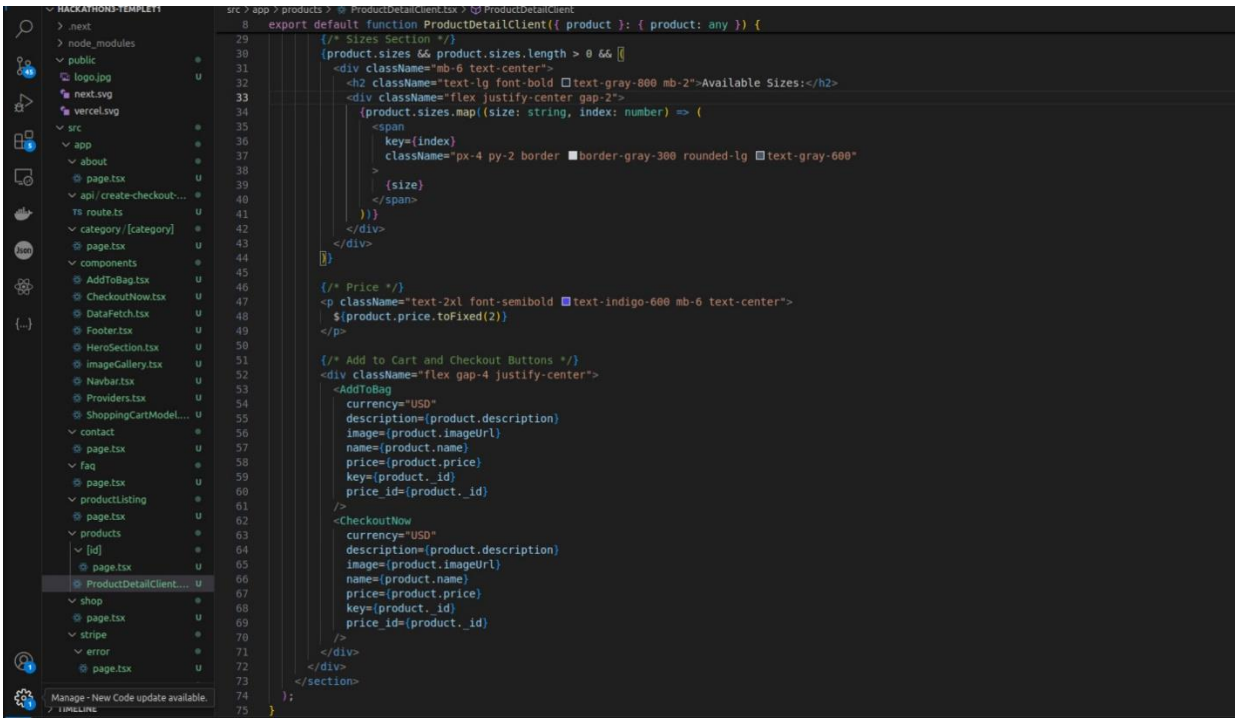
This stylish green bomber jacket offers a sleek and modern twist on a classic design. Made from soft and comfortable fabric, it features snap buttons and ribbed cuffs, giving it a sporty yet refined look. The minimalist style makes it perfect for layering over casual t-shirts or hoodies. Whether you're out with friends or just lounging, this jacket provides a laid-back yet fashionable vibe. Its muted green color adds a subtle, earthy tone that pairs well with a variety of outfits, making it a versatile addition to your casual wardrobe.

Available Sizes:

S XXL XL L

$300.00

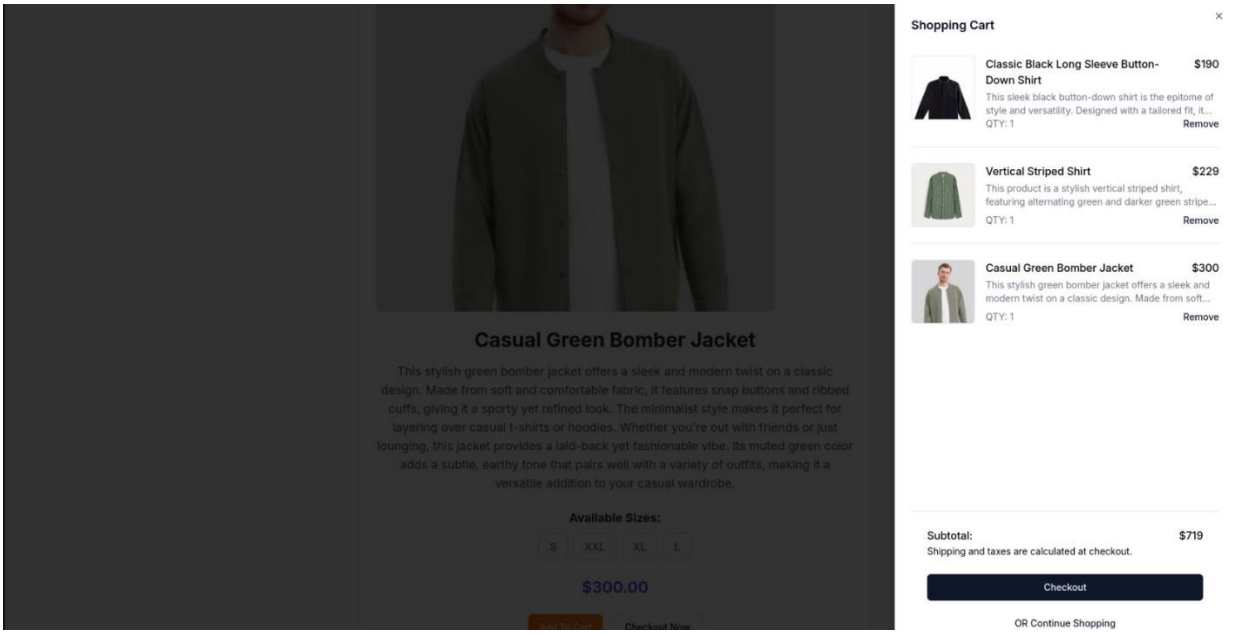
Add To Cart Checkout Now

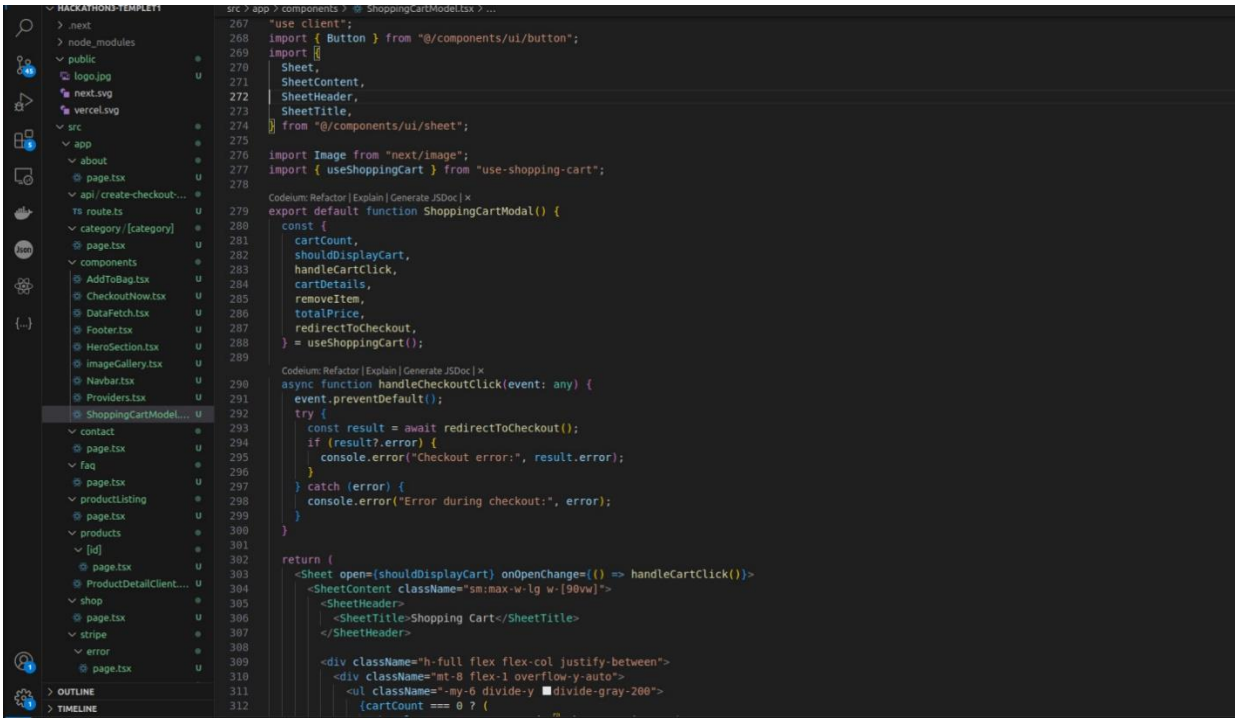


```
  price: 145,
  discountPercentage: null,
  imageUrl: 'https://cdn.sanity.io/images/nss7idml/production/18be6c1b158aacd1475323d7544d2bf5640a5b4d-295x298.png'
},
{
  _id: 'jGHNK2T7WRTGSZ6CroLBTK',
  discountPercentage: null,
  category: 'tshirt',
  isNew: true,
  colors: [ 'Red', 'Black', 'Blue', 'White' ],
  sizes: [ 'S', 'L', 'XL', 'M' ],
  imageUrl: 'https://cdn.sanity.io/images/nss7idml/production/b47f45391dc6913311e87f6afa3522e2468e65ed-295x298.png',
  name: 'Sleeve Stripe T-Shirt',
  description: 'This product is a vibrant orange and black striped t-shirt with a sporty design. The shirt features vertical pinstripes in black on an orange base, complemented by contrasting black ragl
n sleeves for a casual, athletic-inspired look. Ideal for adding a bold touch to your casual wardrobe, this tee combines comfort and style, perfect for daily wear or sporting events. The soft fabric ensu
es a comfortable fit, while the unique design makes it a standout piece for any outfit.',
  price: 130
},
{
  discountPercentage: null,
  category: 'shirt',
  isNew: true,
  colors: [ 'White', 'Black', 'Yellow', 'Blue' ],
  imageUrl: 'https://cdn.sanity.io/images/nss7idml/production/55e845afa21185a2443f7067fa2911ccd4c68e58-295x298.png',
  _id: 'jGHNK2T7WRTGSZ6CroLCBM',
  name: 'Checkered Shirt',
  sizes: [ 'XL', 'L', 'XXL', 'M' ],
  description: "This men's plaid shirt combines timeless style and comfort, featuring a bold red, navy, and white checkered pattern. With a classic button-down collar and long sleeves, it offers a relax
d fit for easy wear. Made from soft, breathable fabric, this shirt is perfect for casual outings, weekends, or layering over a t-shirt. Its versatile design ensures it pairs well with jeans or chinos, ma
ing it a wardrobe essential for any season.",
  price: 178
},
{
  description: 'Elevate your casual style with this sporty and timeless raglan t-shirt. Featuring a crisp white base adorned with vertical pinstripes, it blends a retro vibe with modern appeal. The cont
asting black raglan sleeves add a bold, athletic touch, making it a versatile piece for any wardrobe.\n' +
    '\n' +
    'Key Features:\n' +
    '\n' +
    'Soft and breathable fabric for all-day comfort\n' +
    'Classic pinstripe design for a clean, stylish look\n' +
    'Contrasting raglan sleeves for a sporty edge\n' +
    'Regular fit with a crew neckline\n' +
    'Perfect for casual outings or active days, pair this tee with your favorite jeans or joggers for effortless',
  category: 'tshirt',
  isNew: false,
  sizes: [ 'M', 'L', 'S', 'XXL' ],
  _id: 'jGHNK2T7WRTGSZ6CroLCQb',
  name: 'Black Striped T-Shirt',
  price: 120,
  discountPercentage: null,
  colors: [ 'Blue', 'White', 'Black', 'Red' ],
  imageUrl: 'https://cdn.sanity.io/images/nss7idml/production/07dc8647a64f4a93a9d1fbb41e0a81fc940b903d-295x298.png'
}
```

src > app > components > DataFetch.tsx > DataFetch > uniqueCategories.map() callback > map() callback

```
165 import React from 'react';
166 import Link from 'next/link'; // Import Link for navigation
167 import { client } from '@/sanity/lib/client';
168 import { urlFor } from '@/sanity/lib/image';
169 import Image from 'next/image';
170
    Codeium: Refactor | Explain | Generate JSDoc | ×
171 export default async function Datafetch() {
172   const query = await client.fetch(`
173     *[_type == "products"]{
174       _id, // Include the product ID for dynamic routing
175       name,
176       description,
177       price,
178       discountPercentage,
179       category,
180       isNew,
181       colors,
182       sizes,
183       "imageUrl": image.asset->url
184     }
185   `);
186
187   const groupedData = query.reduce((acc: any, product: any) => {
188     const category = product.category || 'Uncategorized';
189     if (!acc[category]) {
190       acc[category] = [];
191     }
192     acc[category].push(product);
193     return acc;
194   }, {});
195
196   const uniqueCategories = Object.keys(groupedData);
197
198   return (
199     <section className="min-h-screen bg-gradient-to-br from-yellow-100 via-orange-100 to-red-50 py-12 px-6 md:px-12 lg:px-20">
200       <div className="mx-auto max-w-7xl">
201         <h1 className="text-4xl md:text-5xl font-extrabold text-center mb-14 text-yellow-600">
202           Premium Product Collection
203         </h1>
204
205         {/* Category Links */}
206         <div className="flex flex-wrap justify-center gap-6 mb-16">
207           {uniqueCategories.map((category) => (
208             <Link
209               key={category}
210               href={`/category/${category}`}
211               className="py-2 px-4 bg-orange-500 text-white font-bold rounded-lg shadow-lg hover:bg-orange-600 transition-all"
```


Casual Green Bomber Jacket
This stylish green bomber jacket offers a sleek and modern twist on a classic design. Made from soft and comfortable fabric, it features snap buttons and ribbed cuffs, giving it a sporty yet refined look. The minimalist style makes it perfect for layering over casual t-shirts or hoodies. Whether you're out with friends or just lounging, this jacket provides a laid-back yet fashionable vibe. Its muted green color adds a subtle, earthy tone that pairs well with a variety of outfits, making it a versatile addition to your casual wardrobe.
Available Sizes:
S
XXL
XL
L
$300.00
Shopping Cart
Classic Black Long Sleeve Button-Down Shirt
$190
This sleek black button-down shirt is the epitome of style and versatility. Designed with a tailored fit, it...
QTY: 1
Remove
Vertical Striped Shirt
$229
This product is a stylish vertical striped shirt, featuring alternating green and darker green stripe...
QTY: 1
Remove
Casual Green Bomber Jacket
$300
This stylish green bomber jacket offers a sleek and modern twist on a classic design. Made from soft...
QTY: 1
Remove
Subtotal:
$719
Shipping and taxes are calculated at checkout.
Checkout
OR Continue Shopping


HACKATHON3-TEMPLETI
.next
node_modules
public
logo.jpg
next.svg
vercel.svg
src
app
about
page.tsx
api / create-checkout-...
route.ts
category / [category]
page.tsx
components
AddToBag.tsx
CheckoutNow.tsx
DataFetch.tsx
Footer.tsx
HeroSection.tsx
imageGallery.tsx
Navbar.tsx
Providers.tsx
ShoppingCartModel....
contact
page.tsx
faq
page.tsx
productListing
page.tsx
products
[id]
page.tsx
ProductDetailClient....
shop
page.tsx
stripe
error
page.tsx
OUTLINE
TIMELINE
src > app > components > ShoppingCartModel.tsx > ...
"use client";
import { Button } from "@/components/ui/button";
import {
Sheet,
SheetContent,
SheetHeader,
SheetTitle,
} from "@/components/ui/sheet";
import Image from "next/image";
import { useShoppingCart } from "use-shopping-cart";
Codeium: Refactor | Explain | Generate JSDoc | ×
export default function ShoppingCartModal() {
const {
cartCount,
shouldDisplayCart,
handleCartClick,
cartDetails,
removeItem,
totalPrice,
redirectToCheckout,
} = useShoppingCart();
Codeium: Refactor | Explain | Generate JSDoc | ×
async function handleCheckoutClick(event: any) {
event.preventDefault();
try {
const result = await redirectToCheckout();
if (result?.error) {
console.error("Checkout error:", result.error);
}
} catch (error) {
console.error("Error during checkout:", error);
}
}
return (
<Sheet open={shouldDisplayCart} onOpenChange={() => handleCartClick()}>
<SheetContent className="sm:max-w-lg w-[90vw]">
<SheetHeader>
<SheetTitle>Shopping Cart</SheetTitle>
</SheetHeader>

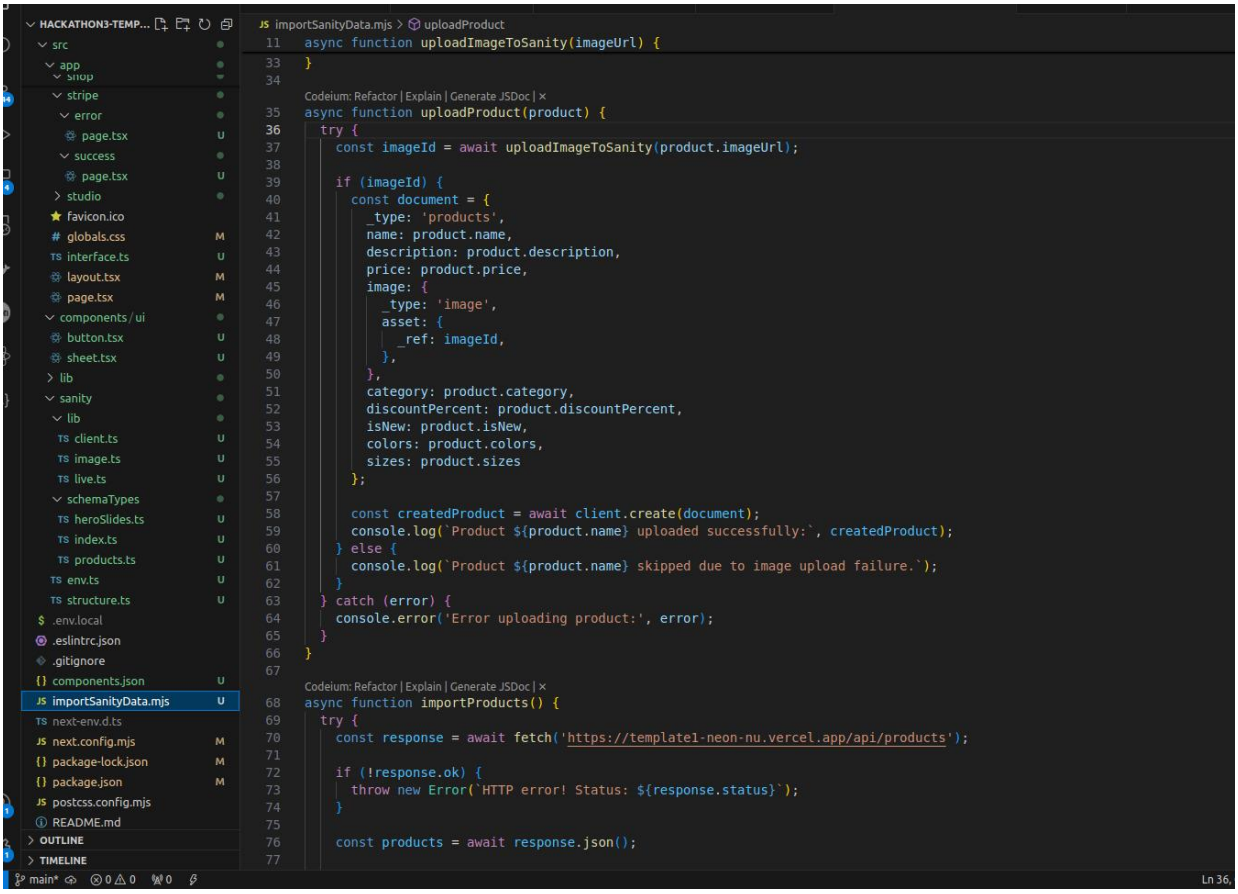
<ul className="-my-6 divide-y divide-gray-200">
{cartCount === 0 ? (

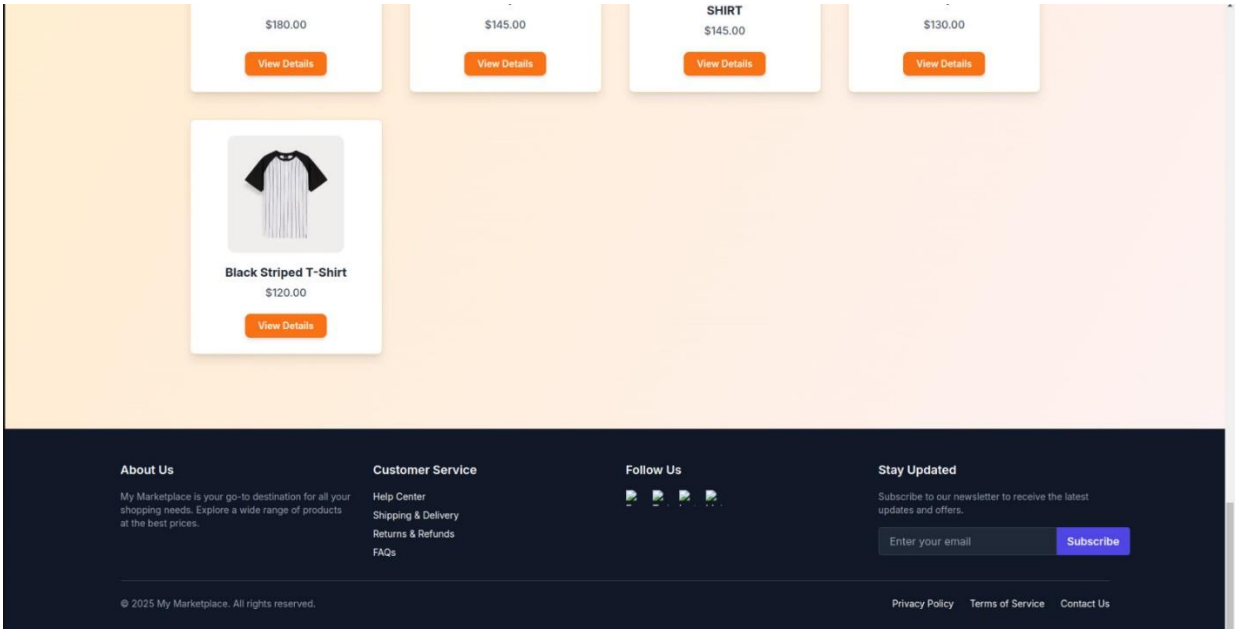
src > sanity > schemaTypes > products.ts > default > fields

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

export default defineType({
    name: 'products',
    title: 'Products',
    type: 'document',
    fields: [
        {
        name: 'name',
        title: 'Name',
        type: 'string',
        },
        {
        name: 'price',
        title: 'Price',
        type: 'number',
        },
        {
            name: 'price_id',
            title: 'Stripe Price ID',
            type: 'string',
          },
        {
        name: 'description',
        title: 'Description',
        type: 'text',
        },
        {
        name: 'image',
        title: 'Image',
        type: 'image',
        },
        {
            name:"category",
            title:"Category",
            type: 'string',
            options:{
                list:[
                    {title: 'T-Shirt', value: 'tshirt'},
                    {title: 'Short', value: 'short'},
                    {title: 'Jeans', value: 'jeans'} ,
                    {title: 'Hoddie', value: 'hoodie'} ,
                    {title: 'Shirt', value: 'shirt'} ,
                ]
            }
        },
        {
            name:"discountPercent",
```

src > sanity > schemaTypes > index.ts > ...

```python
import { type SchemaTypeDefinition } from 'sanity'
import products from './products'
import heroslides from './heroSlides'

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


HACKATHON3-TEMP...
src
app
shop
stripe
error
page.tsx
success
page.tsx
studio
favicon.ico
globals.css
interface.ts
layout.tsx
page.tsx
components/ui
button.tsx
sheet.tsx
lib
sanity
lib
client.ts
image.ts
live.ts
schemaTypes
heroSlides.ts
index.ts
products.ts
env.ts
structure.ts
.env.local
.eslintrc.json
.gitignore
components.json
importSanityData.mjs
next-env.d.ts
next.config.mjs
package-lock.json
package.json
postcss.config.mjs
README.md
OUTLINE
TIMELINE
main*
Ln 36,
importSanityData.mjs > uploadProduct
11 async function uploadImageToSanity(imageUrl) {
33 }
34
Codeium: Refactor | Explain | Generate JSDoc | ×
35 async function uploadProduct(product) {
36 try {
37 const imageId = await uploadImageToSanity(product.imageUrl);
38
39 if (imageId) {
40 const document = {
41 _type: 'products',
42 name: product.name,
43 description: product.description,
44 price: product.price,
45 image: {
46 _type: 'image',
47 asset: {
48 _ref: imageId,
49 },
50 },
51 category: product.category,
52 discountPercent: product.discountPercent,
53 isNew: product.isNew,
54 colors: product.colors,
55 sizes: product.sizes
56 };
57
58 const createdProduct = await client.create(document);
59 console.log(`Product ${product.name} uploaded successfully:`, createdProduct);
60 } else {
61 console.log(`Product ${product.name} skipped due to image upload failure.`);
62 }
63 } catch (error) {
64 console.error('Error uploading product:', error);
65 }
66 }
67
Codeium: Refactor | Explain | Generate JSDoc | ×
68 async function importProducts() {
69 try {
70 const response = await fetch('https://template1-neon-nu.vercel.app/api/products');
71
72 if (!response.ok) {
73 throw new Error(`HTTP error! Status: ${response.status}`);
74 }
75
76 const products = await response.json();
77


$180.00
View Details
$145.00
View Details
SHIRT
$145.00
View Details
$130.00
View Details
Black Striped T-Shirt
$120.00
View Details
About Us
My Marketplace is your go-to destination for all your shopping needs. Explore a wide range of products at the best prices.
Customer Service
Help Center
Shipping & Delivery
Returns & Refunds
FAQs
Follow Us
Stay Updated
Subscribe to our newsletter to receive the latest updates and offers.
Enter your email
Subscribe
© 2025 My Marketplace. All rights reserved.
Privacy Policy
Terms of Service
Contact Us