



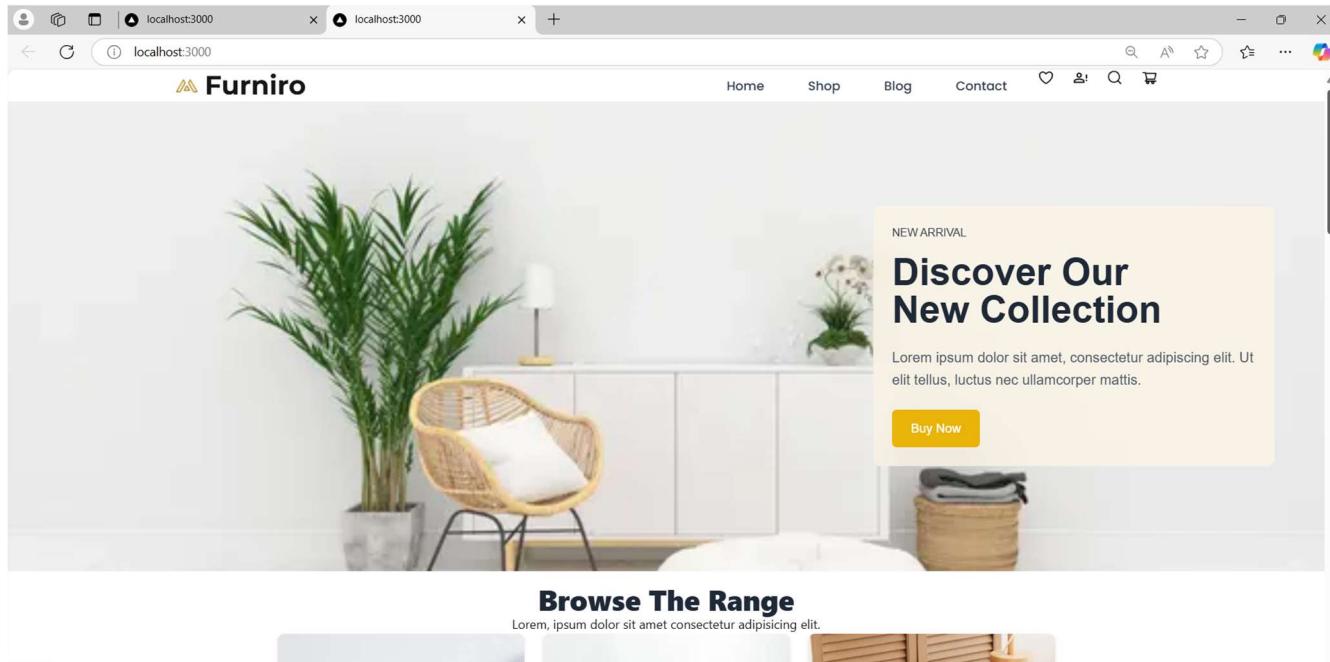
DAY 4 - BUILDING DYNAMIC FRONTEND COMPONENTS FOR MARKETPLACE

Zahida Raees

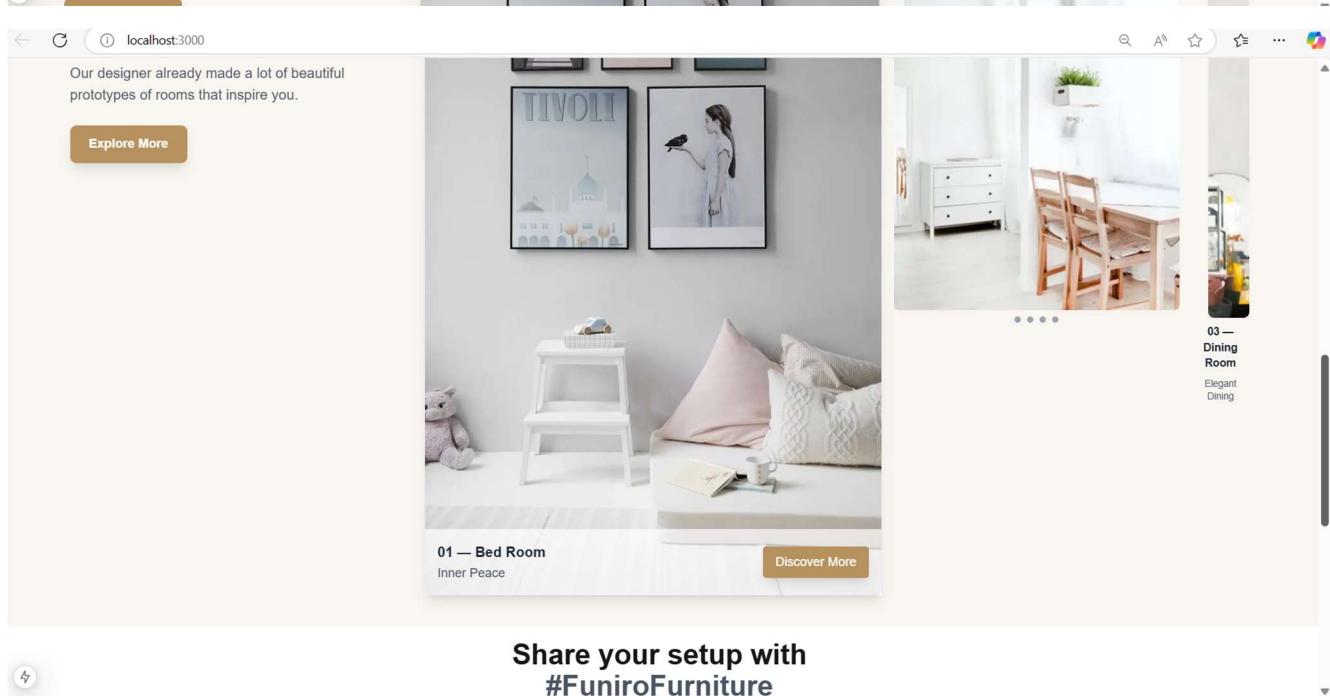
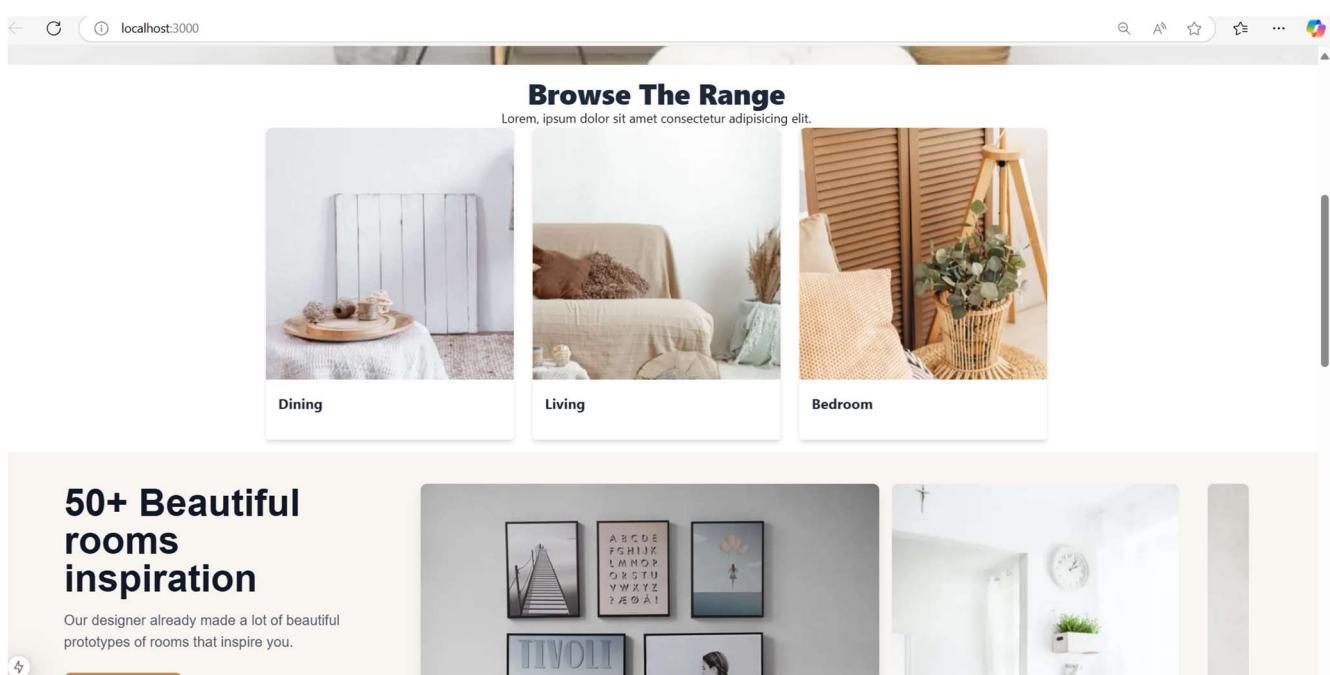


I have made copy version of template 6 and started working on it because my own marketplace require more time to make its UI and collection of product images and other details.

Here is my website's landing page which is modular and responsive:



Here is landing page's second section but its data is coming from local drive



Share your setup with
#FuniroFurniture

I have removed its 3rd section because I have to

**Share your setup with
#FuniroFurniture**



Funiro

400 University Drive Suite 200 Coral
Gables
FL 33134 USA

LINKS

Home
Shop
Blog
Contact

Help

Payment Options
Returns
Privacy Policy

[Enter Your Email Address](#)

[SUBSCRIBE](#)

© 2023 Funiro. All rights reserved.

Here is its responsiveness:

A screenshot of a web browser showing the Funiro website on a mobile device. The page displays a 'NEW ARRIVAL' section with a large image of a chair and a 'Buy Now' button. Below it is a 'Browse The Range' section with a smaller image and descriptive text.

localhost:3000

Share your setup with #FuniroFurniture

The page displays a collection of 10 small, square images arranged in two rows of five. The top row includes: a tall black shelving unit with plants; a white desk with a laptop; a dining room with a round table and chairs; a bed with a white bench at the foot; and a wooden table with chairs near a window. The bottom row includes: a gold-colored armchair; two small wooden stools; a framed picture on a wall; and a close-up of a patterned floor tile.

Funiro

Cart Page data from public folder

Cart

Product	Price	Quantity	Subtotal
	Rs. 250,000	1	Rs. 250,000

Cart Totals

Subtotal	Rs. 250,000
Total	Rs. 250,000

Check Out

High Quality
Crafted from top materials

Warranty Protection
Over 2 years

Free Shipping
Order over \$150

24/7 Support
Dedicated support

Although this product is from local folder but it's clickable and button of check out is also working.

Checkout

Home > Check Out

Billing Details

Your Order

Asgaard Sofa x 1	Rs. 250,000.00
Total	Rs. 250,000.00

Payment Method

Direct Bank Transfer

Cash on Delivery

Place Order

Day 4 ASSIGNMENT

For day 4 I have utilized sanity data which I imported from your mentioned document on day 3

First I have updated **.env.local** file with variables of product id, dataset and then token API from sanity project page. Here is the screen short of Sanity Project:

Sanity Project

The screenshot shows the Sanity Project management interface. At the top, there's a navigation bar with links like 'Getting started', 'Overview', 'Members', 'Studios', 'Datasets', 'Access', 'Activity', 'Usage', 'Plan', 'API', and 'Settings'. The 'Webhooks' tab is currently selected. On the left, there are sections for 'CORS origins' and 'Tokens'. The main content area is titled 'GROQ-powered webhooks' and contains a sub-section for 'HTTP callbacks to a given URL triggered by changes in your content lake'. It shows '0 of 2 webhooks' (2 included in plan) and a button to '+ Create webhook'. Below this, there's a placeholder icon and a message: 'There are no GROQ-powered webhooks in this project. Maybe try creating a new webhook?'. A sidebar on the left has buttons for 'Create Content Mapping', 'Visual Editing', and 'Content Releases'.

S Zahida Raees dayapi 30 days left in trial

Getting started Overview Members Studios Datasets Access Activity Usage Plan API Settings

Webhooks

CORS origins

Hosts that can connect to the project API.

ORIGIN	CREDENTIALS	CREATED
http://localhost:3000	Allowed	25 minutes
http://localhost:3333	Allowed	28 minutes

+ Add CORS origin

Tokens

Tokens are used to authenticate apps and scripts to access project data.

Name: ecommerce hackathon 3 day 3 import api

Permissions: Choose the access privileges for the token.

Contributor: Read and write access to draft content within all datasets, with no access to project settings. (Tokens: read+write drafts)

Deploy Studio (Token only): Access to deploy Sanity Studio and GraphQL APIs to our hosted service.

Developer: Read and write access to all datasets, with access to project settings for developers. (Tokens: read+write)

Editor: Read and write access to all datasets, with limited access to project settings. (Tokens: read+write)

Viewer: Read access to all datasets, with limited access to project settings. (Tokens: read-only)

Save **Cancel**

+ Add API token

Tokens

Tokens are used to authenticate apps and scripts to access project data.

Name: ecommerce hackathon 3 day 3 import api

Permissions: Choose the access privileges for the token.

Contributor: Read and write access to draft content within all datasets, with no access to project settings. (Tokens: read+write drafts)

Deploy Studio (Token only): Access to deploy Sanity Studio and GraphQL APIs to our hosted service.

Developer: Read and write access to all datasets, with access to project settings for developers. (Tokens: read+write)

Editor: Read and write access to all datasets, with limited access to project settings. (Tokens: read+write)

Viewer: Read access to all datasets, with limited access to project settings. (Tokens: read-only)

Save **Cancel**

+ Add API token

Tokens

Tokens are used to authenticate apps and scripts to access project data.

NAME	PERMISSIONS	CREATED
ecommerce hackathon 3 day 3 import api	Developer	0 seconds

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

```
skZnVAb5kyM9tKG6eFx3z7nkT35qnkwP07pkwQIZQFmbbNrUqNR4Mqb60zDTep1R3cH905gginvNgD7Mbgi12e
CcIwVJMVBlhxRRer8nUD50KwHxtTgSJk5Eqkhb12CANjNde1fT5m3LyIHae6Ei5F1LRqe6nL91hdet42eYYjkh
k56b1837
```

.env.local

File Edit Selection View Go Run Terminal Help ← → backup-template6-perfect

EXPLORER

BACKUP-TEMPLATES-PERFECT

- src
- app
 - studio
 - favicon.ico
 - # globals.css
 - index.tsx
 - layout.tsx
 - page.tsx
- sanity
 - lib
 - schemaTypes
 - index.ts
 - product.ts
 - envs.ts
 - structure.ts
 - types
- .env.local
- .eslintrc.json
- .gitignore
- next-env.d.ts
- next.config.ts
- package-lock.json
- package.json
- postcss.config.mjs
- README.md
- sanity.dcts
- sanity.config.ts
- tailwind.config.ts
- tsconfig.json

\$.env.local

```
1 NEXT_PUBLIC_SANITY_PROJECT_ID="godhjrui"
2 NEXT_PUBLIC_SANITY_DATASET="production"
3 NEXT_PUBLIC_SANITY_API_TOKEN="sk2nVab5kyM9tKG6eFx3z7nkT3qrkwP07pKwQIZQfmbbNrUqNR4Mqb6zDTep1R3cH905ggjnvNgD7MbgI12eCcivVJMVBHhxRRe8nUD5OKWhb
4
```

Product Schema

Here is the screen product schema which I got from the pdf about template 6.

```
File Edit Selection View Go Run Terminal Help ← → ⚡ backup-template6-perfect
```

EXPLORER

BACKUP-TEMPLATES

src

app

studio

favicon.ico

globals.css

index.tsx

layout.tsx

page.tsx

sanity

lib

schematypes

TS index.ts

TS products.ts

TS env.ts

TS structure.ts

types

.env.local

.eslintrc.json

.gitignore

TS next-env.d.ts

TS next.config.ts

package-lock.json

package.json

postcss.config.mjs

README.md

TS sanity.client

TS sanity.config.ts

Tailwind.config.ts

tsconfig.json

OUTLINE

TIMELINE

main

products.ts

env.ts

structure.ts

types

.env.local

.eslintrc.json

.gitignore

next-env.d.ts

next.config.ts

package-lock.json

package.json

postcss.config.mjs

README.md

sanity.client

sanity.config.ts

Tailwind.config.ts

TS products.ts

src > sanity > schemaTypes > TS products.ts > [●] product > fields

```
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      validation: (rule) => rule.required(),
17      title: "Description",
18      type: "text",
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      name: "tags",
34      type: "array",
35      title: "Tags",
36      of: [{ type: "string" }]
37    },
38  },
39  {
40    name: "discountPercentage",
41    type: "number",
42    title: "Discount Percentage",
43  },
44  {
45    name: "isNew",
46    type: "boolean",
47    title: "New Badge",
48  }
49 })
```

Ln 18, Col 25 Spaces: 4 UTF-8 CRLF ⓘ TypeScript ⓘ Go Live ⓘ Prettier

Product listing

Product listing on shop page with Product title, image, price, discount. Every image is clickable. Here header title is also dynamic means I am using single image for all pages excluding landing page. Using parameters of title and breadcrumb to update titles.

Shop
Home > Shop

Latest Products

Title	Image	Price	Discount
Rustic Vase Set		\$210	10
Cloud Haven Chair		\$230	N/A
Tropical Vibe		\$550	50
Nordic Elegance		\$280	20
Timeless Elegance		\$320	N/A
The Lucky Lamp		\$200	N/A
Pure Aura		\$280	50
Zen Table		\$250	N/A
Serene Seat		\$350	60
Sunny Chic		\$400	50
Timber Craft		\$150	N/A
Amber Haven		\$260	30
Bold Nest		\$150	60
Bright Space		\$150	60
Vase Set		\$150	60

localhost:3000/shop

<p>Title: Timber Craft</p>  <p>Price: \$320 Discount: 30</p>	<p>Title: Amber Haven</p>  <p>Price: \$150 Discount: N/A</p>	<p>Title: Bold Nest</p>  <p>Price: \$260 Discount: 30</p>	<p>Title: Bright Space</p>  <p>Price: \$180 Discount: 10</p>	<p>Title: Vase Set</p>  <p>Price: \$150 Discount: 60</p>
<p>Title: Wood Chair</p>  <p>Price: \$100 Discount: 10</p>	<p>Title: Retro Vibe</p>  <p>Price: \$340 Discount: N/A</p>	<p>Title: Sleek Living</p>  <p>Price: \$300 Discount: N/A</p>	<p>Title: Modern Serenity</p>  <p>Price: \$480 Discount: N/A</p>	

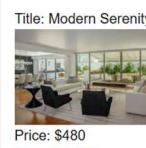
localhost:3000/shop

<p>Price: \$320 Discount: 30</p>	<p>Price: \$100 Discount: 10</p>		
<p>Title: Wood Chair</p>  <p>Price: \$100 Discount: 10</p>	<p>Title: Retro Vibe</p>  <p>Price: \$340 Discount: N/A</p>	<p>Title: Sleek Living</p>  <p>Price: \$300 Discount: N/A</p>	<p>Title: Modern Serenity</p>  <p>Price: \$480 Discount: N/A</p>

<p>Funiro 400 University Drive Suite 200 Coral Gables FL 33134 USA</p>	<p>LINKS Home Shop Blog Contact</p>	<p>Help Payment Options Returns Privacy Policy</p>	<p>Enter Your Email Address</p>	<p>SUBSCRIBE</p>
---	--	---	---	-------------------------

© 2023 Funiro. All rights reserved.

localhost:3000/shop

<p>Price: \$320 Discount: 30</p>	<p>Price: \$100 Discount: 10</p>		
<p>Title: Wood Chair</p>  <p>Price: \$100 Discount: 10</p>	<p>Title: Retro Vibe</p>  <p>Price: \$340 Discount: N/A</p>	<p>Title: Sleek Living</p>  <p>Price: \$300 Discount: N/A</p>	<p>Title: Modern Serenity</p>  <p>Price: \$480 Discount: N/A</p>

<p>Funiro 400 University Drive Suite 200 Coral Gables FL 33134 USA</p>	<p>LINKS Home Shop Blog Contact</p>	<p>Help Payment Options Returns Privacy Policy</p>	<p>Enter Your Email Address</p>	<p>SUBSCRIBE</p>
---	--	---	---	-------------------------

© 2023 Funiro. All rights reserved.

Product listing code

The screenshot shows a code editor interface with the following details:

- File Path:** src > app > shop > ProductList.tsx
- Code Editor:** Visual Studio Code (VSC) interface.
- Explorer View:** Shows the project structure under "BACKUP-TEMPLATES6-PERFECT".
- ProductList.tsx Content:** The code defines a functional component "ProductList" that takes an array of products. It uses the "useRouter" hook from Next.js to handle product clicks. The component maps over the products to create a grid of cards. Each card displays the product title and image (if available). If no image is provided, it shows a placeholder message.

```
src > app > shop > ProductList.tsx ...  
1 // List of Product from the shop page.  
2 "use client";  
3 import Image from "next/image";  
4 import { useRouter } from "next/navigation";  
5 import { Product } from "@types/products";  
6 import { urlFor } from "@sanity/lib/image";  
7  
8 const ProductList = ({ products }: { products: Product[] }) => {  
9   const router = useRouter();  
10  
11   const handleProductClick = (id: string) => {  
12     if (id) {  
13       router.push(`/shop/product/${id}`);  
14     } else {  
15       console.error("Product ID is missing!");  
16     }  
17   };  
18  
19   return (  
20     <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-5 gap-6">  
21       {products.map((product) => {  
22         const imageUrl = product.productImage  
23           ? urlFor(product.productImage).url()  
24           : null;  
25  
26         return (  
27           <div  
28             key={product._id}  
29               className="product-card cursor-pointer border p-4 rounded-lg hover:shadow-lg"  
30               onClick={() => handleProductClick(product._id)}  
31             >  
32               <p>Title: {product.title}</p>  
33  
34               {imageUrl ? (  
35                 <Image  
36                   src={imageUrl}  
37                   alt={`Image of ${product.title}`}  
38                 />  
39               ) : (  
40                 <p>No image available</p>  
41               )}  
42               <p>Price: ${product.price}</p>  
43               <p>Discount: ${product.discountPercentage || "N/A"}</p>  
44             </div>  
45         );  
46       )});  
47     </div>  
48   );  
49 }  
50  
51 };  
52  
53 export default ProductList;
```

Single Product Code:

Here I am getting error in rich text field. On display there is no error but in coding I am getting it and still in search to know its reason.

The screenshot shows a code editor interface with the following details:

- File Path:** page.tsx ...app U
- Code Editor Content:**

```
src > app > shop > product > [id] > page.tsx > ...
1 import { client } from "@sanity/lib/client";
2 import { urlFor } from "@sanity/lib/image";
3 import { Product } from "@types/products";
4 import Image from "next/image";
5
6 const ProductDetails = async ({ params }: { params: { id: string } }) => {
7   const { id } = params;
8
9   if (!id) {
10     console.error("Product ID is undefined!");
11     return <div>Product ID not found</div>;
12   }
13
14   // Fetch the product by ID
15   const product: Product | null = await client.fetch(
16     `*[_type == "product" && _id == ${id}[0]]`,
17     { id }
18   );
19
20   if (!product) {
21     return <div>Product not found</div>;
22   }
23
24   const imageUrl = product.productImage ? urlFor(product.productImage).url() : null;
25
26   return (
27     <div className="max-w-4xl mx-auto px-4 py-8">
28       <h1 className="text-3xl font-bold mb-4">{product.title}</h1>
29       <div className="flex flex-col md:flex-row gap-6">
30         {imageUrl ? (
31           <Image
32             src={imageUrl}
33             alt={`Image of ${product.title}`}
34             width={400}
35             height={400}
36             className="rounded-lg"
37           />
38         ) : (
39           <p>No image available</p>
40         )}
41       <div>
42         <p><strong>Description:</strong> {product.description}</p>
43         <p><strong>Price:</strong> ${product.price}</p>
44         <p><strong>Discount:</strong> {product.discountPercentage || "N/A"}</p>
45         <p><strong>New:</strong> {product.isNew ? "Yes" : "No"}</p>
46       </div>
47     </div>
48   );
49 }
50
51 </div>
52 );
53 };
54
55 export default ProductDetails;
```
- Editor Status Bar:** Ln 1, Col 1 Spaces:4 UTF-8 CRLF (TypeScript JSX Go Live Pre

Single Product Screen Shot

localhost:3000/shop/product/YjFl01g1LQZHIZg27PvtKd

Furniro

Home Shop Blog Contact

Product Detail

Home > Shop > Product

Sleek Living



Description: Welcome to SleekLiving, where modern sophistication meets functional design for the ultimate living experience. Crafted for those who value style, convenience, and innovation, SleekLiving is a collection of premium home essentials that redefine the way you live. With an emphasis on minimalism and elegance, every product in the SleekLiving line is engineered to bring a refined touch to your home without compromising on performance or comfort. SleekLiving is all about transforming your space into a sanctuary of calm and sophistication. Featuring clean lines, contemporary aesthetics, and cutting-edge functionality, this collection offers versatile solutions for every room in your home. Whether you're upgrading your living room, kitchen, bedroom, or office, SleekLiving effortlessly blends into any decor, offering you the freedom to create a space that reflects your unique style. Designed with the modern homeowner in mind, SleekLiving products are not just about looks—they are built for practicality and ease of use. Each item is thoughtfully crafted

you're upgrading your living room, kitchen, bedroom, or office, SleekLiving effortlessly blends into any decor, offering you the freedom to create a space that reflects your unique style. Designed with the modern homeowner in mind, SleekLiving products are not just about looks—they are built for practicality and ease of use. Each item is thoughtfully crafted using high-quality materials that ensure durability and long-lasting performance. From smart furniture solutions to stylish accessories, SleekLiving brings together the best of contemporary design with everyday functionality. Key Features: Modern, minimalist design that enhances any living space. Versatile range of products for every room in your home. High-quality materials for durability and long-lasting performance. Innovative features that combine form and function. Perfect for those who appreciate style and practicality. Transform your home into a sleek, stylish haven with SleekLiving—where contemporary design meets everyday convenience.

Price: \$300
Discount: N/A
New: Yes

Furniro
400 University Drive Suite 200 Coral Gables, FL 33134 USA

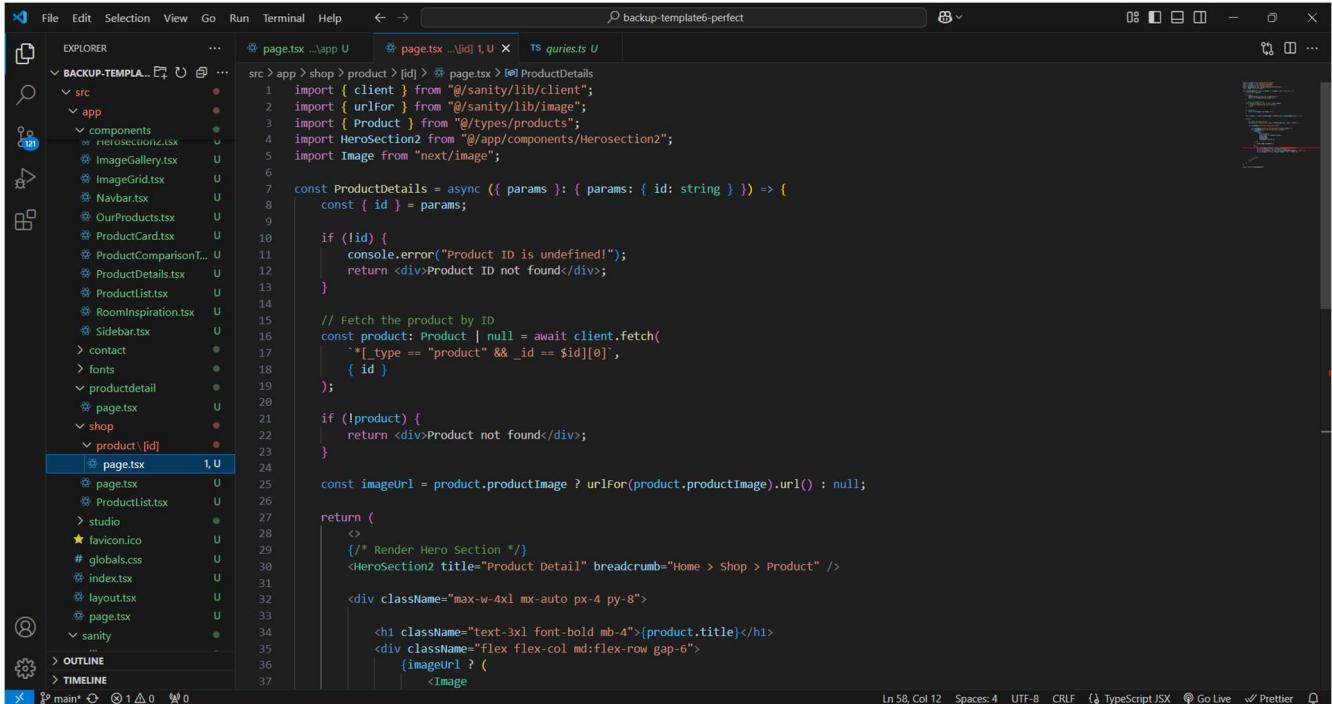
LINKS
Home Shop Blog Contact

Help
Payment Options Returns Privacy Policy

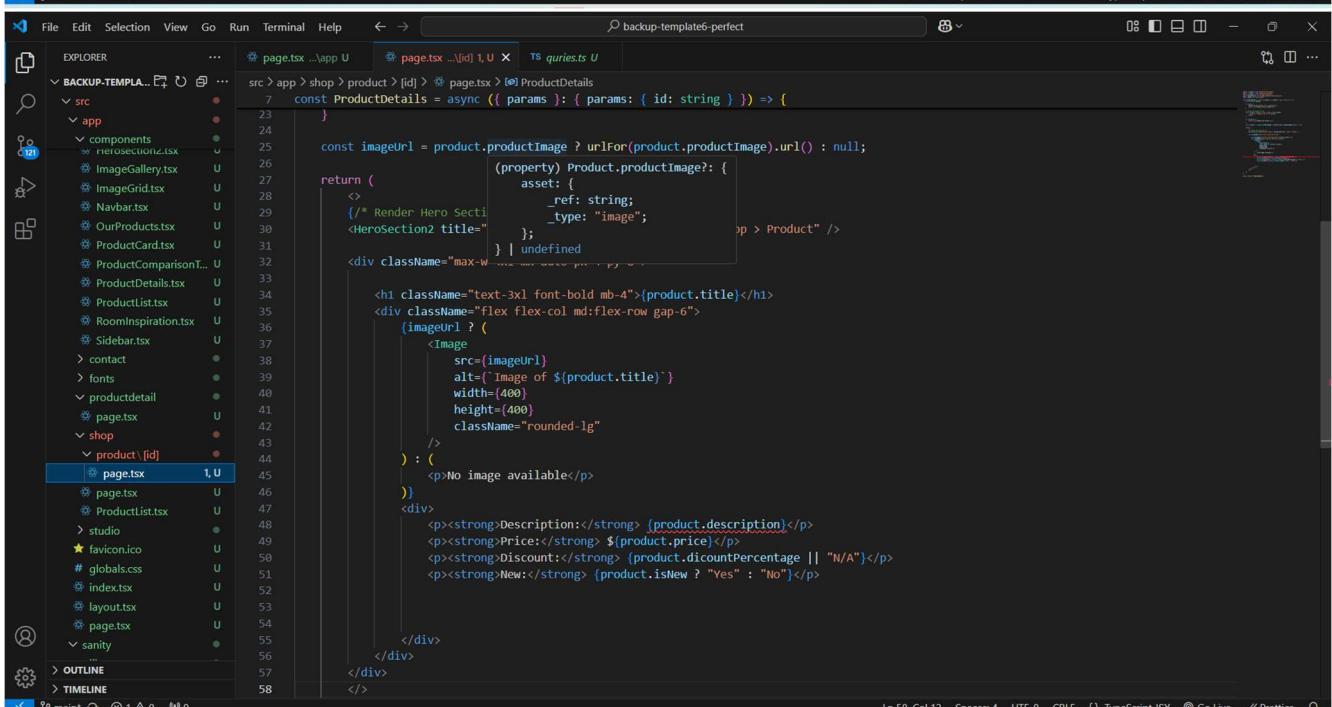
Enter Your Email Address **SUBSCRIBE**

2023 Furniro. All rights reserved.

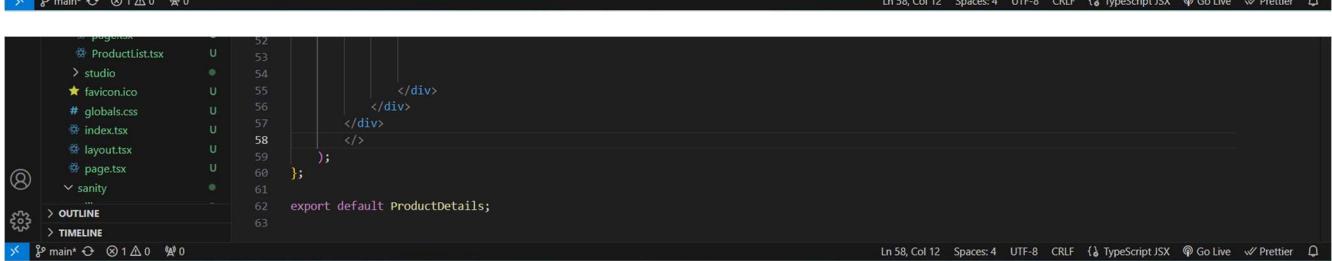
Here is single product code



```
src > app > shop > product > [id] > page.tsx > ProductDetails
1 import { client } from "@/sanity/lib/client";
2 import { urlFor } from "@/sanity/lib/image";
3 import { Product } from "@types/products";
4 import HeroSection2 from "@/app/components/HeroSection2";
5 import Image from "next/image";
6
7 const ProductDetails = async ({ params }: { params: { id: string } }) => {
8   const { id } = params;
9
10  if (!id) {
11    console.error("Product ID is undefined!");
12    return <div>Product ID not found</div>;
13  }
14
15  // Fetch the product by ID
16  const product: Product | null = await client.fetch(
17    `*[_type == "product" && _id == $id][0]`,
18    { id }
19  );
20
21  if (!product) {
22    return <div>Product not found</div>;
23  }
24
25  const imageUrl = product.productImage ? urlFor(product.productImage).url() : null;
26
27  return (
28    <div className="max-w-4xl mx-auto px-4 py-8">
29      <h1 className="text-3xl font-bold mb-4">{product.title}</h1>
30      <div className="flex flex-col md:flex-row gap-6">
31        {imageUrl ? (
32          <Image
33            alt="Image of ${product.title}"
34            width={400}
35            height={400}
36            className="rounded-lg"
37          />
38        ) : (
39          <p>No image available</p>
40        )}
41        <div>
42          <p><strong>Description:</strong> {product.description}</p>
43          <p><strong>Price:</strong> ${product.price}</p>
44          <p><strong>Discount:</strong> {product.discountPercentage} || "N/A"</p>
45          <p><strong>New:</strong> {product.isNew ? "Yes" : "No"}</p>
46        </div>
47      </div>
48    </div>
49  );
50
51  </div>
52
53  </div>
54
55  </div>
56
57  </div>
58
59  );
60
61
62  export default ProductDetails;
```

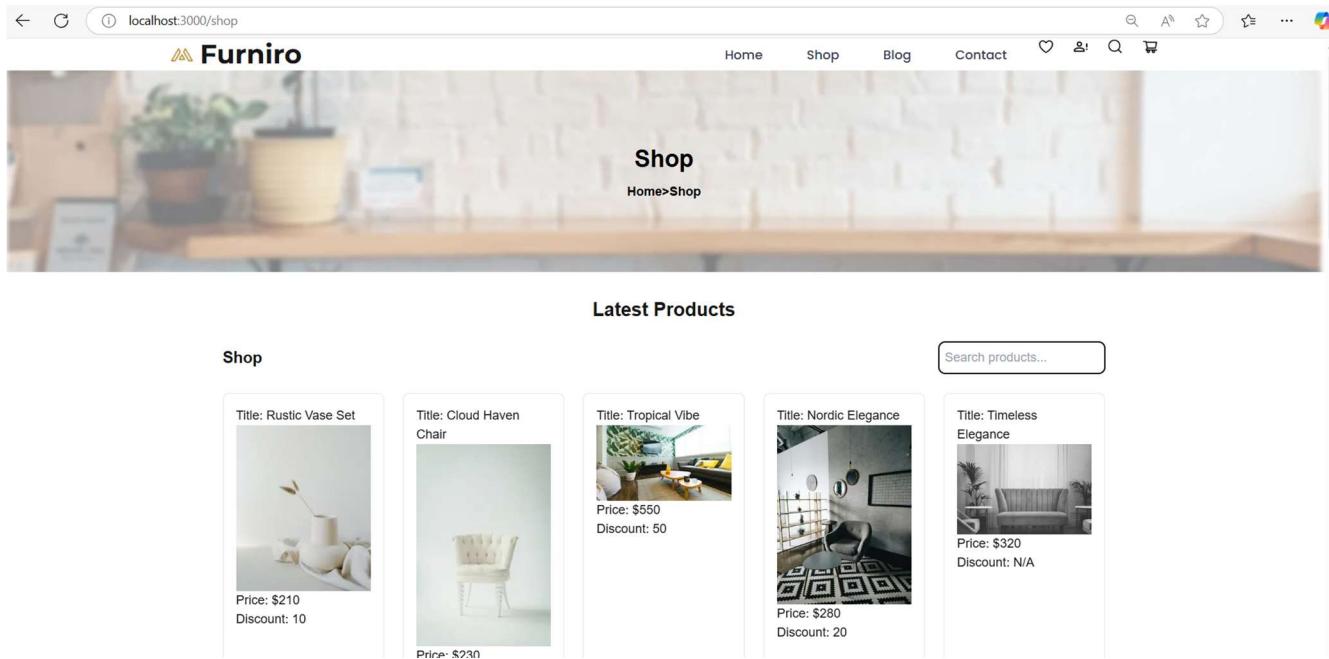


```
src > app > shop > product > [id] > page.tsx > ProductDetails
1 const ProductDetails = async ({ params }: { params: { id: string } }) => {
2
3  const imageUrl = product.productImage ? urlFor(product.productImage).url() : null;
4
5  return (
6    <div className="max-w-4xl mx-auto px-4 py-8">
7      <h1 className="text-3xl font-bold mb-4">{product.title}</h1>
8      <div className="flex flex-col md:flex-row gap-6">
9        {imageUrl ? (
10          <Image
11            alt="Image of ${product.title}"
12            width={400}
13            height={400}
14            className="rounded-lg"
15          />
16        ) : (
17          <p>No image available</p>
18        )}
19        <div>
20          <p><strong>Description:</strong> {product.description}</p>
21          <p><strong>Price:</strong> ${product.price}</p>
22          <p><strong>Discount:</strong> {product.discountPercentage} || "N/A"</p>
23          <p><strong>New:</strong> {product.isNew ? "Yes" : "No"}</p>
24        </div>
25      </div>
26    </div>
27  );
28
29  </div>
30
31  </div>
32
33  </div>
34
35  </div>
36
37  );
38
39
40  export default ProductDetails;
```



```
src > app > shop > product > [id] > page.tsx > ProductDetails
1 const ProductDetails = async ({ params }: { params: { id: string } }) => {
2
3  const imageUrl = product.productImage ? urlFor(product.productImage).url() : null;
4
5  return (
6    <div className="max-w-4xl mx-auto px-4 py-8">
7      <h1 className="text-3xl font-bold mb-4">{product.title}</h1>
8      <div className="flex flex-col md:flex-row gap-6">
9        {imageUrl ? (
10          <Image
11            alt="Image of ${product.title}"
12            width={400}
13            height={400}
14            className="rounded-lg"
15          />
16        ) : (
17          <p>No image available</p>
18        )}
19        <div>
20          <p><strong>Description:</strong> {product.description}</p>
21          <p><strong>Price:</strong> ${product.price}</p>
22          <p><strong>Discount:</strong> {product.discountPercentage} || "N/A"</p>
23          <p><strong>New:</strong> {product.isNew ? "Yes" : "No"}</p>
24        </div>
25      </div>
26    </div>
27  );
28
29  </div>
30
31  </div>
32
33  </div>
34
35  </div>
36
37  );
38
39
40  export default ProductDetails;
```

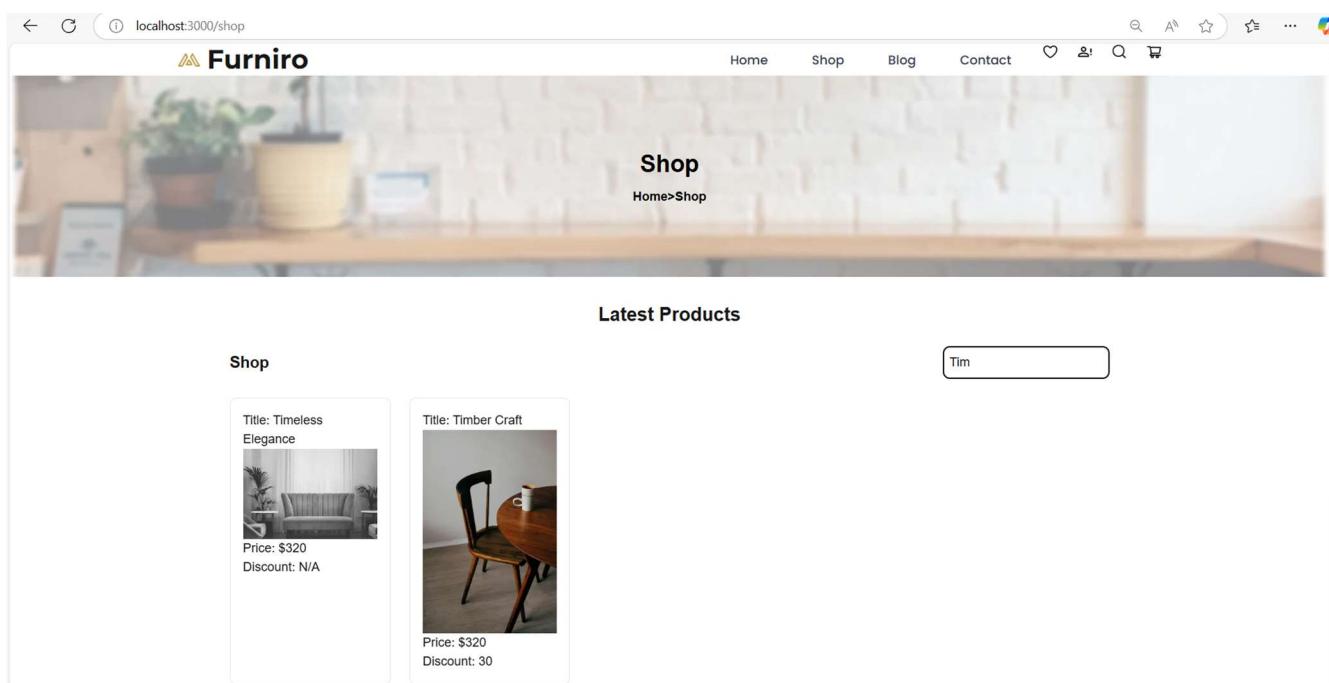
Now added Search Button of Product within Product List



A screenshot of a web browser displaying the 'Shop' page of the 'Furniro' website. The URL in the address bar is 'localhost:3000/shop'. The page features a header with the 'Furniro' logo and navigation links for Home, Shop, Blog, and Contact. A search icon is located in the top right corner. The main content area is titled 'Latest Products' and displays five product cards. Each card includes a thumbnail image, the product title, price, and discount information. A search input field labeled 'Search products...' is positioned on the right side of the product grid.

Title	Price	Discount
Rustic Vase Set	\$210	10
Cloud Haven Chair	\$230	
Tropical Vibe	\$550	50
Nordic Elegance	\$280	20
Timeless Elegance	\$320	N/A

Search Specific Product within Product List

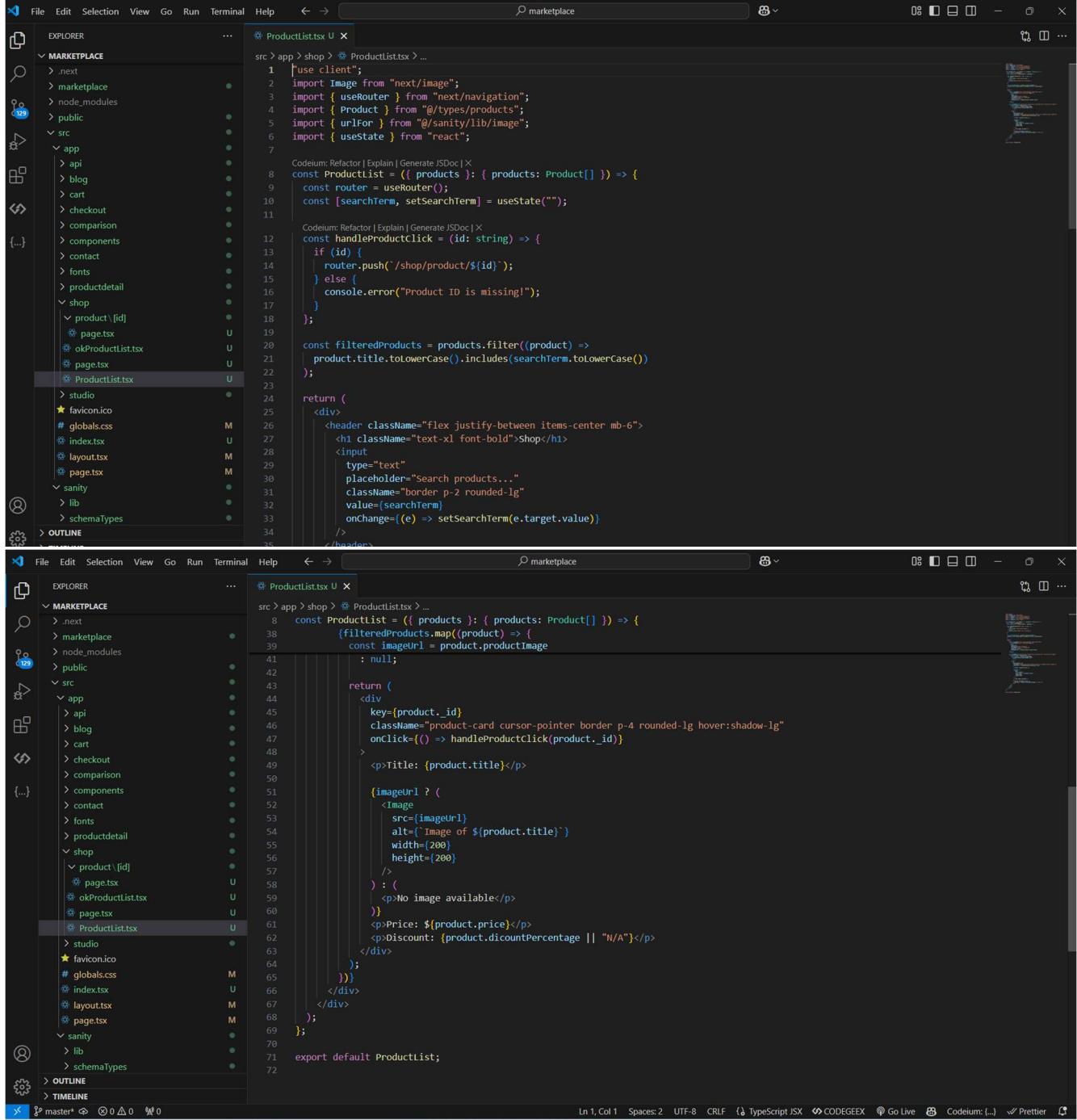


A screenshot of a web browser displaying the 'Shop' page of the 'Furniro' website. The URL in the address bar is 'localhost:3000/shop'. The page features a header with the 'Furniro' logo and navigation links for Home, Shop, Blog, and Contact. A search icon is located in the top right corner. The main content area is titled 'Latest Products' and displays two product cards. A search input field containing the text 'Tim' is positioned on the right side of the product grid.

Title	Price	Discount
Timeless Elegance	\$320	N/A
Timber Craft	\$320	30

Search Product within Product List Code

Here you will observe two files of `ProductList.tsx` which is because I always make copy of correct file to experiment new page or content and after successful completion of testing I remove previous file and update new one.



The image shows two side-by-side code editors, likely from the Codeium IDE, displaying two versions of the `ProductList.tsx` component. Both editors have a dark theme and show the same file structure in the Explorer sidebar, including `.next`, `marketplace`, `node_modules`, `public`, `src`, `app`, `comparison`, `components`, `contact`, `fonts`, `productdetail`, `shop`, `product\{id\}`, `page.tsx`, `okProductList.tsx`, `page.tsx`, and `ProductList.tsx`.

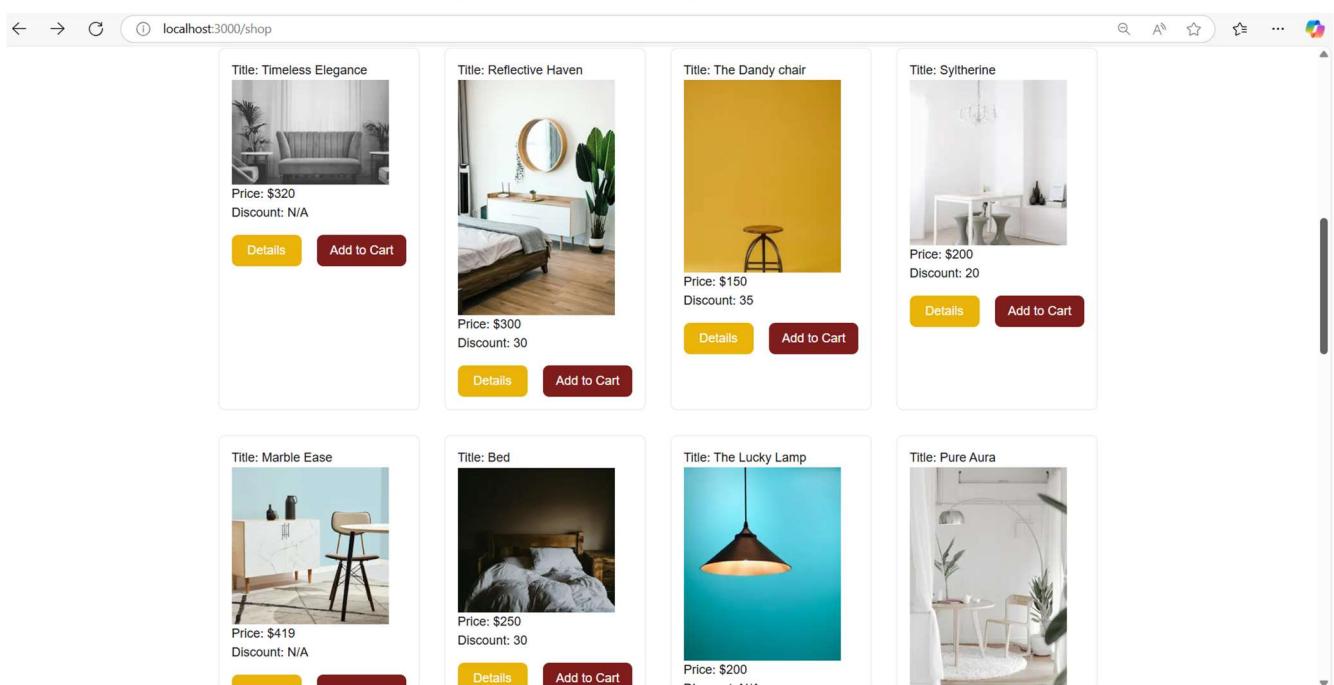
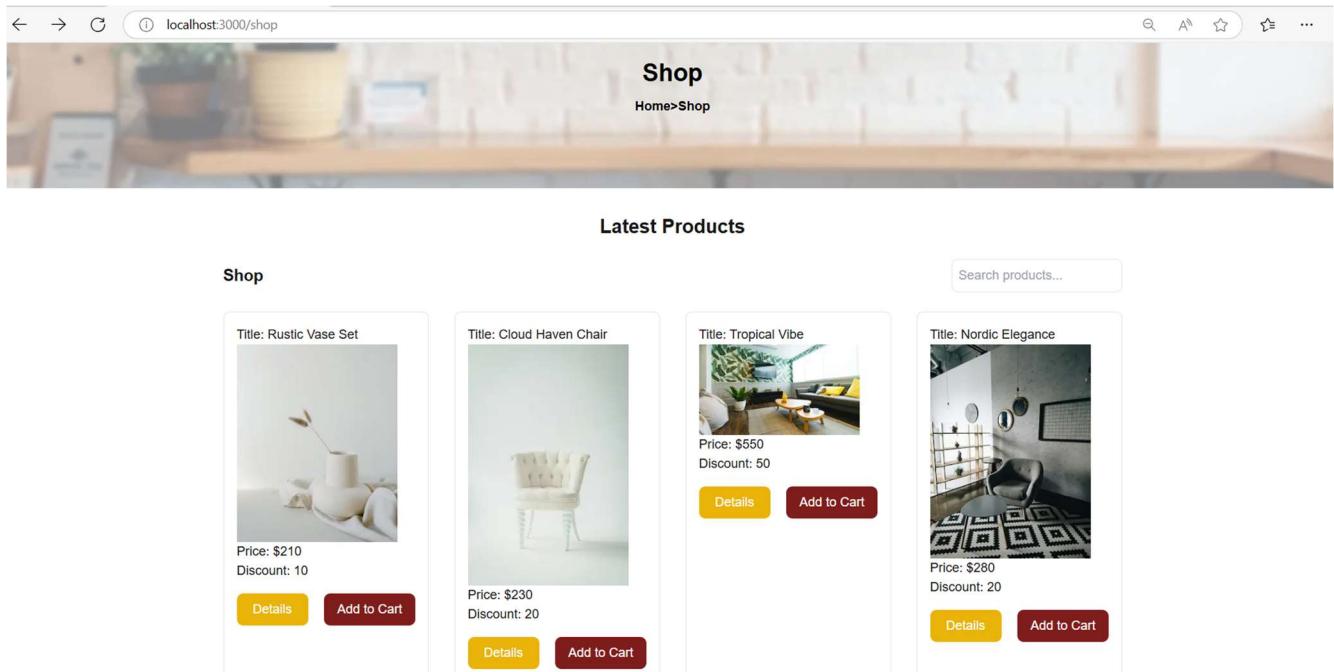
Editor 1 (Left):

```
src > app > shop > ProductList.tsx ...
1  |use client";
2  |import Image from "next/image";
3  |import { useRouter } from "next/navigation";
4  |import { Product } from "@types/products";
5  |import { urlFor } from "@/sanity/lib/image";
6  |import { useState } from "react";
7
8  Codeium: Refactor | Explain | Generate JSDoc | X
9  const ProductList = ({ products }: { products: Product[] }) => {
10   const router = useRouter();
11   const [searchTerm, setSearchTerm] = useState("");
12
13   Codeium: Refactor | Explain | Generate JSDoc | X
14   const handleProductClick = (id: string) => {
15     if (id) {
16       router.push(`/shop/product/${id}`);
17     } else {
18       console.error("Product ID is missing!");
19     }
20
21   const filteredProducts = products.filter((product) =>
22     product.title.toLowerCase().includes(searchTerm.toLowerCase())
23   );
24
25   return (
26     <div>
27       <header className="flex justify-between items-center mb-6">
28         <h1 className="text-xl font-bold">Shop</h1>
29         <input
30           type="text"
31           placeholder="Search products..."
32           className="border p-2 rounded-lg"
33           value={searchTerm}
34           onChange={(e) => setSearchTerm(e.target.value)}
35         />
36       </header>
```

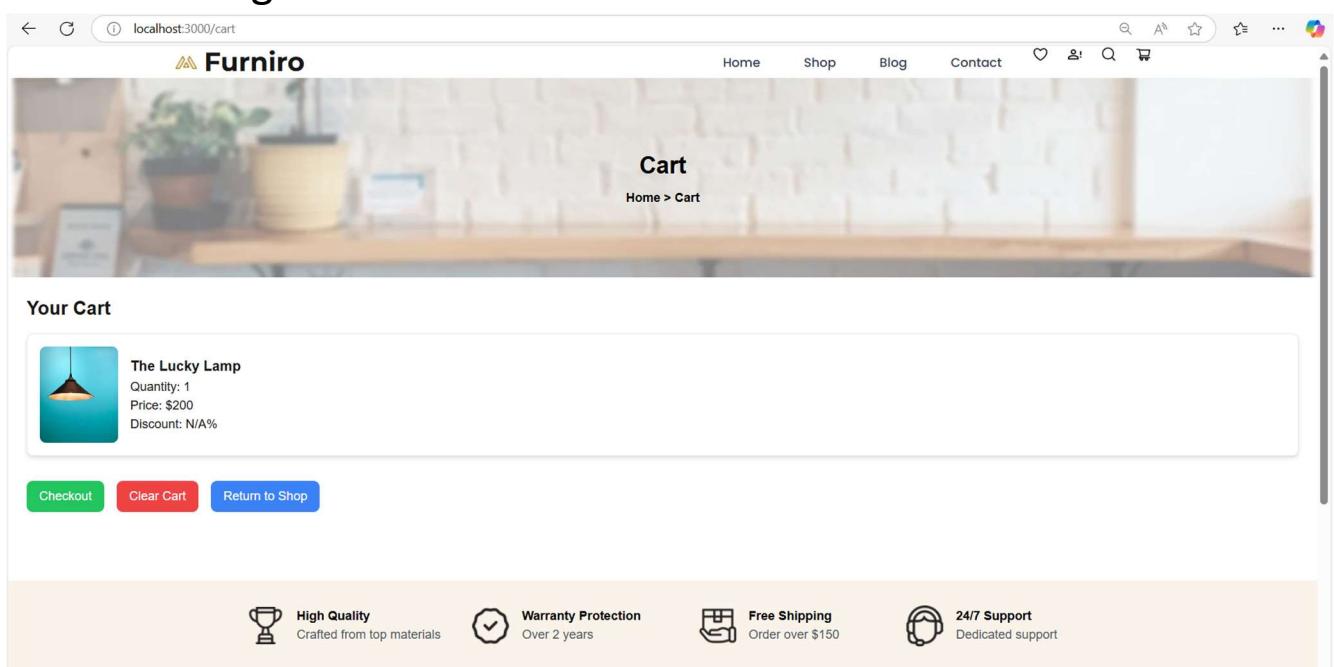
Editor 2 (Right):

```
src > app > shop > ProductList.tsx ...
8  const ProductList = ({ products }: { products: Product[] }) => {
38   const filteredProducts = products.map((product) => {
39     const imageUrl = product.productImage
40     : null;
41
42     return (
43       <div
44         key={product.id}
45         className="product-card cursor-pointer border p-4 rounded-lg hover:shadow-lg"
46         onClick={() => handleProductClick(product.id)}
47       >
48         <p>Title: {product.title}</p>
49
50         {imageUrl ? (
51           <Image
52             src={imageUrl}
53             alt={`Image of ${product.title}`}
54             width={200}
55             height={200}
56           />
57         ) : (
58           <p>No image available</p>
59         )}
60         <p>Price: ${product.price}</p>
61         <p>Discount: ${product.discountPercentage} || "N/A"</p>
62       </div>
63     );
64   );
65
66   </div>
67 </div>
68 );
69
70 export default ProductList;
```

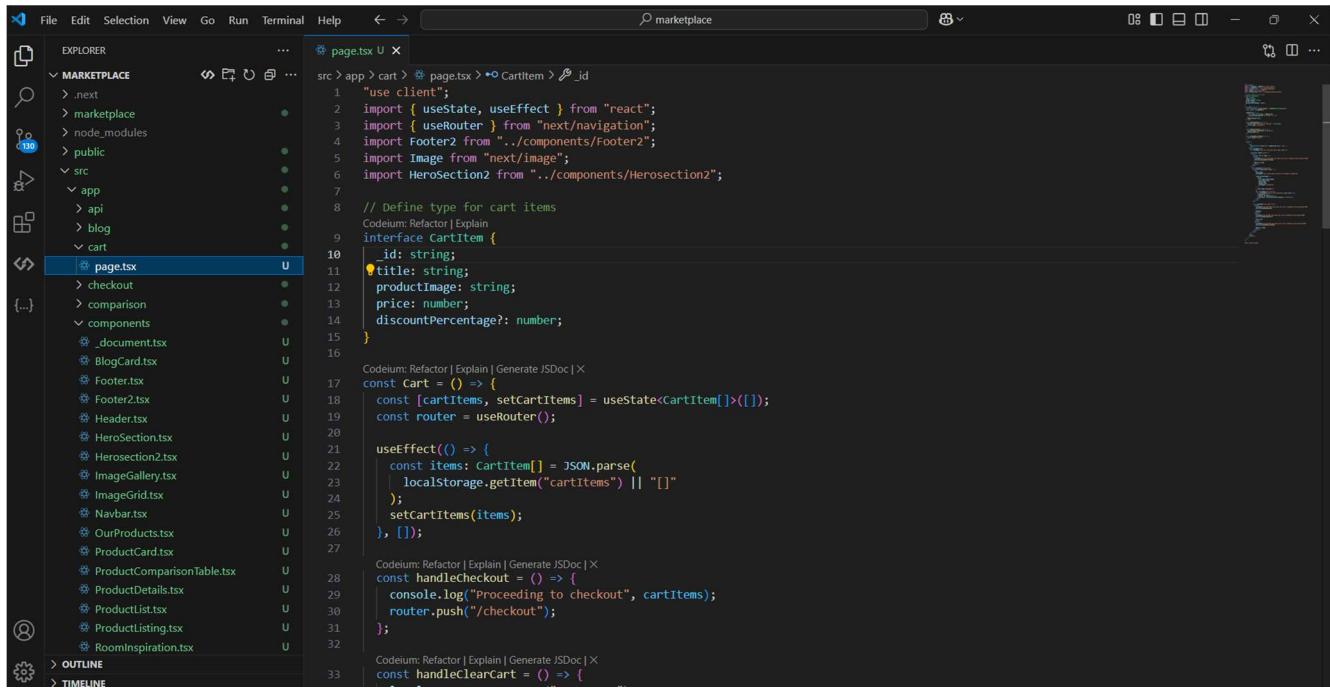
Now Added, Add to Cart and Product Detail Buttons under each product of Product List



Clicking on Add to Cart button result will be shown here



Here is the code of /cart/page.tsx



The screenshot shows a code editor interface with the following details:

- File Path:** src/app/cart/page.tsx
- Code Editor Content:**

```
src > app > cart > page.tsx > CartItem > _id
1  "use client";
2  import { useState, useEffect } from "react";
3  import { useRouter } from "next/navigation";
4  import Footer2 from "../components/Footer2";
5  import Image from "next/image";
6  import HeroSection2 from "../components/HeroSection2";
7
8 // Define type for cart items
9 interface CartItem {
10   _id: string;
11   title: string;
12   productImage: string;
13   price: number;
14   discountPercentage?: number;
15 }
16
17 Codeium: Refactor | Explain | Generate JSDoc | X
18 const Cart = () => {
19   const [cartItems, setCartItems] = useState<CartItem[]>([]);
20   const router = useRouter();
21
22   useEffect(() => {
23     const items: CartItem[] = JSON.parse(
24       localStorage.getItem("cartItems") || "[]"
25     );
26     setCartItems(items);
27   }, []);
28
29   Codeium: Refactor | Explain | Generate JSDoc | X
30   const handleCheckout = () => {
31     console.log("Proceeding to checkout", cartItems);
32     router.push("/checkout");
33   };
34
35   Codeium: Refactor | Explain | Generate JSDoc | X
36   const handleClearCart = () => {
37     localStorage.removeItem("cartItems");
38     setCartItems([]);
39   };
40 }
```
- Explorer View:** Shows the project structure with files like .next, marketplace, node_modules, src, app, api, blog, cart, and various component files.
- Bottom Status Bar:** Displays "marketplace" and other standard editor icons.

The image displays three vertically stacked screenshots of a code editor interface, likely VS Code, showing the progression of a React component named 'page.tsx'. The code is part of a 'MARKETPLACE' project structure.

Top Screenshot: Shows the initial state of the component. It defines a 'Cart' function that handles clearing the cart and returning to the shop. It checks if the cart is empty and displays a message or a button to return to the shop. If the cart is not empty, it maps over the items to render a grid of products.

```
const Cart = () => {
  const handleClearCart = () => {
    localStorage.removeItem("cartItems");
    setCartItems([]);
  };

  const handleReturnToShop = () => {
    router.push("/shop");
  };

  return (
    <>
      <div>
        <HeroSection2 title="Cart" breadcrumb="Home > Cart" />
      </div>
      <div className="p-6">
        <h1 className="text-2xl font-bold mb-4">Your Cart</h1>

        {cartItems.length === 0 ? (
          <div>
            <p>Your cart is empty.</p>
            <button
              className="mt-4 bg-blue-500 text-white py-2 px-4 rounded-lg hover:bg-blue-600"
              onClick={handleReturnToShop}
            >
              Return to Shop
            </button>
          </div>
        ) : (
          <div className="grid gap-4">
            {cartItems.map((item, index) => (
              <div
                key={index}
                className="flex items-center border p-4 rounded-lg shadow-md"
              >
                {item.productImage ? (

```

Middle Screenshot: Shows the component after adding image rendering logic. It checks if each item has a 'productImage' and renders it using the 'Image' component from 'next/image'. If no image is available, it displays a placeholder message.

```
                {item.productImage ? (
                  <Image
                    src={item.productImage}
                    alt={item.title}
                    width={100}
                    height={100}
                    className="rounded-lg"
                  />
                ) : (
                  <p>No image available</p>
                )}
              </div>
            ))}

            <div className="flex gap-4 mt-4">
              <button
                className="bg-green-500 text-white py-2 px-4 rounded-lg hover:bg-green-600"
                onClick={handleCheckout}
              >
                Checkout
              </button>
              <button
                className="bg-red-500 text-white py-2 px-4 rounded-lg hover:bg-red-600"
                onClick={handleClearCart}
              >
                Clear Cart
              </button>
            </div>
          </div>
        )
      );
    );
};

export default Cart;
```

Bottom Screenshot: Shows the final state of the component. The 'Checkout' and 'Clear Cart' buttons have been swapped in position. The 'Return to Shop' button now has a blue background color.