# Day 4 - Building Dynamic Frontend Components For Hackathon Ecommerce Website

💡 **Mutahir Hussain Shah - 0062580**

## Main Objective

On Day 4, the focus is on designing and developing dynamic frontend

components that display marketplace data fetched from Sanity CMS or APIs. This task emphasizes creating modular, reusable components and follows real-world practices to build scalable, responsive web applications.

## 1 . Build Dynamic Frontend Components To Display Data From Sanity CMS

To build dynamic frontend components for my Hekto E-Commerce project, I connected my Next.js app with Sanity CMS. I configured the Sanity client and

Day 4 - Building Dynamic Frontend Components For Hekto Ecommerce 1

used GROQ queries to fetch essential data, including product ID, image, name, stock, and price.

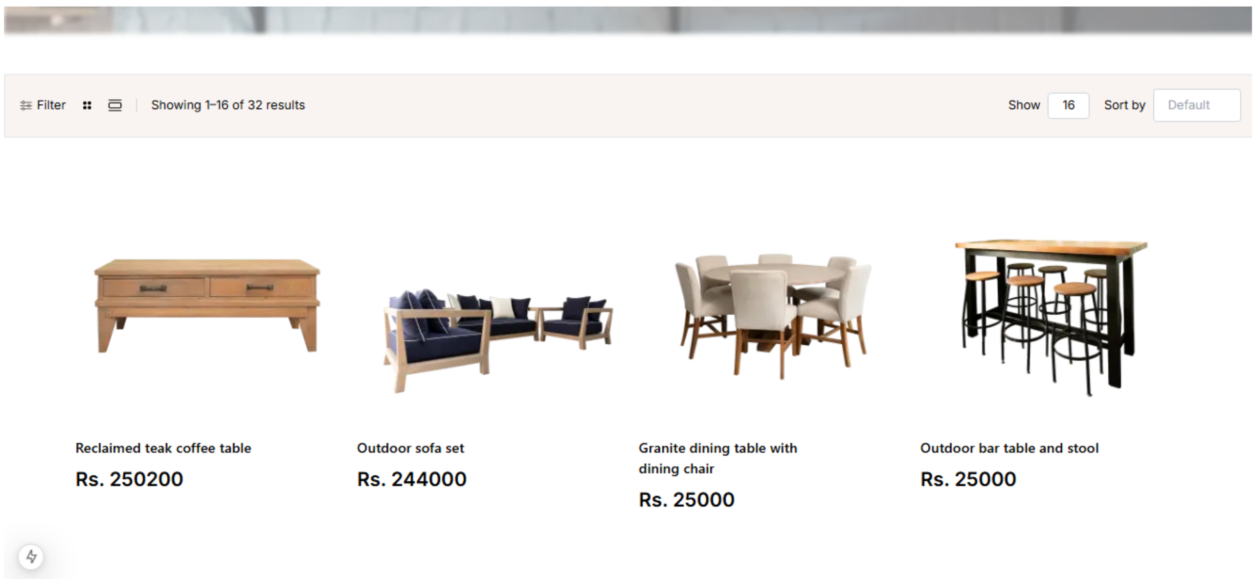
```
const MainPage = async ({ params }: Props) => {
  const query = `
    *[_type == "product" && id == $id] {
      _id,
      name,
      price,
      description,
      "image": image.asset->url,
      rating,
      reviews,
      id,
      stockInHand,
      size,
      color

    }
  `;
```

After fetching data, I mapped it into frontend components, enabling flexible, dynamic displays that automatically reflect the latest information without manual code updates. Utilizing Sanity’s real-time data enhanced the user experience with consistently fresh and easily editable content.

```
{/* Main Content */}
      <div className="container mx-auto px-4 py-8">
        <div className="flex flex-wrap md:flex-nowrap gap-8">
          {/* Left: Image */}
          <div className="w-full md:w-1/2 flex flex-col md:flex-
            <div className="rounded-lg overflow-hidden">
              {product.image ? (
                <Image
                  src={product.image || "/placeholder.svg"}
                  alt={product.name}
                  width={550}
                  height={550}
                  className="object-cover"
                />
              ) : (
                <div className="w-full h-64 bg-gray-200 flex ite
                  No Image Available
                </div>
              )}
            </div>
          </div>

          {/* Right: Product Details */}
          <div className="w-full md:w-1/2 pt-15">
            <h1 className="text-[40px] font-bold text-gray-800">
              {product.name}
            </h1>
            <p className="text-xl lg:text-2xl font-[600] text-[#
              Rs. {product.price}
            </p>

            {/* Ratings */}
            <div className="flex items-center mt-4">
              <div className="flex text-yellow-400">
                <span className="text-[25px]">★</span>
                <span className="text-[25px]">★</span>
                <span className="text-[25px]">★</span>
                <span className="text-[25px]">★</span>
                <span className="text-[25px]">☆</span>
              </div>
              <p className="text-sm text-gray-600 ml-10">
                | {product.reviews} Customer Review
              </p>

              <StockDisplay productId={product.id} initialStock=
            </div>

            {/* Description */}
            <p className="text-sm text-black mt-4">{product.desc

            {/* Size Options */}
            <div className="mt-6">
              <p className="text-sm font-semibold text-gray-800"
              <div className="flex gap-2 mt-2">
                <button className="px-3 py-1 border rounded-lg t

              </div>
            </div>

            {/* Color Options */}
            <div className="mt-6">
              <p className="text-sm font-semibold text-gray-800"

              <Color/>
            </div>

            {/* Quantity and Add to Cart */}
            <div className="flex items-center mt-6">
               <AddToCartButton product={product}/>
              <Button name="Check Out" />
            </div>
```

```
          {/* Additional Info */}
          <div className="mt-8">
            <p className="text-md text-gray-600">
              SKU <span className="ml-16">: SS001</span>
            </p>
            <p className="text-md text-gray-600 mt-2">
              Category<span className="ml-8">: {product.catego
            </p>
            <p className="text-md text-gray-600 mt-2">
              Tags<span className="ml-16">: Sofa, Chair, Home,
            </p>
          </div>

          {/* Share and Wishlist */}
          <div className="flex justify-start items-center mt-
            <p className="text-md text-gray-600">Share</p>
            <div className="flex justify-center items-center 
              <Image src={facebook || "/placeholder.svg"} alt=
              <Image src={linkdin || "/placeholder.svg"} alt="
              <Image src={twitter || "/placeholder.svg"} alt="
            </div>
          </div>
        </div>
      </div>
    </div>

    <ProductDetails />
  </div>
```

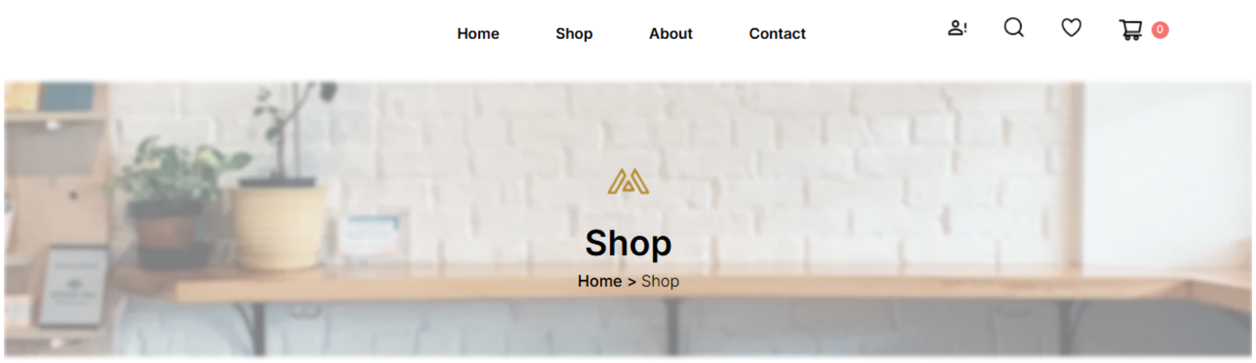
## As a result

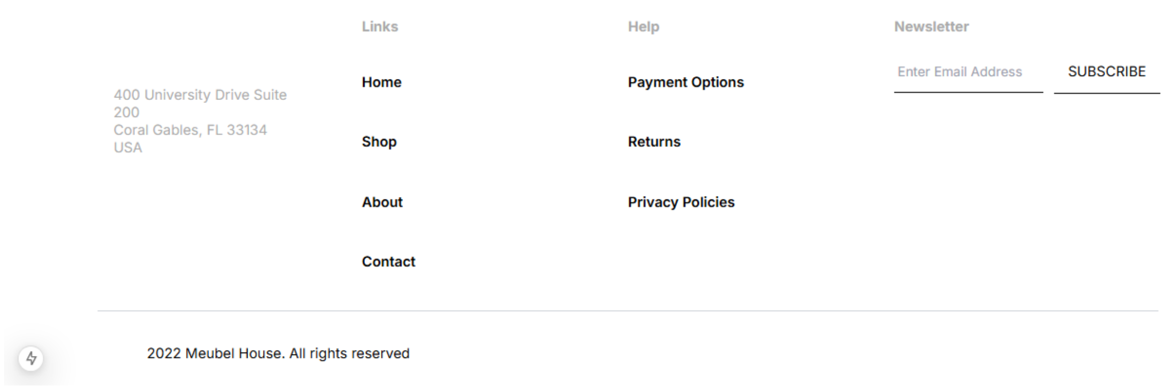

Filter
Showing 1–16 of 32 results
Show
16
Sort by
Default
Reclaimed teak coffee table
Rs. 250200
Outdoor sofa set
Rs. 244000
Granite dining table with dining chair
Rs. 25000
Outdoor bar table and stool
Rs. 25000

## 2. Implement reusable and modular components.

I created components such as footer , banner, Featured Producsts  and individual product displays, ensuring they can be reused across multiple pages without rewriting the same code.

By creating a reusable Products components, I can display individual products and sections of code  anywhere within the app without duplicating the code. This keeps the codebase modular and easier to maintain.





400 University Drive Suite 200
Coral Gables, FL 33134
USA

Links

Home

Shop

About

Contact

Help

Payment Options

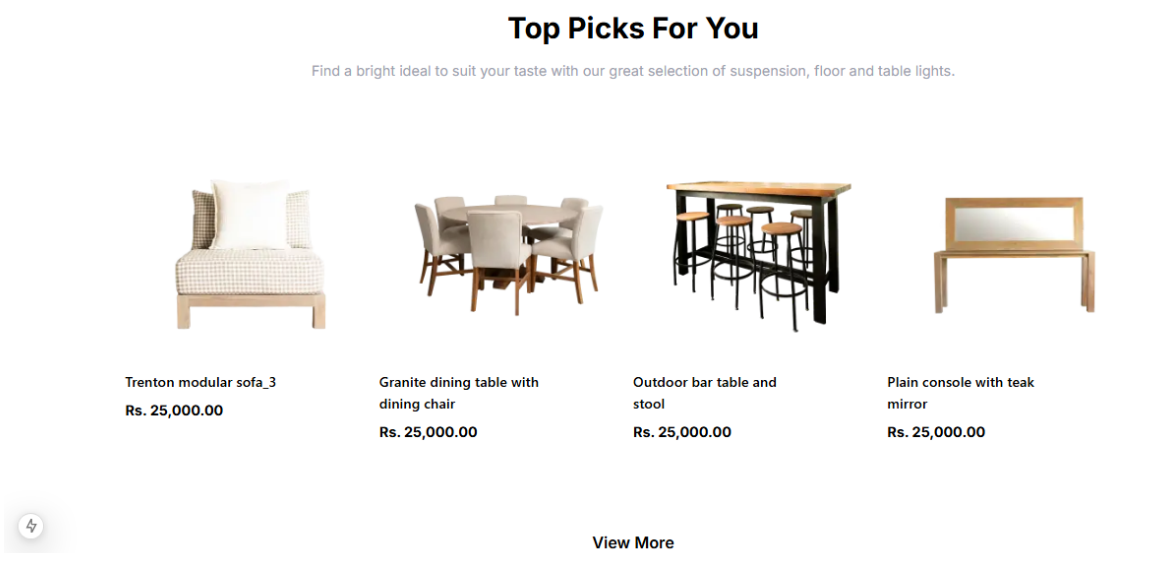
Returns

Privacy Policies

Newsletter

Enter Email Address SUBSCRIBE

2022 Meubel House. All rights reserved



Top Picks For You

Find a bright ideal to suit your taste with our great selection of suspension, floor and table lights.

Trenton modular sofa_3
Rs. 25,000.00

Granite dining table with dining chair
Rs. 25,000.00

Outdoor bar table and stool
Rs. 25,000.00

Plain console with teak mirror
Rs. 25,000.00

View More

## 3. Understand and apply state management techniques.

I used **React's useState** and **useEffect** hooks to manage the product data.

**useState** is used to declare a state variable that holds the product data (like name, price etc).

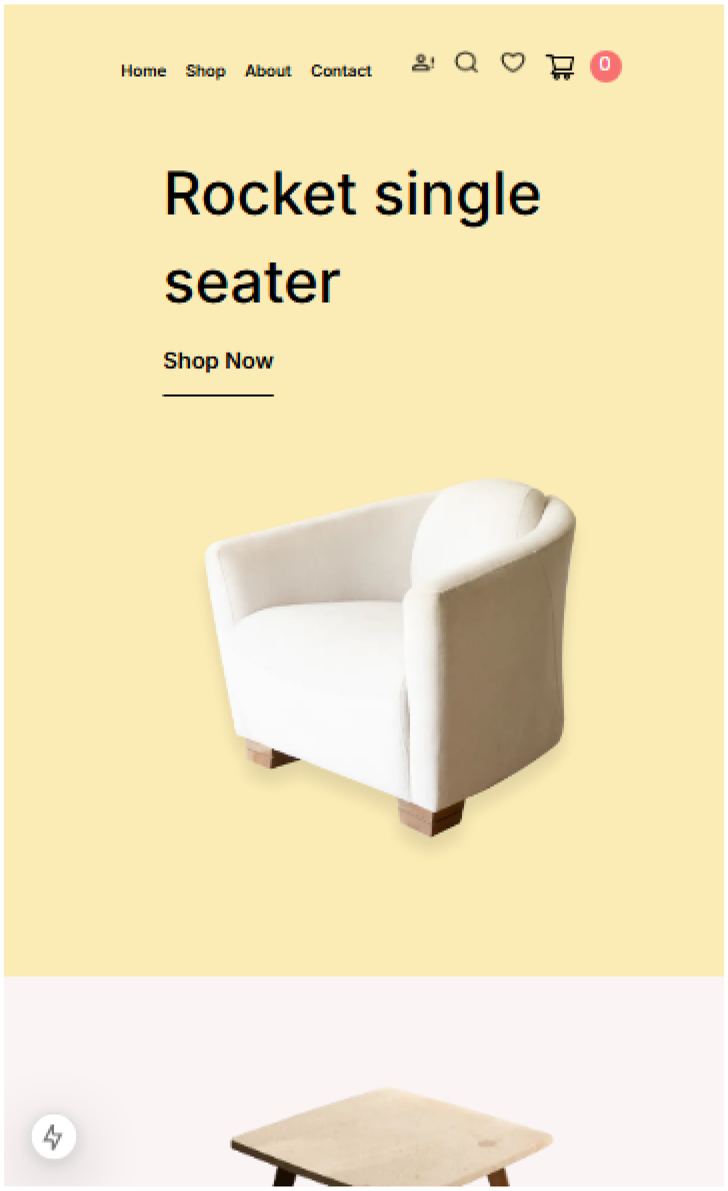
```
const [products, setProducts] = useState<ProductType[]>([]);
```

**useEffect** is used to fetch the product data from the backend (Sanity CMS) when the component mounts.

```
      useEffect(() => {

// Fetch products when the component mounts

const getProducts = async () => {

const fetchedProducts = await fetchProducts();

const filteredProducts = fetchedProducts.filter(product => S

// Update the state with the fetched products setProducts(filte

getProducts();
```

## 4. Learn the importance of responsive design and UX/UI best practices.

Throughout this process, I ensured that the product listing components were responsive on all devices, providing an optimal user experience for mobile and desktop users alike.



## 5. How I Replicated Professional Workflows for Real-World Client Projects:

### 1. Modular Components:

I followed the best practice of breaking the project into reusable components, making the code scalable and easier to maintain.

Day 4 - Building Dynamic Frontend Components For Hekto Ecommerce 6

## 2. State Management with Hooks:

I used React’s useState and useEffect to manage the application state efficiently. For larger projects, I also implemented Redux for state management to handle complex states across different components.

## Error Handling and Edge Case Management:

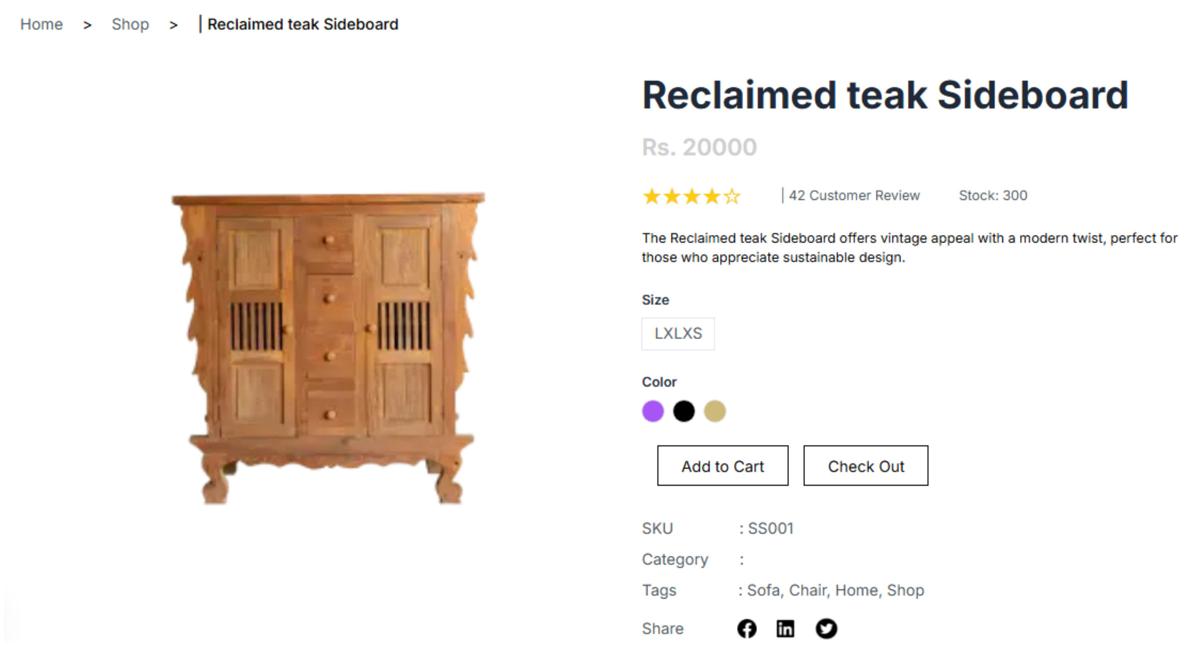
For a production-ready application, I incorporated error handling and managed edge cases

```
      if (typeof window !== "undefined") {

try {

const serializedWishlist = localStorage.getItem('wishlist retur

console.error('Error loading wishlist from localStorage:', retu

}

}
```

## 6. Create Individual Product Detail Pages Using Dynamic Routing in Next.js

To create individual product detail pages using dynamic routing in Next.js, I created a dynamic route in the src/app/FeaturedProduct/[id]/page.tsx file. This structure allows for each product to have its own unique URL based on the product ID, and it dynamically loads the product details from a backend API or CMS (such as Sanity).

I included all Necessary Fields as you can see in the picture



Reclaimed teak Sideboard

Rs. 20000

★★★★☆ | 42 Customer Review Stock: 300

The Reclaimed teak Sideboard offers vintage appeal with a modern twist, perfect for those who appreciate sustainable design.

Size

LXLXS

Color

SKU : SS001
Category :
Tags : Sofa, Chair, Home, Shop

## 7. Cart Component:

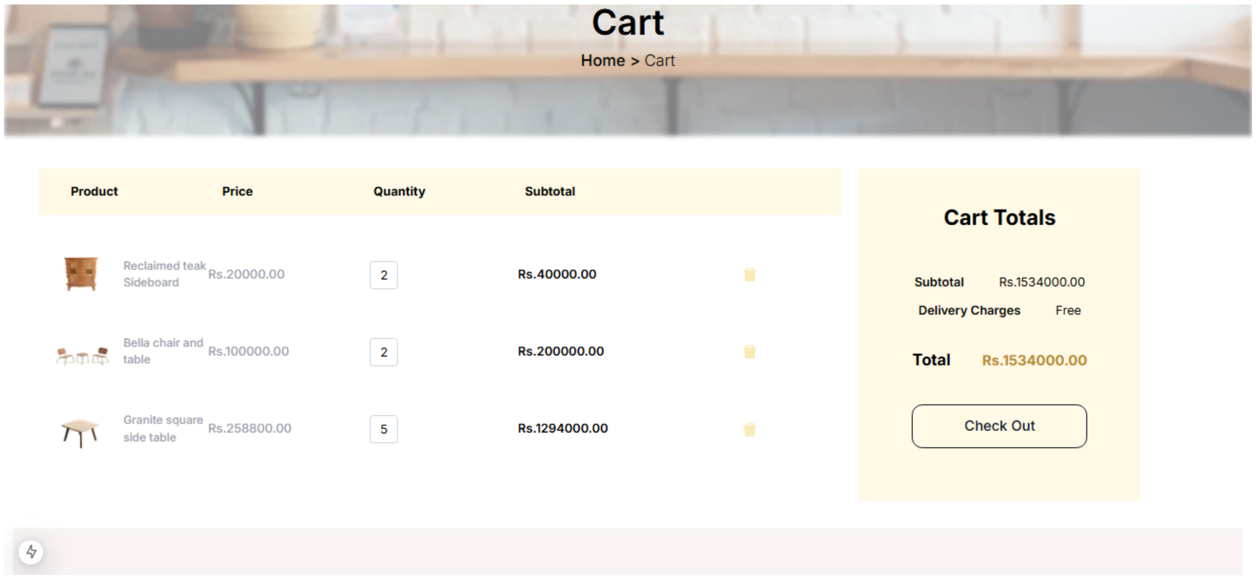
## 1. Displaying Added Items, Quantity, and Total Price:

●

The Cart component tracks the products that are added to the cart, showing key details such as the product name, price, quantity, and the total price.

●

It displays the products in a list format with options to modify the quantity or remove the product from the cart.



Cart

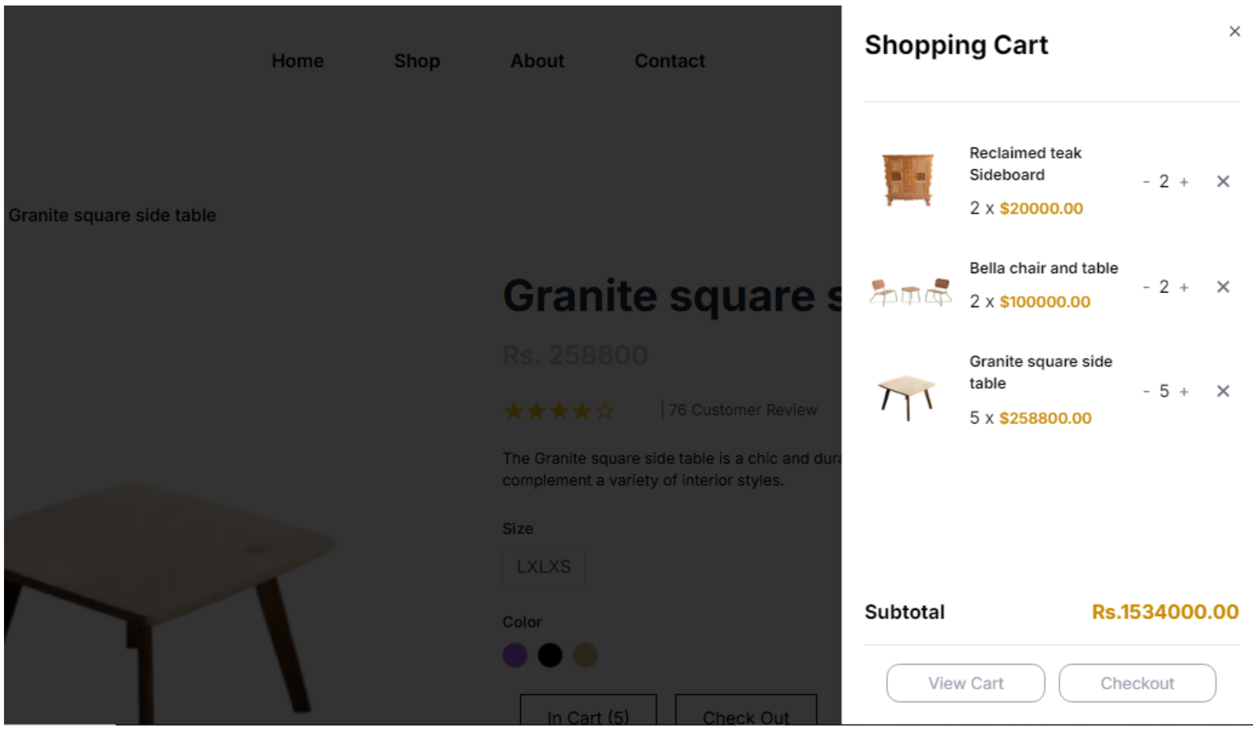
Home > Cart

| Product | Price | Quantity | Subtotal |
|---|---|---|---|
| Reclaimed teak Sideboard | Rs.20000.00 | 2 | Rs.40000.00 |
| Bella chair and table | Rs.100000.00 | 2 | Rs.200000.00 |
| Granite square side table | Rs.258800.00 | 5 | Rs.1294000.00 |

Cart Totals

| | |
|---|---|
| Subtotal | Rs.1534000.00 |
| Delivery Charges | Free |
| Total | Rs.1534000.00 |

●


Home
Shop
About
Contact

Granite square side table

Granite square s
Rs. 258800
★★★★☆ | 76 Customer Review
The Granite square side table is a chic and dur
complement a variety of interior styles.
Size
LXLXS
Color
In Cart (5)
Check Out

Shopping Cart

| | | | |
|---|---|---|---|
| Reclaimed teak Sideboard | 2 x $20000.00 | - 2 + | ✕ |
| Bella chair and table | 2 x $100000.00 | - 2 + | ✕ |
| Granite square side table | 5 x $258800.00 | - 5 + | ✕ |

Subtotal Rs.1534000.00

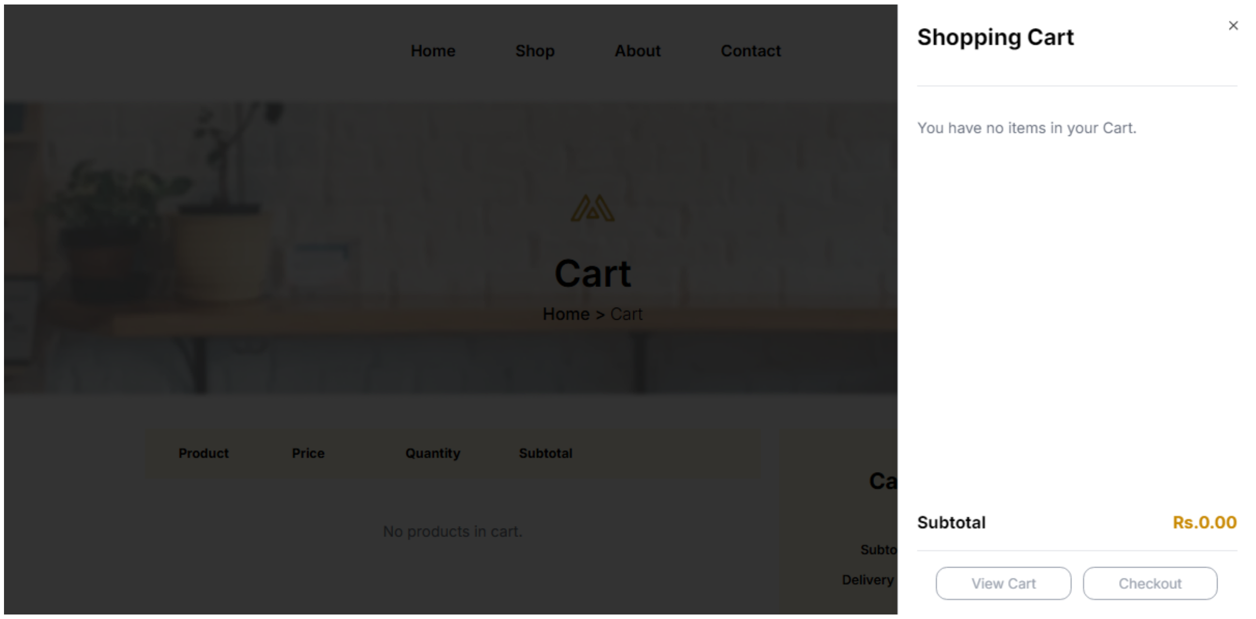
View Cart
Checkout

You can increase the product quantity up to the available stock and decrease it to a minimum of one. If you try to reduce the quantity below one, the product will be removed from the cart. Additionally, adjusting the quantity with the "+" or "−" buttons will dynamically update the total price displayed on the right-hand side.

If the cart page is empty, I implemented user-friendly code that displays a message encouraging users to browse our collection. Below this explanation, I have included code snippets to illustrate this functionality.

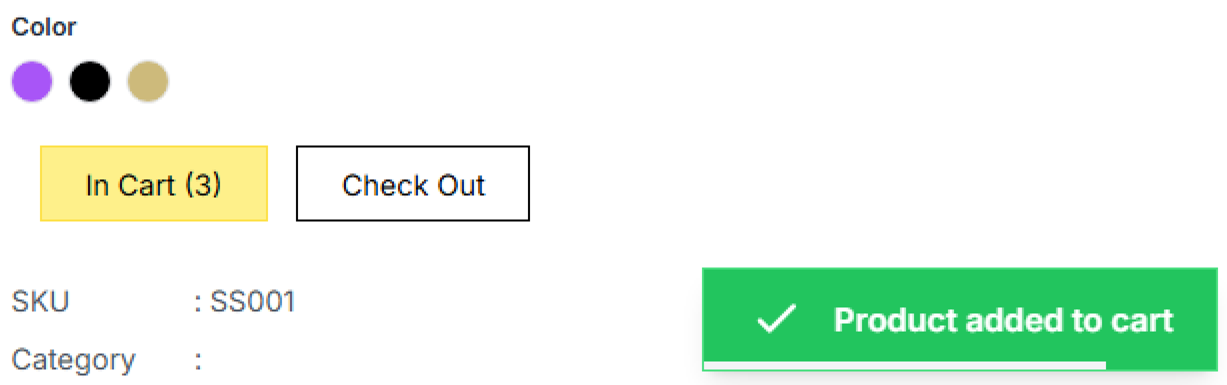
```
{/* Scrollable area for cart items */}
<div className="flex-1 overflow-y-auto max-h-[400px] p
  {cartItems.length > 0 ? (
    cartItems.map((item) => (
      <div className="flex items-center gap-4 pb-4" ke
        <Image
          src={item.image || "/placeholder.svg"}
          alt={item.name}
          width={80}
          height={80}
          className="w-[80px] h-[80px] object-cover"
        />
        <div className="flex flex-col gap-2">
          <h3 className="text-sm font-semibold">{item
          <p>
            {item.quantity} x{" "}
            <span className="text-yellow-600 font-sem
              ${item.price?.toFixed(2) || "0.00"}
            </span>
          </p>
        </div>
        <div className="ml-auto flex items-center gap
          <QuantityControl
            quantity={item.quantity}
            onIncrease={() => updateQuantity(item.id,
            onDecrease={() => updateQuantity(item.id,
          />
          <button
            onClick={() => removeFromCart(item.id)}
            className="ml-4 text-gray-500 hover:text-
          >
            ✖
          </button>
        </div>
      </div>
    ))
  ) : (
    <p className="text-gray-500">You have no items in
  )}
</div>
```

wanna see how it looks like ???
here it  is :)



## 8. Notification Component

I implemented a beautifully designed, well-structured toast notification system that seamlessly triggers after adding an item to the wishlist or cart. The notifications are elegant, user-friendly, and provide instant feedback with clear messages, enhancing the overall user experience.



```
<div
    className={`fixed bottom-5 right-6 transition-all duratior
```

```
        show ? 'opacity-100 translate-y-0' : 'opacity-0 translat
      } bg-green-500 text-white py-3 px-6 rounded- shadow-lg fle
    >
      {/* Checkmark Icon */}
      <svg
        xmlns="http://www.w3.org/2000/svg"
        fill="none"
        viewBox="0 0 24 24"
        stroke="currentColor"
        className="w-8 h-8 text-white"
      >
        <path
          stroke-linecap="round"
          stroke-linejoin="round"
          stroke-width="2"
          d="M5 13l4 4L19 7"
        />
      </svg>

      <span className="text-[15px] md:text-[18px] font-semibold"

      {/* Processing Line (below the notification) */}
      <div className="absolute bottom-0 left-[-16px] w-full h-[5
    </div>
```