



DAY 4 - DYNAMIC FRONTEND COMPONENTS - SHOP.CO



OVERVIEW:

On Day 4, I implemented key features to enhance the functionality of my e-commerce website, Shop.co, ensuring a dynamic and user-friendly experience. Below are the details of the work completed:

FEATURES IMPLEMENTED

Product Listing Page:

Product Detail Page:

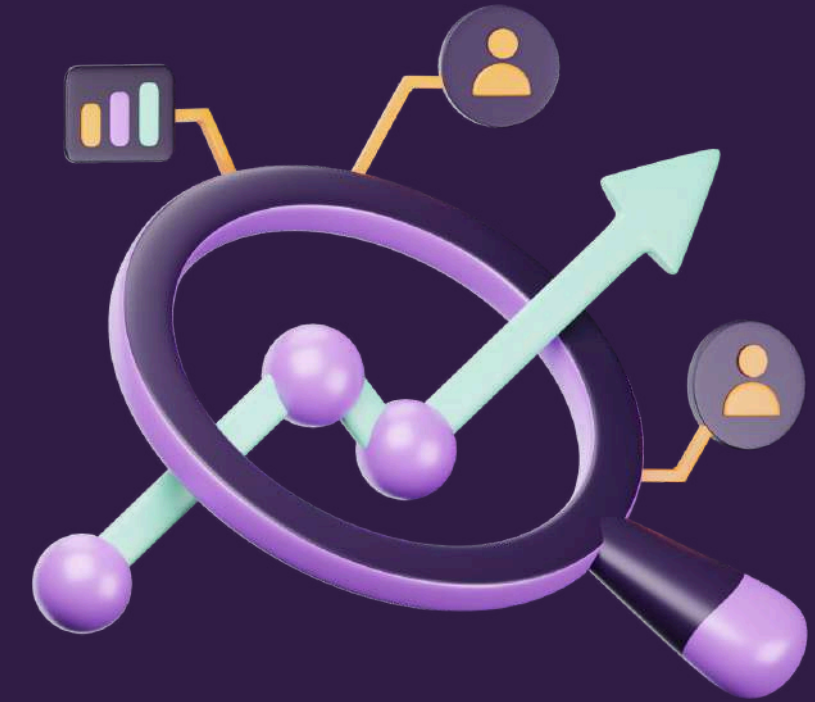
Add to Cart Functionality with Redux:

Pagination with ShadCN UI:

Responsive Design with Tailwind css



TECHNICAL WORKFLOW



Product Listing Page Workflow

- Fetched dynamic data from Sanity CMS to populate the product grid.
- Integrated category filters, a search bar, and pagination to allow users to browse and locate products seamlessly.

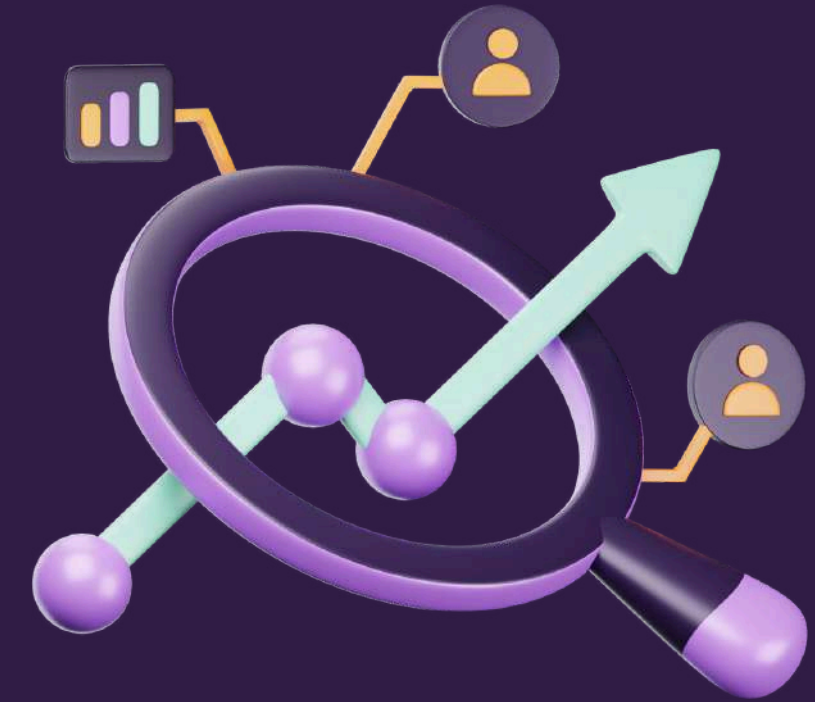
Category Page:

- Dynamically fetched products based on their categories using Sanity CMS.
- Rendered category-specific product listings with pagination for better user navigation.

Product Detail Page:

- Implemented individual product detail pages with accurate dynamic routing using Next.js.
- Each detail page renders data dynamically from Sanity CMS, including

TECHNICAL WORKFLOW



Add to Cart Workflow:

- Developed a slice using Redux Toolkit to manage cart state.
- Configured store.ts to include persistence middleware for saving cart data in local storage.
- Connected the toast notification component to trigger upon successful cart actions.

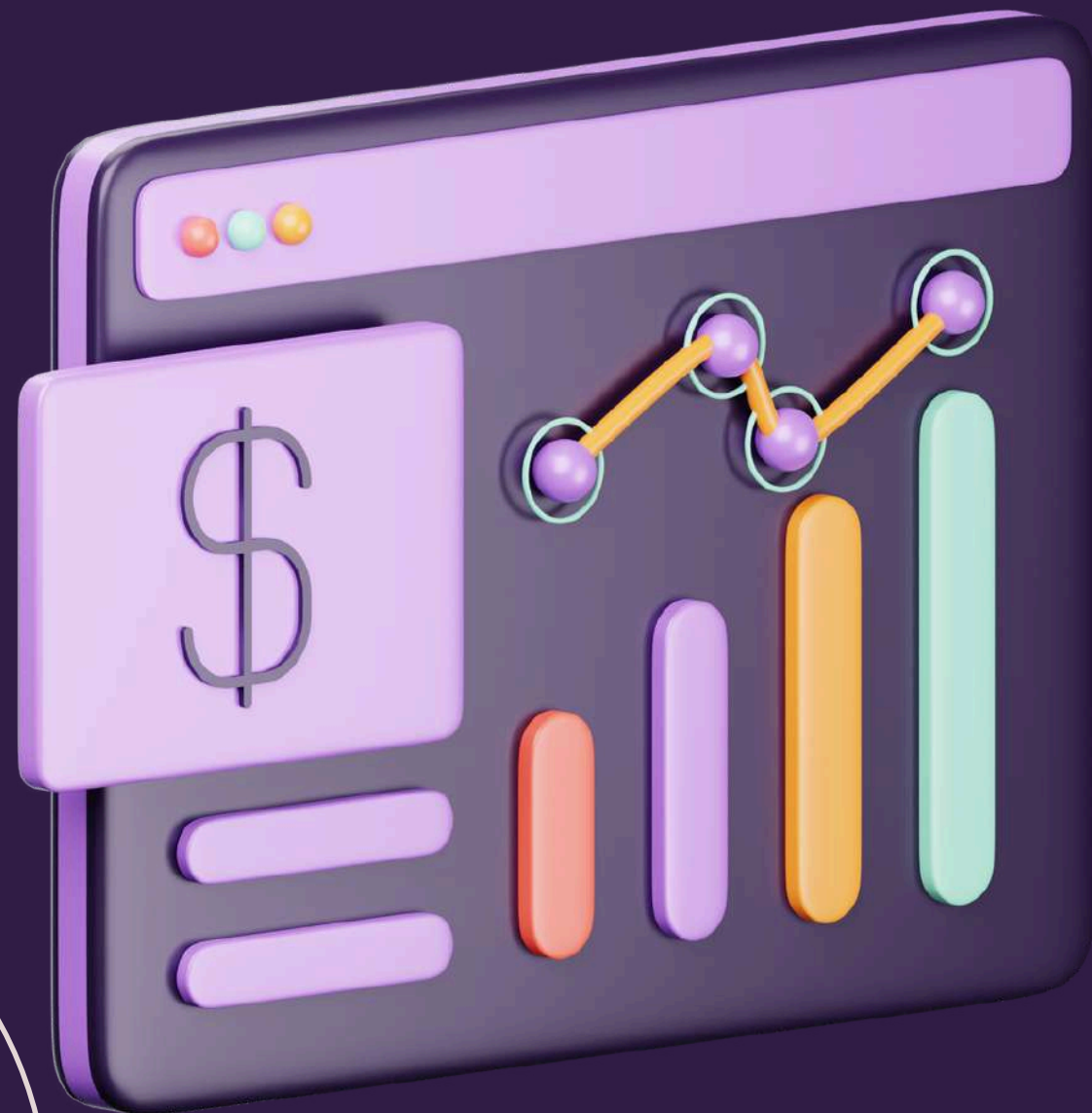
Responsive Design:

- Thoroughly tested and adjusted the layout for different screen sizes, ensuring consistent functionality and aesthetics.

Toastify Notifications:

- Created a client-rendered component to manage toast notifications efficiently.
- Added success messages to confirm user actions, such as adding items to the cart.

BEST PRACTICES FOLLOWED



- Ensured code modularity by separating concerns into dedicated folders and files.
- Followed clean code principles, making the project easy to maintain and extend.
- Implemented efficient state management using Redux Toolkit and redux-persist.
- Tested the application thoroughly to ensure responsiveness and functionality across devices.

Product Listing Page

```
1  "use client"
2  import { Button } from "@components/ui/button";
3  import { client } from "@sanity/lib/client";
4  import { urlFor } from "@sanity/lib/image";
5  import Image from "next/image";
6  import Link from "next/link";
7  import { FaStar } from "react-icons/fa";
8  import { useEffect, useState } from "react";
9
10 // Adding key prop in star array
11 let star = [
12   <FaStar key={1} />,
13   <FaStar key={2} />,
14   <FaStar key={3} />,
15   <FaStar key={4} />,
16   <FaStar key={5} />,
17 ];
18
19 interface Iproducts {
20   image: string;
21   discountPercent: number;
22   isNew: boolean;
23   name: string;
24   description: string;
25   price: number;
26   _id: string;
27 }
28
29 export default function Product() {
30   const [products, setProducts] = useState<Iproducts[]>([]);
31
32   useEffect(() => {
33     const fetchProducts = async () => {
34       try {
35         const fetchedProducts = await client.fetch(
36           `*[_type == 'products' && category == 'tshirt']{
37             "image": image.asset->url,
38             category,
39             discountPercent,
40             isNew,
41             name,
42             description,
43             price,
44             _id
45           }[0...4]`
46         );
47         setProducts(fetchedProducts);
48       } catch (error) {
49         console.error("Error fetching products:", error);
50       }
51     };
52   });
```

```
44         _id
45       }[0...4]`
46     );
47     setProducts(fetchedProducts);
48   } catch (error) {
49     console.error("Error fetching products:", error);
50   }
51 };
52
53   fetchProducts();
54 }, []);
55
56 return (
57   <>
58     <div className="w-full h-full mt-10 lg:mt-36 max-w-screen-xl mx-auto">
59       <h1 className="text-3xl md:text-4xl font-bold text-center">NEW ARRIVALS</h1>
60       <div className="relative mt-10 overflow-x-auto flex space-x-5 px-8">
61         {products.length > 0 ? (
62           products.map((data) => {
63             return (
64               <div key={data._id} className="flex-shrink-0">
65                 <Link href={` /product/${data._id}`}>
66                   <div className="w-[200px] md:w-[283px] h-[200px] md:h-[290px] bg-[#F0EEEE] rounded-[20px]">
67                     {data.image && (
68                       <Image
69                         src={urlFor(data.image).url()}
70                         alt={data.name}
71                         className="w-full h-full rounded-[20px]"
72                         width={100}
73                         height={100}
74                       />
75                     )}
76                   </div>
77                 </Link>
78                 <div className="pl-2">
79                   <p className="text-lg mt-2 font-bold">{data.name}</p>
80                   <div className="flex text-yellow-400">
81                     {star.map((icon, index) => (
82                       <span key={index}>{icon}</span>
83                     ))}
84                   </div>
85                   <p className="font-bold mt-1">
86                     {data.price}{" "}
87                     <span className="text-gray-400 font-bold line-through">
88                       {data.discountPercent}
89                     </span>
90                   </p>
91                 </div>
92               </div>
93             );
94           })
95         ) : (
96           <p className="text-center text-gray-500 w-full">
97             No products available at the moment. Please try again later.
98           </p>
99         )}
100       </div>
101     <div className="flex justify-center items-start mt-5">
102       <Link href="/casual">
103         <Button
104           variant={"outline"}
105           className="sm:mt-0 w-[80%] sm:w-[200px] rounded-[20px]"
106         >
107           View all
108         </Button>
109       </Link>
110     </div>
111   </>
112 );
113 }
114 }
115 }
```



```
1  "use client";
2  import Image from "next/image";
3  import { FaStar } from "react-icons/fa";
4  import { Minus, Plus } from "lucide-react";
5  import { useEffect, useState } from "react";
6  import { client } from "@sanity/lib/client";
7  import { urlFor } from "@sanity/lib/image";
8  import CustomerTestimonials from "@components/AllReviews";
9  import TopSell from "@components/nextVsls";
10 import { BreadcrumbCollapsed } from "@components/Breadcrumb";
11 import { useDispatch } from "react-redux";
12 import Toastify from "react-toastify";
13
14 // Adding key prop in star array
15 let star = [
16   <FaStar key={1} />,
17   <FaStar key={2} />,
18   <FaStar key={3} />,
19   <FaStar key={4} />,
20   <FaStar key={5} />,
21 ];
22
23 interface Iproducts {
24   image: string[];
25   discountPercent: number;
26   isNew: boolean;
27   name: string;
28   description: string;
29   price: number;
30   _id: string;
31   colors: string[];
32   sizes: string[];
33 }
34
35 export default function SlugPage({ params }: { params: { id: string } }) {
36   const [product, setProduct] = useState<Iproducts | null>(null);
37   const [cartItem, setCartItem] = useState<any>(null);
38   const [loading, setLoading] = useState<boolean>(true);
39   const [error, setError] = useState<boolean>(false);
40   const dispatch = useDispatch();
41   useEffect(() => {
42     const fetchProduct = async () => {
43       try {
44         setLoading(true);
45         const products: Iproducts[] = await client.fetch(
46           `[_type == 'products']{
47             "image": image.asset->url,
48             category,
49             discountPercent,
50             isNew,
51             name,
52             description,
53             price,
54             _id,
55             colors,
56             sizes
57           }`
58         );
59       } catch (err) {
60         setError(true);
61       } finally {
62         setLoading(false);
63       }
64     };
65     const slug = products.find((item) => item._id === params.id);
66     if (!slug) {
67       setError(true);
68     } else {
69       setProduct(slug);
70       setCartItem({
71         id: slug._id,
72         name: slug.name,
73         image: slug.image,
74         price: slug.price,
75         size: slug.sizes[0],
76         color: slug.colors[0],
77         qty: 1,
78         discount: slug.discountPercent,
79       });
80     }
81   }, [params.id]);
82
83   fetchProduct();
84 }, [params.id]);
85
86
87
88 if (loading) {
89   return <h1 className="text-center mt-28 font-bold">Loading...</h1>;
90 }
91
92 if (error || !product) {
93   return <h1 className="mt-28 text-center font-bold">Product not found</h1>;
94 }
95
96
97 return (
98   <>
99     <div className="mt-28 md:mt-36">
100       <BreadcrumbCollapsed />
101       <div className="flex h-full items-center flex-col md:flex-row justify-center sm:justify-evenly sm:p-0 max-w-screen-2xl mx-auto">
102         {/* Left */}
103         <div className="flex space-x-4 md:space-x-0 md:space-y-3 p-5 md:flex-col justify-between items-center md:w-[200px] order-2 md:order-1">
104           {product.image &&
105             <Image
106               key={product._id}
107               src={urlFor(product.image).url()}
108               className="w-[100px] h-[100px] md:h-[150px] lg:mt-3 rounded-[20px]"
109               alt={product.name}
110               width={100}
111               height={100}
112             />
113           }
114           {product.image &&
115             <Image
116               key={product._id}
117               src={urlFor(product.image).url()}
118               className="w-[100px] h-[100px] md:h-[150px] lg:mt-3 rounded-[20px]"
119               alt={product.name}
120               width={100}
121               height={100}
122             />
123           }
124           {product.image &&
125             <Image
126               key={product._id}
127               src={urlFor(product.image).url()}
128               className="w-[100px] h-[100px] md:h-[150px] lg:mt-3 rounded-[20px]"
129               alt={product.name}
130               width={100}
131               height={100}
132             />
133           }
134         </div>
135         {/* Mid */}
136         <div className="w-[90%] p-3 h-[200px] lg:w-[500px] md:h-[300px] mt-5 lg:mt-0 order-1 md:order-2">
137           {product.image &&
138             <Image
139               key={product._id}
140               src={urlFor(product.image).url()}
141               className="w-full h-full sm:mt-3 rounded-[20px]"
142               alt={product.name}
143               width={100}
144               height={100}
145             />
146           }
147         </div>
148         {/* Right */}
149         <div className="w-full p-5 lg:w-[500px] lg:h-[500px] order-3">
150           <h1 className="text-2xl lg:text-3xl font-bold">{cartItem.name}</h1>
151           <div className="flex text-yellow-400">{star}</div>
152           <div className="flex items-center space-x-2">
153             <p className="font-bold">{cartItem.price * cartItem.qty}</p>
154             {cartItem.discount > 0 && {
155               <span className="text-gray-400 line-through">
156                 {(cartItem.price - (cartItem.price * cartItem.discount) / 100) *
157                   cartItem.qty}
158               </span>
159             )}
160             {cartItem.discount > 0 && {
161               <span className="bg-red-400 rounded-[10px]">{-(cartItem.discount)%}</span>
162             )}
163           </div>
164           <p className="text-sm">{product.description}</p>
165           {/* Select Color */}
166           <div className="mt-5">
167             <p className="text-gray-500">Select Colors</p>
168             <div className="flex space-x-3 mt-2">
169               {product.colors.map((color, i) => (
170                 <button
171                   key={i}
172                   onClick={() => setCartItem({ ...cartItem, color })}
173                   className="w-[30px] h-[30px] border border-black active:outline rounded-full flex justify-center items-center"
174                   style={{ backgroundColor: color }}
175                 >></button>
176               ))}
177             </div>
178             {/* Choose Size */}
179             <div className="mt-4">
180               <p className="text-gray-500">Choose Size</p>
181               <div className="flex space-x-3 mt-2">
182                 {product.sizes.map((size, i) => (
183                   <button
184                     key={i}
185                     onClick={() => setCartItem({ ...cartItem, size })}
186                     className="w-[80px] h-[40px] flex justify-center items-center active:outline rounded-[62px] bg-[#000000] text-gray-400"
187                   >
188                     {size}
189                   </button>
190                 ))}
191             </div>
192             {/* Quantity & Add to Cart */}
193             <div className="flex justify-start items-center mt-2 space-x-4">
194               <button
195                 onClick={() =>
196                   setCartItem({
197                     ...cartItem,
198                     qty: cartItem.qty <= 1 ? 1 : --cartItem.qty,
199                   })
200               >
201                 <Minus />
202               </button>
203               <span>{cartItem.qty}</span>
204               <button
205                 onClick={() =>
206                   setCartItem({ ...cartItem, qty: ++cartItem.qty })
207               >
208                 <Plus />
209               </button>
210             </div>
211             {/* Add to Cart Button */}
212             <div>
213               <button
214                 onClick={()=>handleAdd(cartItem)} className="bg-black text-white w-[300px]">Add to Cart</button>
215             </div>
216             <div>
217               <div>
218                 <div>
219                   <div>
220                     <div>
221                       <div>
222                         <div>
223                           <div>
224                             <div>
225                               <div>
```

Product Detail Page

Dynamic Routing

```
88   if (loading) {
89     return <h1 className="text-center mt-28 font-bold">Loading...</h1>;
90   }
91
92   if (error || !product) {
93     return <h1 className="mt-28 text-center font-bold">Product not found</h1>;
94   }
95
96
97   return (
98     <>
99       <div className="mt-28 md:mt-36">
100         <BreadcrumbCollapsed />
101         <div className="flex h-full items-center flex-col md:flex-row justify-center sm:justify-evenly sm:p-0 max-w-screen-2xl mx-auto">
102           {/* Left */}
103           <div className="flex space-x-4 md:space-x-0 md:space-y-3 p-5 md:flex-col justify-between items-center md:w-[200px] order-2 md:order-1">
104             {product.image &&
105               <Image
106                 key={product._id}
107                 src={urlFor(product.image).url()}
108                 className="w-[100px] h-[100px] md:h-[150px] lg:mt-3 rounded-[20px]"
109                 alt={product.name}
110                 width={100}
111                 height={100}
112               />
113             }
114             {product.image &&
115               <Image
116                 key={product._id}
117                 src={urlFor(product.image).url()}
118                 className="w-[100px] h-[100px] md:h-[150px] lg:mt-3 rounded-[20px]"
119                 alt={product.name}
120                 width={100}
121                 height={100}
122               />
123             }
124             {product.image &&
125               <Image
126                 key={product._id}
127                 src={urlFor(product.image).url()}
128                 className="w-[100px] h-[100px] md:h-[150px] lg:mt-3 rounded-[20px]"
129                 alt={product.name}
130                 width={100}
131                 height={100}
132               />
133             }
134           </div>
135           {/* Mid */}
136           <div className="w-[90%] p-3 h-[200px] lg:w-[500px] md:h-[300px] mt-5 lg:mt-0 order-1 md:order-2">
137             {product.image &&
138               <Image
139                 key={product._id}
140                 src={urlFor(product.image).url()}
141                 className="w-full h-full sm:mt-3 rounded-[20px]"
142                 alt={product.name}
143                 width={100}
144                 height={100}
145               />
146             }
147           </div>
148           {/* Right */}
149           <div className="w-full p-5 lg:w-[500px] lg:h-[500px] order-3">
150             <h1 className="text-2xl lg:text-3xl font-bold">{cartItem.name}</h1>
151             <div className="flex text-yellow-400">{star}</div>
152             <div className="flex items-center space-x-2">
153               <p className="font-bold">{cartItem.price * cartItem.qty}</p>
154               {cartItem.discount > 0 && {
155                 <span className="text-gray-400 line-through">
156                   {(cartItem.price - (cartItem.price * cartItem.discount) / 100) *
157                     cartItem.qty}
158                 </span>
159               }
160               {cartItem.discount > 0 && {
161                 <span className="bg-red-400 rounded-[10px]">{-(cartItem.discount)%}</span>
162               }
163             </div>
164             <p className="text-sm">{product.description}</p>
165             {/* Select Color */}
166             <div className="mt-5">
167               <p className="text-gray-500">Select Colors</p>
168               <div className="flex space-x-3 mt-2">
169                 {product.colors.map((color, i) => (
170                   <button
171                     key={i}
172                     onClick={() => setCartItem({ ...cartItem, color })}
173                     className="w-[30px] h-[30px] border border-black active:outline rounded-full flex justify-center items-center"
174                     style={{ backgroundColor: color }}
175                   >></button>
176                 ))}
177             </div>
178             {/* Choose Size */}
179             <div className="mt-4">
180               <p className="text-gray-500">Choose Size</p>
181               <div className="flex space-x-3 mt-2">
182                 {product.sizes.map((size, i) => (
183                   <button
184                     key={i}
185                     onClick={() => setCartItem({ ...cartItem, size })}
186                     className="w-[80px] h-[40px] flex justify-center items-center active:outline rounded-[62px] bg-[#000000] text-gray-400"
187                   >
188                     {size}
189                   </button>
190                 ))}
191             </div>
192             {/* Quantity & Add to Cart */}
193             <div className="flex justify-start items-center mt-2 space-x-4">
194               <button
195                 onClick={() =>
196                   setCartItem({
197                     ...cartItem,
198                     qty: cartItem.qty <= 1 ? 1 : --cartItem.qty,
199                   })
200               >
201                 <Minus />
202               </button>
203               <span>{cartItem.qty}</span>
204               <button
205                 onClick={() =>
206                   setCartItem({ ...cartItem, qty: ++cartItem.qty })
207               >
208                 <Plus />
209               </button>
210             </div>
211             {/* Add to Cart Button */}
212             <div>
213               <button
214                 onClick={()=>handleAdd(cartItem)} className="bg-black text-white w-[300px]">Add to Cart</button>
215             </div>
216             <div>
217               <div>
218                 <div>
219                   <div>
220                     <div>
221                       <div>
222                         <div>
223                           <div>
224                             <div>
225                               <div>
```


Redux Store

```
1 import { combineReducers, configureStore } from '@reduxjs/toolkit'
2 import cartSlice from '../features/cart'
3 import storage from 'redux-persist/lib/storage';
4 import persistReducer from 'redux-persist/lib/persistReducer';
5
6 const persistconfig = {
7   key: "root",
8   version: 1,
9   storage,
10 }
11 const reducer = combineReducers({
12   cart: cartSlice
13 })
14 const persistedReducer = persistReducer(persistconfig, reducer)
15
16 export const store = configureStore({
17   reducer: persistedReducer,
18   middleware: (getDefaultMiddleware) =>
19     getDefaultMiddleware({serializableCheck: false}),
20 })
21
22 // Infer the `RootState`, `AppDispatch`, and `AppStore` types from the store itself
23 export type RootState = ReturnType<typeof store.getState>
24 // Inferred type: {posts: PostsState, comments: CommentsState, users: UsersState}
25 export type AppDispatch = typeof store.dispatch
26 export type AppStore = typeof store
```

Hooks



```
1  import { useDispatch, useSelector } from 'react-redux'
2  import type { AppDispatch, RootState } from './store'
3
4  // Use throughout your app instead of plain `useDispatch` and `useSelector`
5  export const useAppDispatch = useDispatch.withTypes<AppDispatch>()
6  export const useAppSelector = useSelector.withTypes<RootState>()
```


Cartslice

```
1 import { createSlice, PayloadAction } from '@reduxjs/toolkit'
2
3 // Define the initial state using that type
4
5 export const cartSlice = createSlice({
6   name: 'products',
7   // `createSlice` will infer the state type from the `initialState` a
8   initialState: [],
9   reducers: {
10     // add to cart functionality
11     add(state:any,action){
12       let uuid = Math.floor(1000+Math.random()*9000)
13       let newobj = {...action.payload,uuid}
14       state.push(newobj)
15     },
16     // delete from cart
17     remove(state:any,{payload}){
18       return state.filter((val:any)=> val.uuid !== payload)
19     },
20     // addition of item
21     addition(state:any,action){
22       let obj = state.find(
23         (val:any)=>
24           val.id == action.payload.id &&
25           val.color == action.payload.color &&
26           val.size == action.payload.size
27       );
28       if(obj){
29         ++obj.qty;
```

```

30       }
31       ++obj.qty;
32       let newState = state.filter((val:any)=> val.id !== obj.id);
33       state = [...newState,obj];
34       return
35     },
36     // subtraction of item
37     subtraction(state:any,action){
38       let obj = state.find(
39         (val:any)=>
40           val.id == action.payload.id &&
41           val.color == action.payload.color &&
42           val.size == action.payload.size
43       );
44       if(obj !== undefined){
45         --obj.qty;
46         let newState = state.filter((val:any)=> val.uuid !== obj.uuid);
47         state = [...newState,obj];
48         return;
49       }
50     }
51   })
52
53 export const {add, remove,subtraction,addition} = cartSlice.actions
54
55 export default cartSlice.reducer
```


Provider



```
1  "use client"
2  import { Provider } from "react-redux";
3  import { store } from "../app/Redux/store";
4  import { PersistGate } from "redux-persist/integration/react";
5  import { persistStore } from "redux-persist";
6
7
8  function Providers({ children, }: Readonly<{children: React.ReactNode;}>) {
9    let persistore = persistStore(store)
10   return (
11     <Provider store={store}>
12       <PersistGate persistor={persistore}>
13         {children}
14       </PersistGate>
15     </Provider>
16   )
17 }
18
19 export default Providers
```



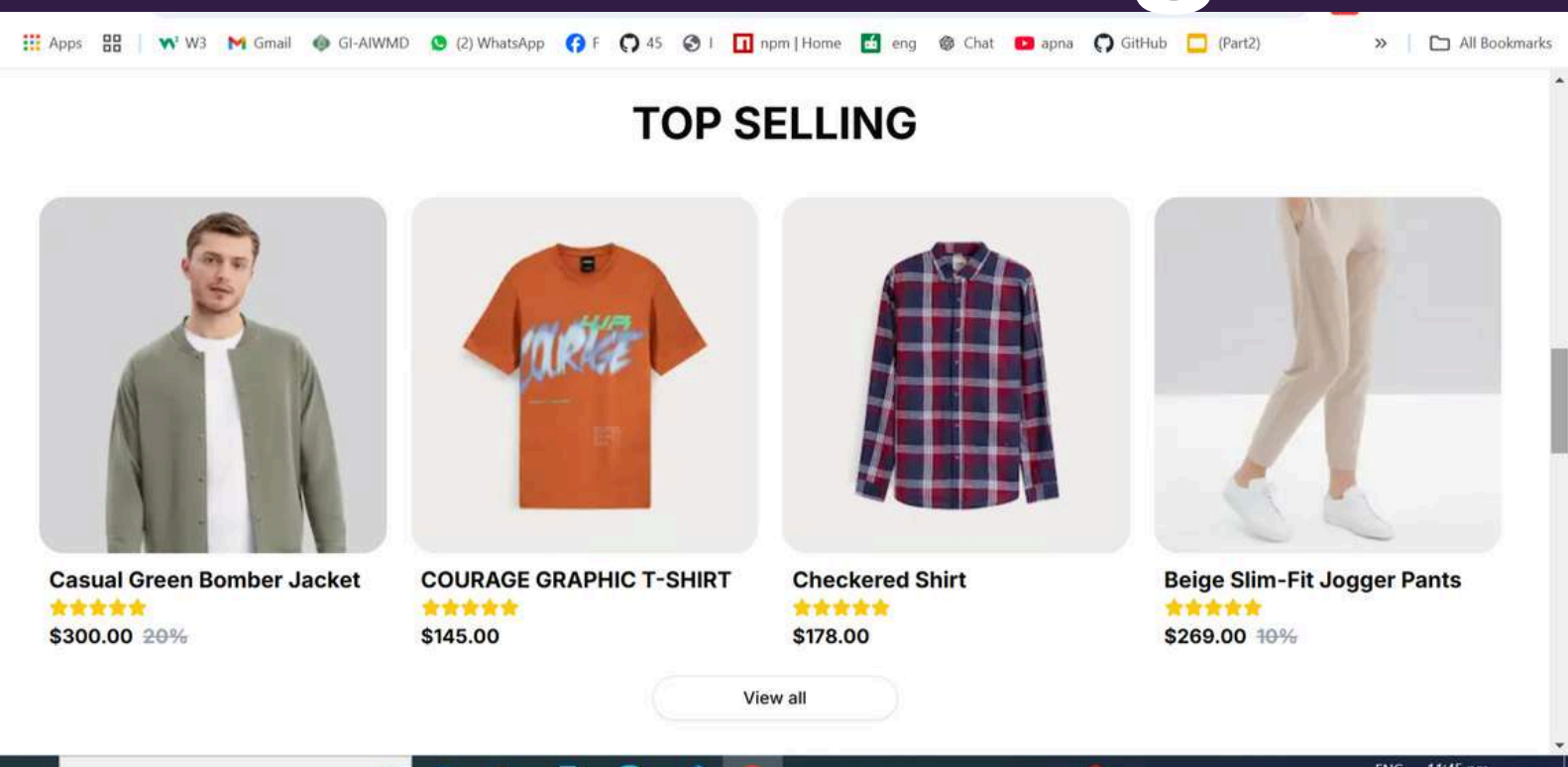
```
1  import {
2    Pagination,
3    PaginationContent,
4    PaginationEllipsis,
5    PaginationItem,
6    PaginationLink,
7    PaginationNext,
8    PaginationPrevious,
9  } from "@components/ui/pagination"
10 import React from 'react'
11
12 function Paginationpage() {
13   return (
14     <div className="w-full mt-4">
15       <Pagination>
16         <PaginationContent className="w-full space-x-4 flex justify-center lg:ml-7">
17           <PaginationItem>
18             <PaginationPrevious href="/" />
19           </PaginationItem>
20
21           <PaginationItem className="hidden lg:block">
22             <PaginationLink href="/">Home</PaginationLink>
23           </PaginationItem>
24           <PaginationItem className="hidden lg:block">
25             <PaginationEllipsis />
26           </PaginationItem>
27
28           <PaginationItem className="hidden lg:block">
29             <PaginationLink href="/sell">Top Sell</PaginationLink>
30           </PaginationItem>
31           <PaginationItem className="hidden lg:block">
32             <PaginationEllipsis />
33           </PaginationItem>
34
35           <PaginationItem>
36             <PaginationLink href="/brand">Brands</PaginationLink>
37           </PaginationItem>
38           <PaginationItem>
39             <PaginationEllipsis />
40           </PaginationItem>
41
42           <PaginationItem>
43             <PaginationNext href="/cart" />
44           </PaginationItem>
45         </PaginationContent>
46       </Pagination>
47     </div>
48   )
49 }
50
51
52 export default Paginationpage
```

Pagination & Toastify

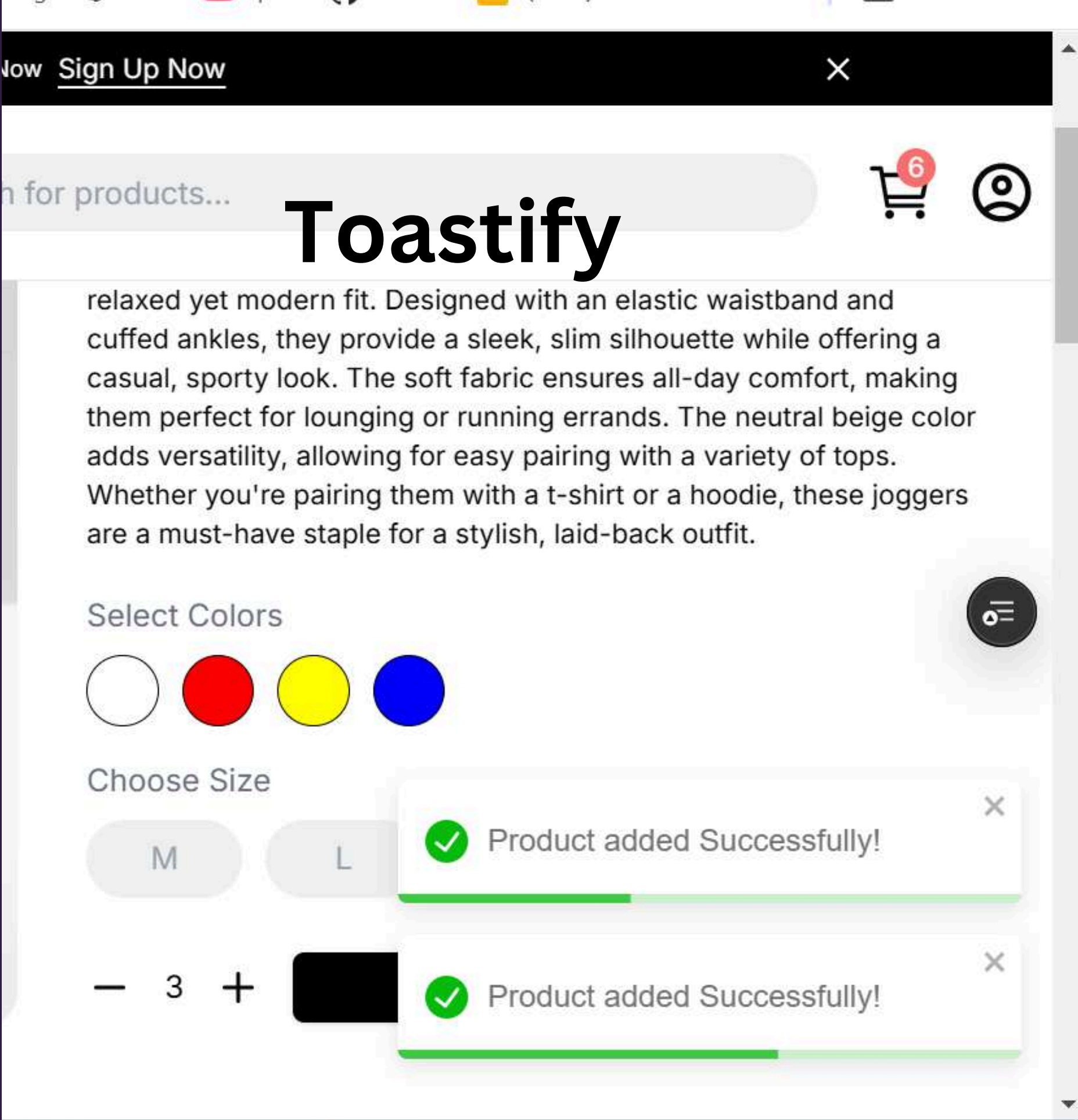
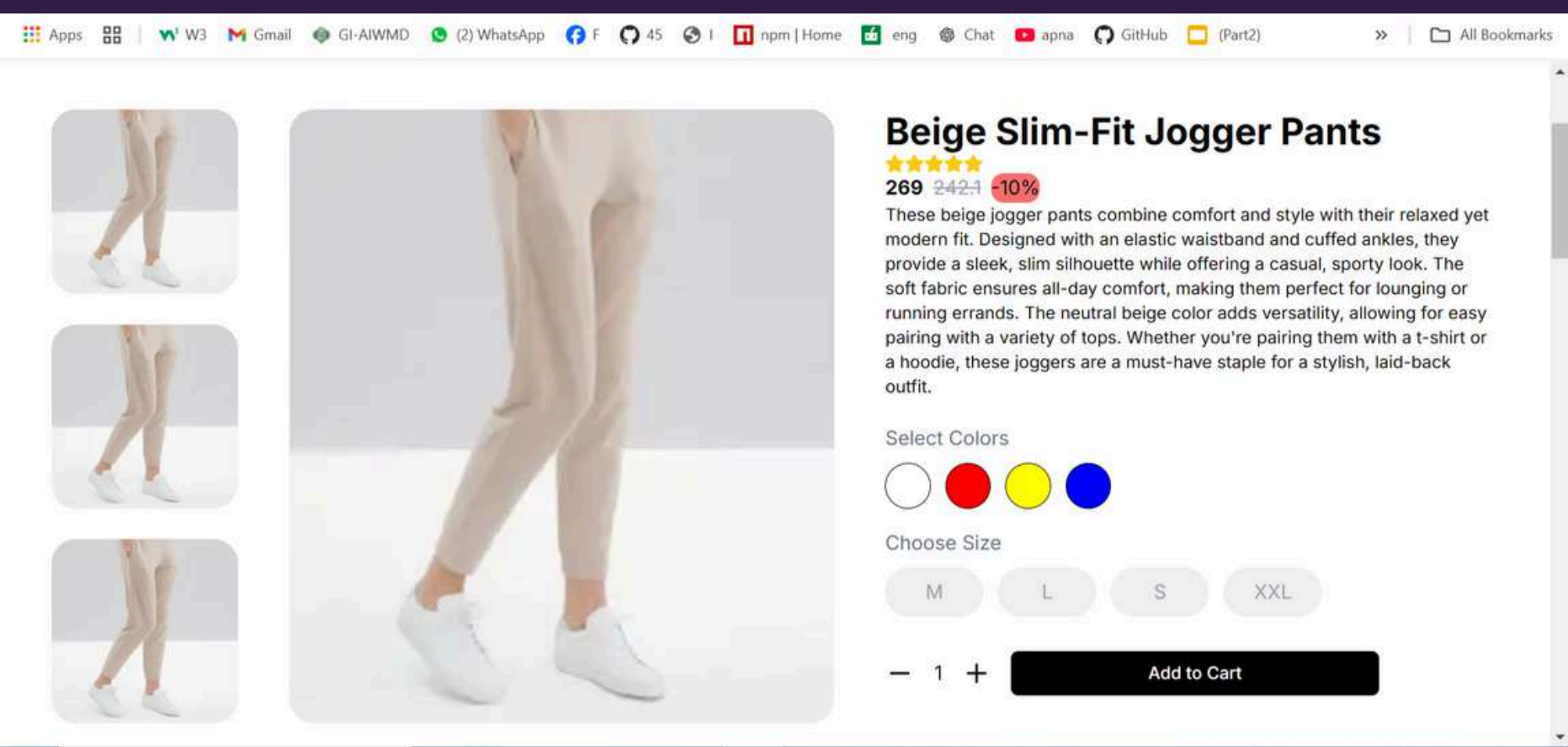


```
1  "use client"
2  import React from 'react';
3  import { useDispatch } from 'react-redux';
4  import { Bounce, ToastContainer, toast } from 'react-toastify';
5  import "react-toastify/ReactToastify.css";
6  import { add } from '../Redux/features/cart';
7  import { Button } from '@components/ui/button';
8
9
10 function Toastify({cartItem}:any) {
11
12   const dispatch = useDispatch()
13
14   const handleadd = (cartItem:any)={
15     dispatch(add(cartItem))
16   }
17
18
19   const notify = () =>
20   toast.success('Product added Successfully!', {
21     position: "bottom-right",
22     autoClose: 5000,
23     hideProgressBar: false,
24     closeOnClick: false,
25     pauseOnHover: true,
26     draggable: true,
27     progress: undefined,
28     theme: "light",
29     transition: Bounce,
30   });
31   return (
32     <>
33       <div onClick={()=>handleadd(cartItem)}>
34         <Button onClick={notify} className="bg-black text-white w-[300px]"
35           >Add to Cart</Button>
36       </div>
37       <ToastContainer
38         position="bottom-right"
39         autoClose={5000}
40         hideProgressBar={false}
41         newestOnTop={false}
42         closeOnClick={false}
43         rtl={false}
44         pauseOnFocusLoss
45         draggable
46         pauseOnHover
47         theme="light"
48         transition={Bounce}
49       />
50     </>
51   )
52 }
53
54
55 export default Toastify
```

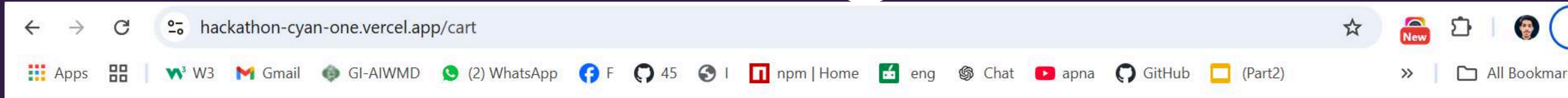

Product Listing



Detail Page



Cart Page



Sign up and get 20% off to your first order. [Sign Up Now](#) [Sign Up Now](#)

SHOP.CO

Shop ▾ On Sale New Arrivals Brands

Search for products...



Home > Shop > Mens T-shirts >



Casual Green Bomber Jacket

Size:XL

Color:Yellow

\$960

— 4 +

Order Summary

Subtotal	\$960
Discount (-20%)	-\$0
Delivery Fee	\$0

Total **\$960**

Add promo code

Apply

[Go to Checkout](#)

THANK YOU

