DAY-4

DOCUMENTATION FOR:-

BUILDING DYNAMIC FRONTEND COMPONENTS FOR MARKETPLACE

“THE FURNITURE PARLOR”

In this documentation, I worked out functionality of ‘Add to Cart’ in my Furniture Parlor Website for Marketplace and day 3 I already completed dynamic routing.

ADD TO CART:-

First of all make a folder in app folder (actions) and create a file (actions.ts).

actions/actions.ts

Now write a code for add to cart.

```
import { Product } from "../../types/products";


export const addToCart = (product:Product) => {

    const cart : Product[] =JSON.parse(localStorage.getItem('cart') || '[]')

    const existingProductIndex =cart.findIndex(item=>item._id === product._id)

    if(existingProductIndex>-1){

        cart[existingProductIndex].stockLevel +=1        //the name comes form
sanity/schema/product.ts (stockLevel)

    }
    else{
        cart.push({...product, stockLevel:1})
    }
    localStorage.setItem('cart',JSON.stringify(cart))
}
export const removeFromCart = (productId:string) => {
    let cart : Product[] = JSON.parse(localStorage.getItem('cart') || '[]')
    cart = cart.filter(item=>item._id !== productId)
    localStorage.setItem('cart',JSON.stringify(cart))
}
export const updateCartQuantity =(productId:string, quantity:number)=>{
    const cart :Product[] = JSON.parse(localStorage.getItem('cart') || '[]')
    const productIndex = cart.findIndex(item=>item._id === productId)
    if (productIndex > -1){
        cart [productIndex].stockLevel=quantity;
        localStorage.setItem ('cart',JSON.stringify(cart))
    }
}
 else{
        cart.push({...product, stockLevel:1})
    }
```

Find the name that written as stockLevel or inventory sanity/schema/product.ts and now go to types/product.ts paste name and define as number i.e.

```
     current : string;
    };
 stockLevel: number;
}
```

And to get item the following code be added in actions.ts file.

```
export const getCartItems =():Product[]=>{
    return JSON.parse(localStorage.getItem('cart') || '[]')
    }
```

The code for Add to Cart has been completed, now we use this code into that component where we want to use add to cart option/button i.e. featuredProduct/page.tsx.

Open the file featuredProduct/page.tsx and the following code be added after use effect and before return.

```
const handleAddToCart = (e:React.MouseEvent, product: Product) =>{

    e.preventDefault()

    Swal.fire({

        position: "top",

        icon: "success",

        title: `${product.name} added to cart`,

        showConfirmButton: false,

        timer:1000

    })

    addToCart(product)

}
```

Now make a button of Add to Cart and this code can be write after the p tag of price.

```
<button className="bg-gradient-to-r from-blue-500 to-purple-500 text-white font-bold py-2 px-4 rounded-lg shadow-md " onClick={(e)=> handleAddToCart(e, product)}>

Add To Cart

</button>
```

And now install sweet alert, type the following command in cmd.

npm install sweetalert2

After the installation the following code before addToCart

```
  Swal.fire({

        position: "top",

        icon: "success",

        title: `${product.name} added to cart`,

        showConfirmButton: false,

        timer:1000


    })
```

The complete code of featuredProduct/page.tsx is as under:-

```
"use client"


import Image from "next/image";

import React, {useEffect, useState} from "react"

import { Product } from "../../../types/products";

import { client } from "@/sanity/lib/client";

import { four } from "@/sanity/lib/queries";

import { urlFor } from "@/sanity/lib/image";

import Link from "next/link";

import { addToCart } from "@/app/actions/actions";

import Swal from "sweetalert2";
```

```
const FeaturedProduct = () => {
   const [product, setProduct] = useState<Product[]>([])


    useEffect(() => {
        async function fetchproduct(){
            const fetchedProduct : Product[]= await client.fetch(four)
            setProduct(fetchedProduct)
        }
        fetchproduct()
    },[])
      const handleAddToCart = (e:React.MouseEvent, product: Product) =>{
    e.preventDefault()
    Swal.fire({
        position: "top",
        icon: "success",
        title: `${product.name} added to cart`,
        showConfirmButton: false,
        timer:1000   //1000ms = 1s 2000ms = 2s
    })
    addToCart(product)
}
    return (
        <div className="max-w-6xl mx-auto px-4 py-8">
            <h1 className="text-2xl font-bold mb-6 text-center">Featured Products</h1>
            <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-
cols-4 gap-6">
            {product.map((product) =>(
                <div key={product._id}
                className="border rounded-lg shadow-md p-4 hover:shadow-lg transition
duration-200 ">
                    <Link href={`/product/${product.slug.current}`} >


                    {product.image && (
                        <Image
                        src={urlFor(product.image).url()}
                        alt="image"
                        width={200}
                        height={200}
                        className="w-full h-48 object-cover rounded-md"/>
                    )}
                    <h2 className="text-lg font-semibold mt-4">{product.name}</h2>
                    <p className="text-gray-500 mt-2">
                        {product.price}
                       </p>
                       <button className="bg-gradient-to-r from-blue-500 to-purple-500 text-white font-bold py-2 px-4 rounded-lg shadow-md " onClick={(e)=> handleAddToCart(e, product)}>
                    Add To Cart
                    </button>
                    </Link>
                </div>
            )
            )}
        </div>
        </div>
    )
}
export default FeaturedProduct
```

Now to save products in local storage we have to import these products for this we have to create a page cart/page.tsx in app directory

app/cart/page.tsx

In this file add the following code:-

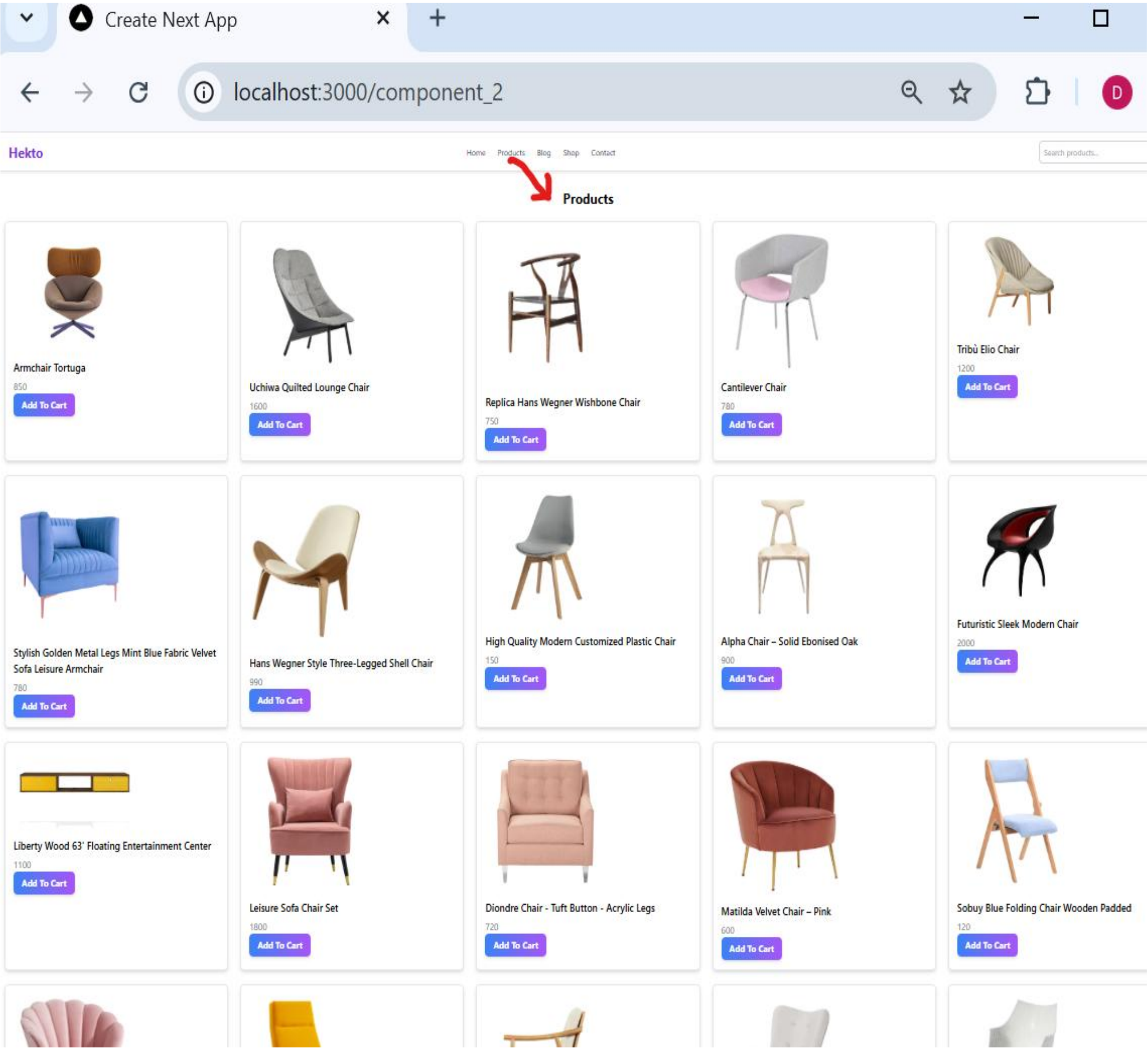
```
"use client";


import { Product } from '@/types/products';

import React, { useEffect, useState } from 'react';

import { getCartItems, removeFromCart, updateCartQuantity } from '../actions/actions';

import Swal from 'sweetalert2';

import { urlFor } from '@/sanity/lib/image';

import Image from 'next/image';


const CartPage = () => {

   const [cartItems, setCartItems] = useState<Product[]>([]);


    useEffect(() => {

        setCartItems(getCartItems());

   }, []);


    const handleRemove = (id: string) => {

        Swal.fire({

            title: 'Are you sure?',

            text: 'You will not be able to recover this item!',

            position: 'top',

            icon: 'warning',

            showCancelButton: true,

            confirmButtonText: 'Yes, remove it!',

        }).then((result) => {

            if (result.isConfirmed) {

                removeFromCart(id);

                setCartItems(getCartItems());

                Swal.fire('Removed!', 'This product has been removed', 'success');

            }

        });
   };

    const handleQuantity = (id: string, quantity: number) => {
        updateCartQuantity(id, quantity);
        setCartItems(getCartItems());
   };

    const handleIncrement = (id: string) => {
        const product = cartItems.find((item) => item._id === id);
        if (product) handleQuantity(id, product.stockLevel + 1);
   };

    const handleDecrement = (id: string) => {
        const product = cartItems.find((item) => item._id === id);
        if (product && product.stockLevel > 1) handleQuantity(id, product.stockLevel -
1);
   };

    const calculateTotal = () => {
        return cartItems.reduce((total, item) => total + item.price * item.stockLevel,
0);
   };

    const handleProceed = () => {
        Swal.fire({
            title: 'Proceed to checkout',
            text: 'Are you sure you want to proceed to checkout?',
            position: 'top',
            icon: 'info',
            showCancelButton: true,
            confirmButtonText: 'Yes',
        }).then((result) => {
            if (result.isConfirmed) {
                Swal.fire('Success', 'Your order has been successfully added',
'success');
                setCartItems([]);
            }
        });
   };

    return (
        <div className="container mx-auto p-4">
            <h2 className="text-2xl font-bold mb-4">Shopping Cart</h2>
            <div className="bg-white p-4 rounded-lg shadow-md">
                {cartItems.length === 0 ? (
                    <p className="text-center text-gray-500">Your cart is empty.</p>
                ) : (
                    <>
                        {cartItems.map((item) => (
     <div key={item._id} className="flex justify-between items-center border-b py-4">
                                <div className="flex items-center gap-4">
                                    {item.image &&(
                                      <Image
                                      src={urlFor(item.image).url()}
                                      alt='image'
                                      width={500}
                                      height={500}
                                   className="w-16 h-16 rounded-md" />
                                    )}
                                    <div>
                       <h3 className="font-medium">{item.name}</h3>
                        <p className="text-sm text-gray-500">${item.price}</p>
                         </div>
                      </div>
                      <div className="flex items-center gap-4 ">
                      <button onClick={() => handleDecrement(item._id)} className="px-3 py-1 bg-gray-300 rounded">-</button>

                       <span>{item.stockLevel}</span>

                       <button onClick={() => handleIncrement(item._id)} className="px-3 py-1 bg-gray-300 rounded">+</button>

                                </div>

                       <button onClick={() => handleRemove(item._id)} className="px-3 py-1 bg-red-500 text-white rounded">Remove</button>

                       </div>

                        ))}

                       <div className="flex justify-between items-center mt-4">

                  <h3 className="text-lg font-bold">Total: ${calculateTotal()}</h3>

                  <button onClick={handleProceed} className="px-4 py-2 bg-blue-500 text-white rounded">Proceed to Checkout</button>

                        </div>

                    </>

                )}

            </div>

        </div>

    );

};

export default CartPage;
```

Now Add to Cart functionality has been competed and after these steps we can perform following functionality in our marketplace:-

- **Can add any product in add to cart.**
- **We can remove any product/item form add to cart.**
- **We can increase quantity of products.**

- **We can decrease quantity of products.**
- **Total amount of products also showing.**
- **Proceed option is also working.**

Product Listing Page:-



Products

Armchair Tortuga
850

Uchiwa Quilted Lounge Chair
1600

Replica Hans Wegner Wishbone Chair
750

Cantilever Chair
780

Tribù Elio Chair
1200

Stylish Golden Metal Legs Mint Blue Fabric Velvet Sofa Leisure Armchair
780

Hans Wegner Style Three-Legged Shell Chair
990

High Quality Modern Customized Plastic Chair
150

Alpha Chair – Solid Ebonised Oak
900

Futuristic Sleek Modern Chair
2000

Liberty Wood 63' Floating Entertainment Center
1100

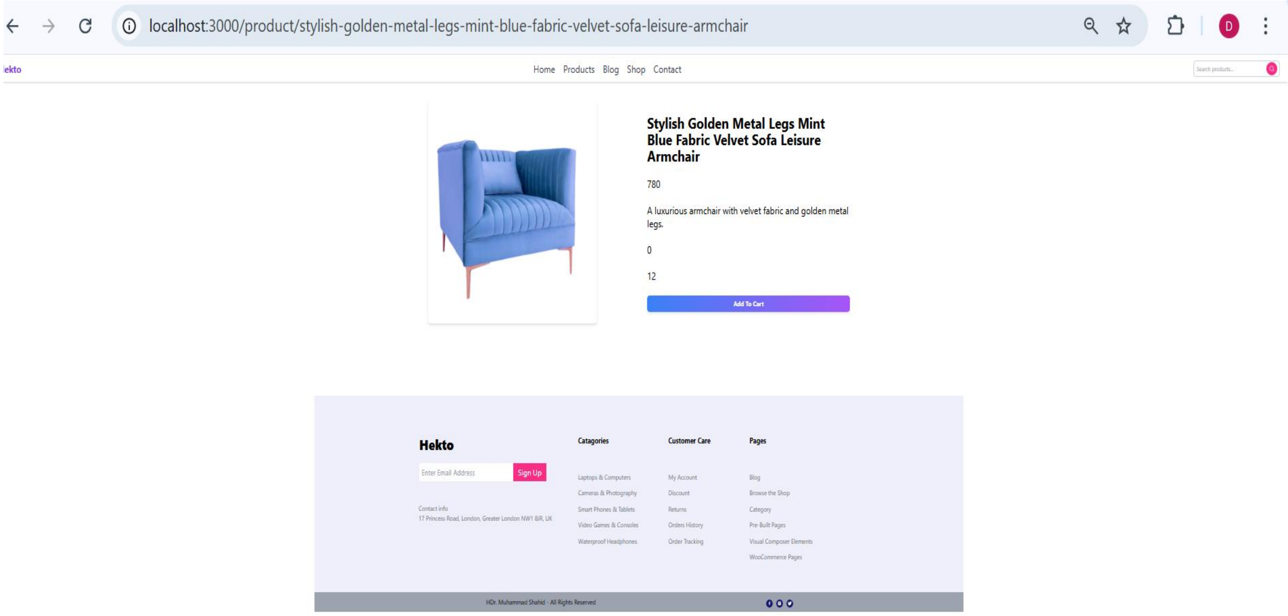
Leisure Sofa Chair Set
1800

Diondre Chair - Tuft Button - Acrylic Legs
720

Matilda Velvet Chair – Pink
600

Sobuy Blue Folding Chair Wooden Padded
120

Individual product detail page with routing:-



localhost:3000/product/stylish-golden-metal-legs-mint-blue-fabric-velvet-sofa-leisure-armchair

Stylish Golden Metal Legs Mint Blue Fabric Velvet Sofa Leisure Armchair

780

A luxurious armchair with velvet fabric and golden metal legs.

0

12

Hekto

Contact info
17 Princess Road, London, Greater London NW1 8JR, UK

HDr. Muhammad Shahid - All Rights Reserved

Day-4, Building Dynamic Frontend Components For Marketplace is completed.

Thanks.