Day 4

Building Dynamic Frontend Components for "GENERAL E-COMMERCE"

A guide to developing reusable and interactive UI components for efficiently displaying and managing e-commerce product data.

1. Product Listing

- Fetch and display product data dynamically using APIs.
- Show product images, names, prices, and tags in a grid or list format.
- Implement lazy loading or infinite scrolling for better performance.

2. Product Details

- Display detailed information about selected products.
- Include product images, descriptions, prices, available stock, and customer reviews.
- Implement image zoom, carousel, or video previews for better visualization.

3. Cart & Checkout

- Add and remove products from the shopping cart dynamically.
- Update product quantities and calculate total costs in real time.
- Implement a smooth and secure checkout process with multiple payment options.

4. Filters & Sorting

- Enable users to filter products based on categories, price range, brand, and ratings.
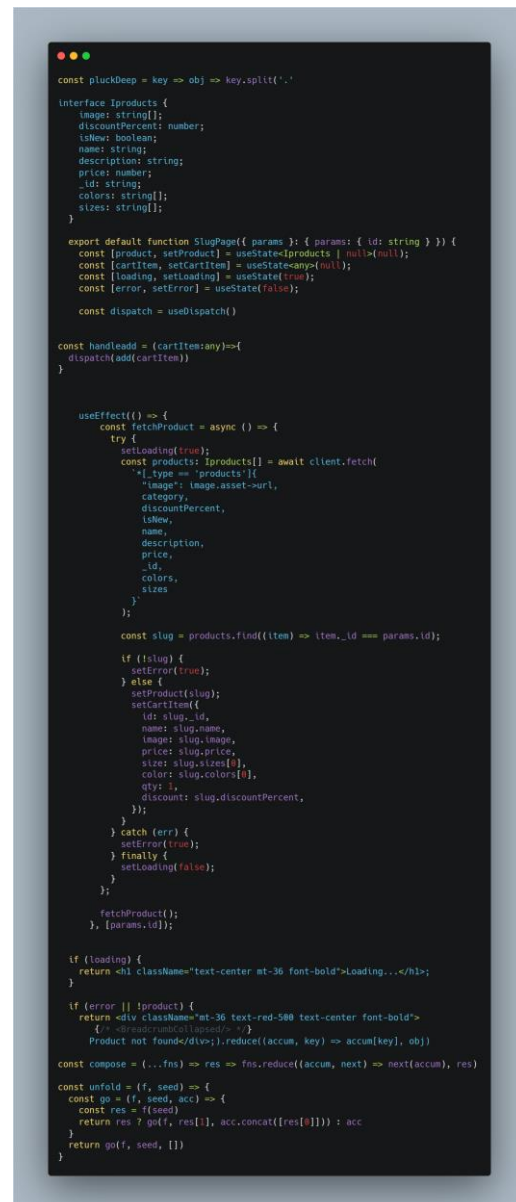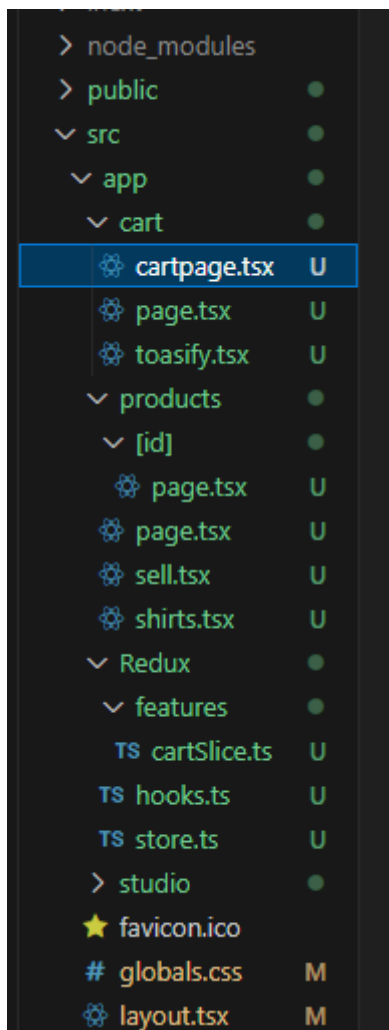
5. Search Functionality

- Implement a real-time search bar for quick product discovery.

6. Responsive & Mobile-Friendly Design

Optimize images and layouts to improve page speed and loading time.

7. User Authentication & Wishlist

- Implement login/signup features using authentication providers (Google, Facebook, etc.).


```
const pluckDeep = key => obj => key.split('.')

interface Iproducts {
    image: string[];
    discountPercent: number;
    isNew: boolean;
    name: string;
    description: string;
    price: number;
    _id: string;
    colors: string[];
    sizes: string[];
}

export default function SlugPage({ params }: { params: { id: string } }) {
    const [product, setProduct] = useState<Iproducts | null>(null);
    const [cartItem, setCartItem] = useState<any>(null);
    const [loading, setLoading] = useState(true);
    const [error, setError] = useState(false);

    const dispatch = useDispatch()

const handleadd = (cartItem:any)=>{
    dispatch(add(cartItem))
}

    useEffect(() => {
        const fetchProduct = async () => {
            try {
                setLoading(true);
                const products: Iproducts[] = await client.fetch(
                    `*[_type == 'products']{
                        "image": image.asset->url,
                        category,
                        discountPercent,
                        isNew,
                        name,
                        description,
                        price,
                        _id,
                        colors,
                        sizes
                    }`
                );

                const slug = products.find((item) => item._id === params.id);

                if (!slug) {
                    setError(true);
                } else {
                    setProduct(slug);
                    setCartItem({
                        id: slug._id,
                        name: slug.name,
                        image: slug.image,
                        price: slug.price,
                        size: slug.sizes[0],
                        color: slug.colors[0],
                        qty: 1,
                        discount: slug.discountPercent,
                    });
                }
            } catch (err) {
                setError(true);
            } finally {
                setLoading(false);
            }
        };

        fetchProduct();
    }, [params.id]);

    if (loading) {
        return <h1 className="text-center mt-36 font-bold">Loading...</h1>;
    }

    if (error || !product) {
        return <div className="mt-36 text-red-500 text-center font-bold">
            {/* <BreadcrumbCollapsed/> */}
            Product not found</div>;).reduce((accum, key) => accum[key], obj)

const compose = (...fns) => res => fns.reduce((accum, next) => next(accum), res)

const unfold = (f, seed) => {
    const go = (f, seed, acc) => {
        const res = f(seed)
        return res ? go(f, res[1], acc.concat([res[0]])) : acc
    }
    return go(f, seed, [])
}
```

```
"use client"
import { createSlice, PayloadAction } from '@reduxjs/toolkit'

// Define the initial state using that type
export const cartSlice = createSlice({
  name: 'products',
  initialState: [],
  reducers: {
    // add to cart functionality
    add(state: any, action) {
      let uuid = Math.floor(1000 + Math.random() * 9000)
      let newobj = { ...action.payload, uuid }
      state.push(newobj)
    },
    // delete from cart
    remove(state: any, { payload }) {
      return state.filter((val: any) => val.uuid !== payload)
    },
    // addition of item
    addition(state: any, action) {
      let obj = state.find(
        (val: any) =>
          val.id == action.payload.id &&
          val.color == action.payload.color &&
          val.size == action.payload.size
      )
      if (obj) {
        ++obj.qty
        let newState = state.filter((val: any) => val.id !== obj.id)
```

COURAGE GRAPHIC T-SHIRT

★★★★★
145

This product is the Courage Graphic T-Shirt, a bold and expressive tee featuring a standout graphic design. Made with soft, breathable fabric, this t-shirt combines comfort with statement style. The eye-catching "Courage" graphic on the front makes it a perfect addition to any casual wardrobe. Ideal for those who want to showcase their bold personality, this t-shirt pairs easily with jeans, shorts, or casual pants for an effortlessly cool look.

Select Colors

⚪ 🔵 ⚫ 🔴

Choose Size

| S | XL | XXL | M |
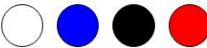
— 1 + | **Add to Cart**

Black Striped T-Shirt 🗑
Size: L
Color: Blue
$240

— 2 +

Order Summary

Subtotal	$240
Discount (-20%)	-$0
Delivery Fee	$0
Total	**$240**

Add promo code | Apply

Go to Checkout