

Day: 04

Dynamic Frontend Component:

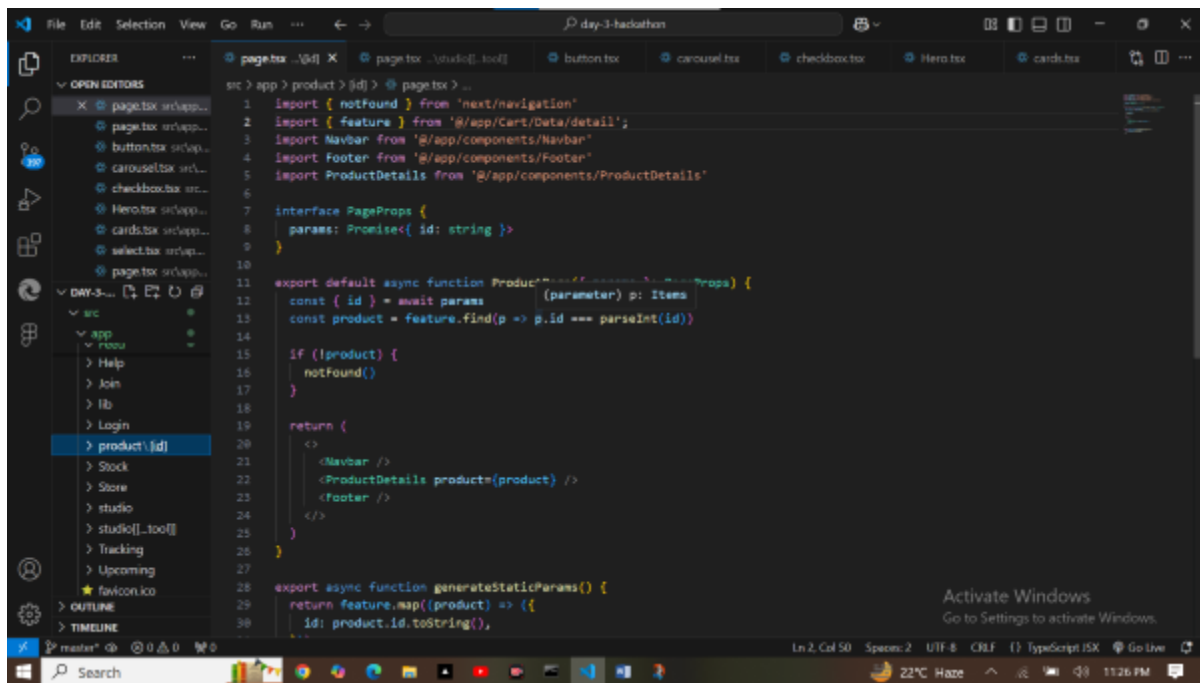
Goal:

"Goal for day 04 is to display data from sanity CMS or APIs with individual product listing and detail pages using dynamic routing along with Modular, reusable components and specific filters."

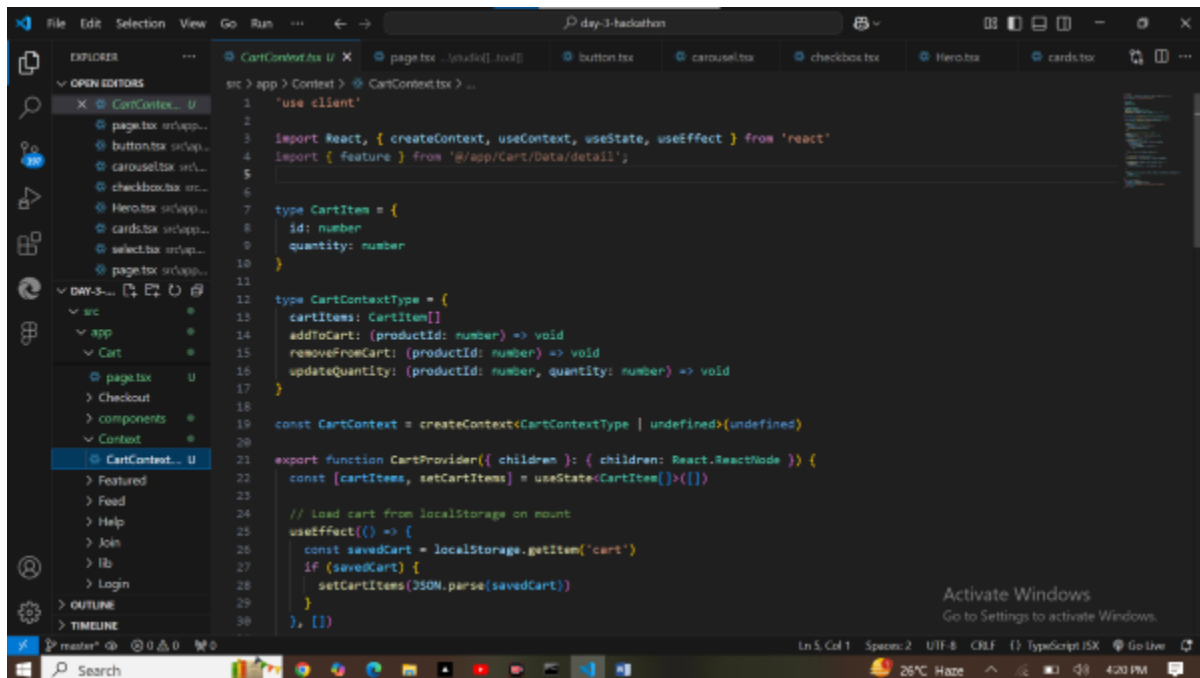
Key Points:

- *Dynamic frontend components are interactive, reusable UI element that adapt to user input, data changes, or application state.*
- *I used to enhance user experience by providing real-time updates, and seamless interactions without requiring full page reloads.*
- *Add dynamic product filters, shopping carts and recommendation systems.*
- *I add real time data visualizations, drag and drop widgets, and customizable layouts.*
- *I made fully responsive website that can easily be used in mobile phones and other devices.*

Here some code snippets of dynamic routing and filters:

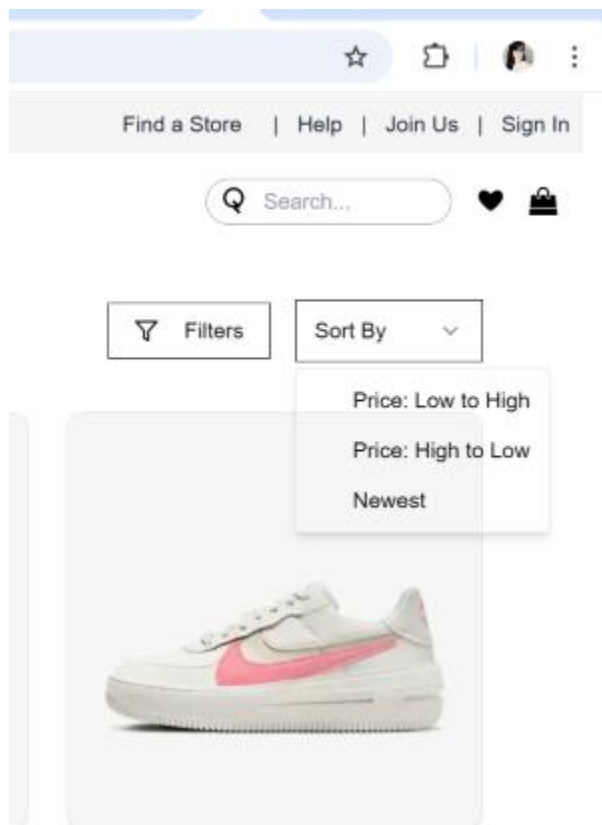


```
src > app > product > [id] > page.jsx > ...
1 import { NotFound } from 'next/navigation'
2 import { feature } from '@app/Cart/Data/detail'
3 import Navbar from '@app/components/Navbar'
4 import Footer from '@app/components/Footer'
5 import ProductDetails from '@app/components/ProductDetails'
6
7 interface PageProps {
8   params: Promise<{ id: string }>
9 }
10
11 export default async function ProductPage({ params }) {
12   const { id } = await params
13   const product = feature.find(p => p.id === parseInt(id))
14
15   if (!product) {
16     NotFound()
17   }
18
19   return (
20     <>
21     <Navbar />
22     <ProductDetails products={product} />
23     <Footer />
24     </>
25   )
26 }
27
28 export async function generateStaticParams() {
29   return feature.map((product) => ({
30     id: product.id.toString(),
31   }))
32 }
```

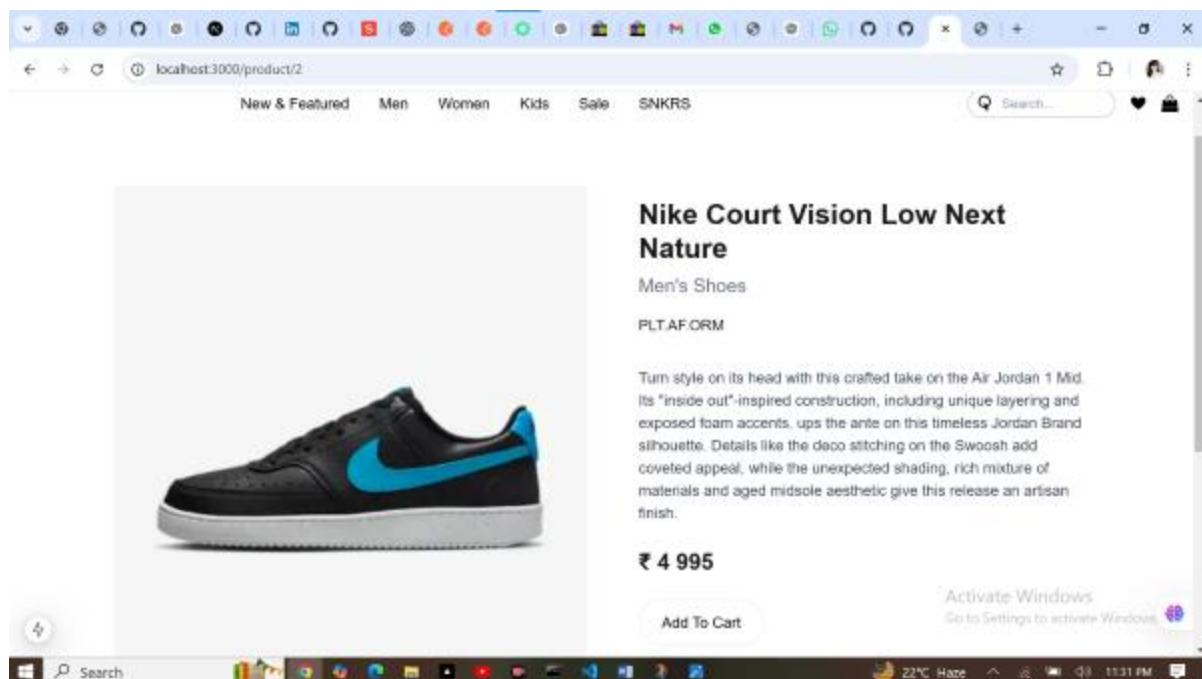


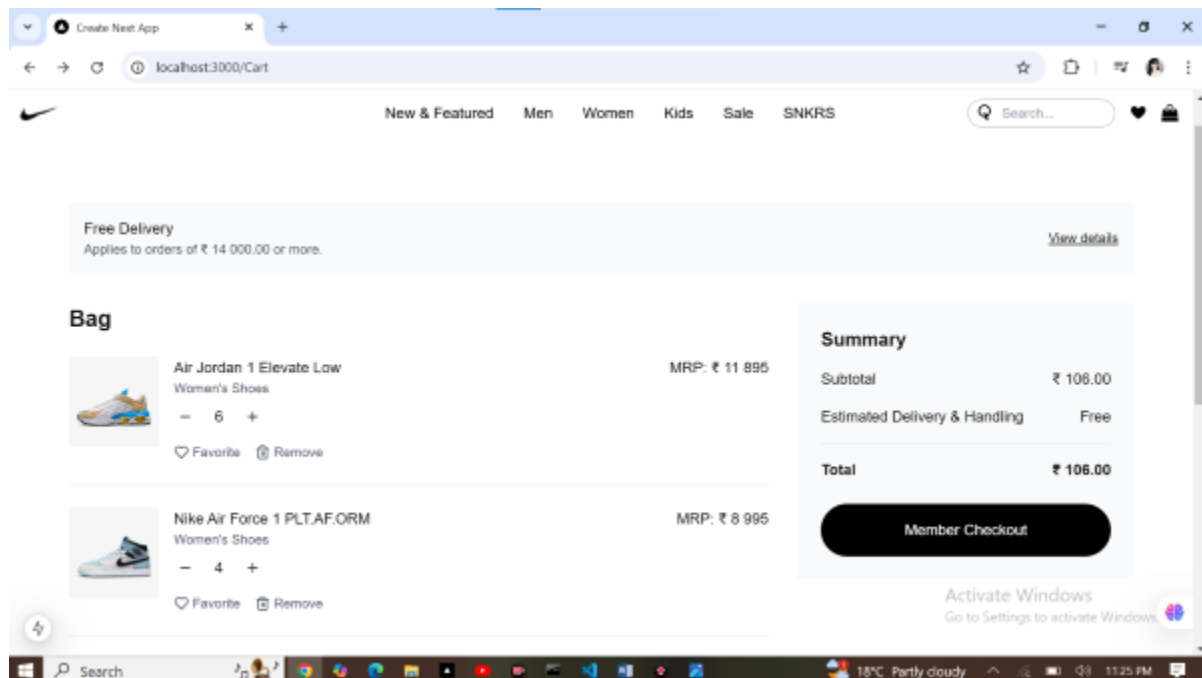
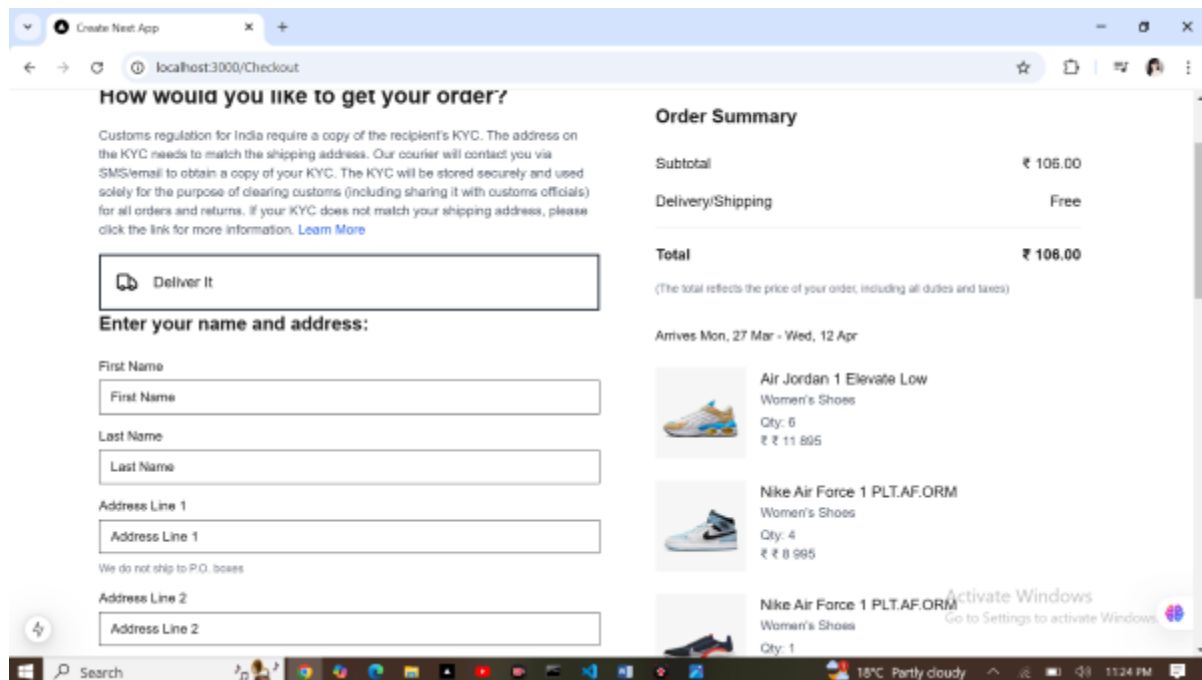
```
src > app > Context > CartContext.tsx > ...
1 'use client'
2
3 import React, { createContext, useContext, useState, useEffect } from 'react'
4 import { feature } from '@app/Cart/Data/detail'
5
6 type CartItem = {
7   id: number
8   quantity: number
9 }
10
11 type CartContextType = {
12   cartItems: CartItem[]
13   addToCart: (productId: number) => void
14   removeFromCart: (productId: number) => void
15   updateQuantity: (productId: number, quantity: number) => void
16 }
17
18 const CartContext = createContext<CartContextType | undefined>(undefined)
19
20 export function CartProvider({ children }: { children: React.ReactNode }) {
21   const [cartItems, setCartItems] = useState<CartItem[]>([])
22
23   // Load cart from localStorage on mount
24   useEffect(() => {
25     const savedCart = localStorage.getItem('cart')
26     if (savedCart) {
27       setCartItems(JSON.parse(savedCart))
28     }
29   }, [])
30 }
```

- I have implemented a validate dynamic route parameters to ensure they match expected formats (e.g. numeric IDs and validate slug).



Add filters and sorting for specific and better products according to demand.





Intuitive Design:

- The "Add to Cart" button is prominently placed and visually appealing, ensuring users can easily add items to their cart.

Item Quantity Adjustment:

- Users can easily increase and decrease the quantity of items directly from the cart.

