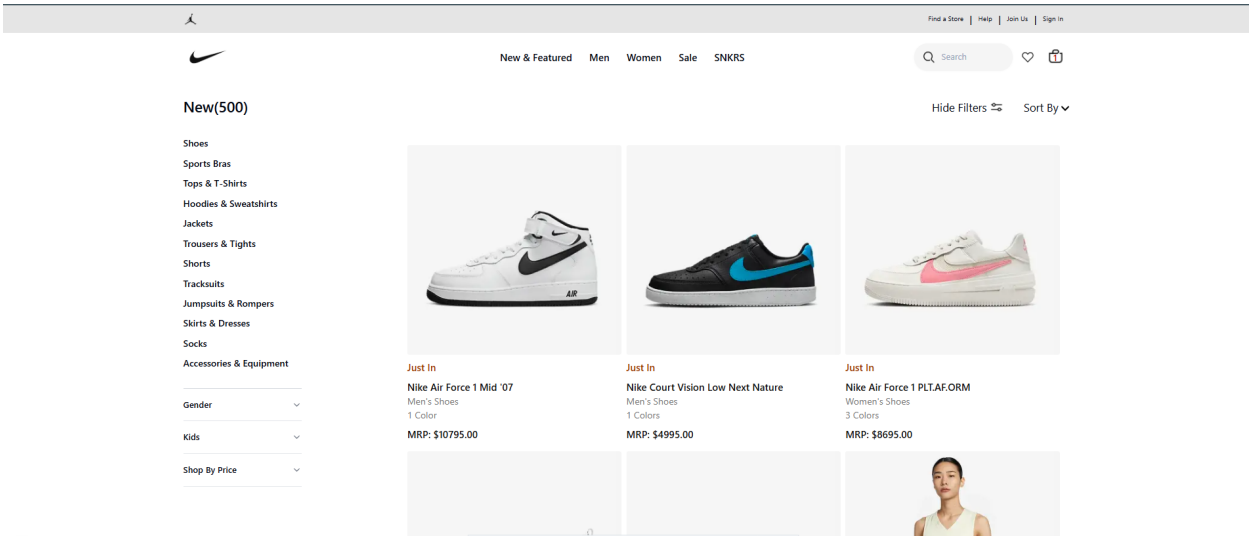
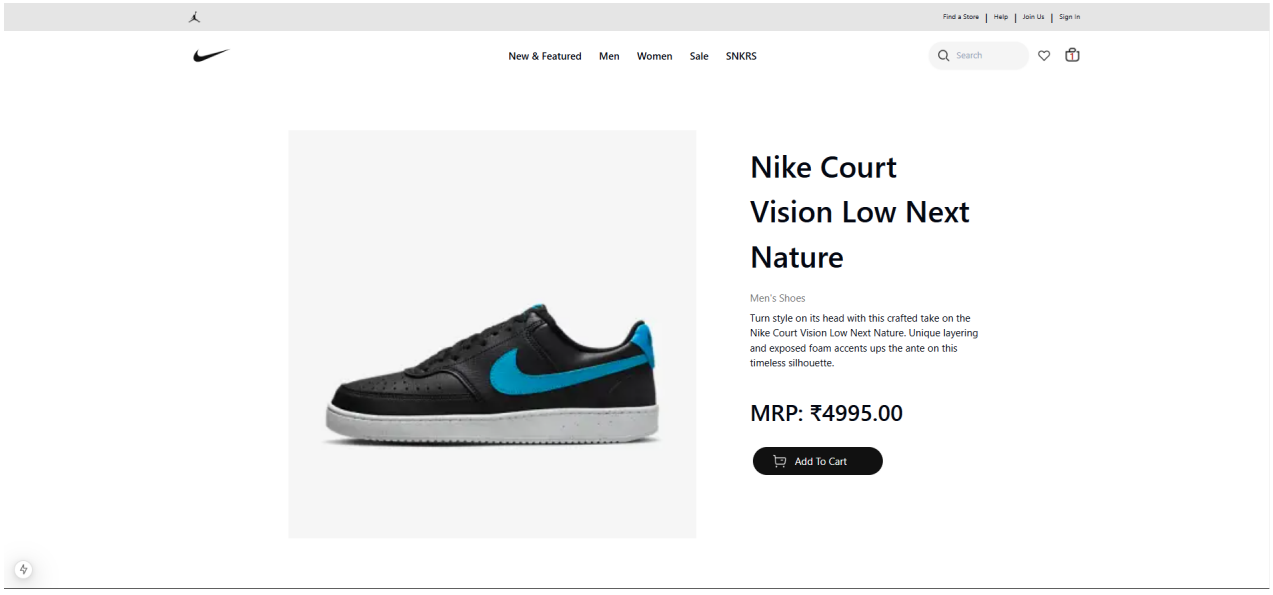
Day 4 - Dynamic Frontend Components – Nike

Screenshots of:

1. ***Product Listing Component:***

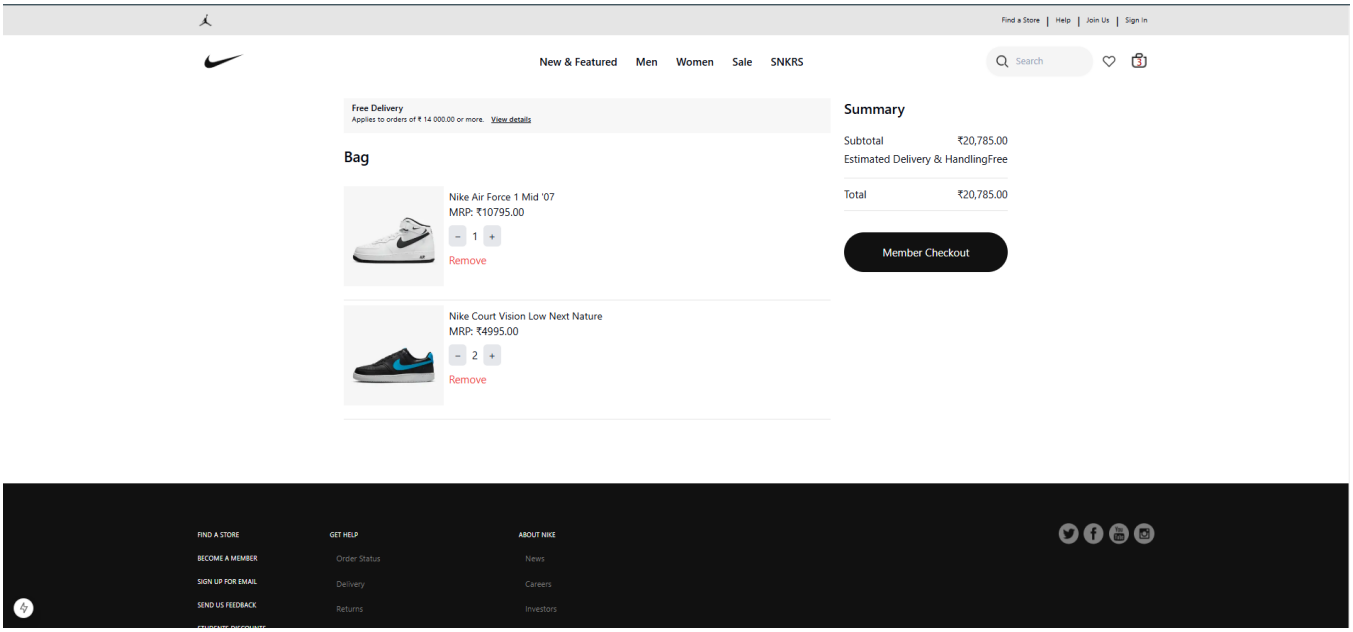


2. ***Dynamic Functionality:***



Umer Adnan (00053633)

3.Cart Functionality:



Code:

page.tsx M | product.tsx

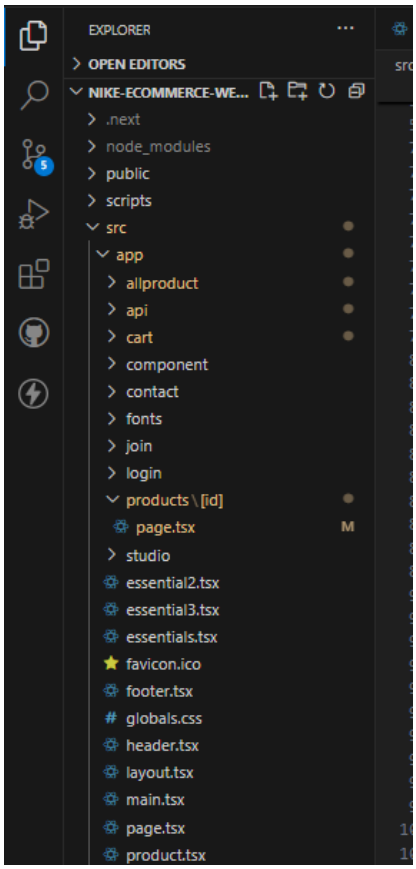
src > app > product.tsx > Product

```tsx
export default function Product() {
  <div className="flex justify-between py-8">···
  </div>

  {/* Slider */}
  <div className="overflow-hidden relative">
    <div
      className="flex transition-transform duration-500"
      style={{
        transform: `translateX(-${currentIndex * (370 + 15)}px)`,
        gap: "20px",
        width: `${cards.length * (370 + 15)}px`,
      }}
    >
      {cards.map((card, index) => (
        <div
          key={index}
          className="flex-shrink-0"
          style={{ width: "370px" }}
        >
          <div className="bestSellingBox">
            <div className="addToCartOverlay"><Link href={`../allproduct`}>View Product Page</Link></div>
            <Image src={card.image} alt={card.title} className="mainImage" />
          </div>
          <div className="flex justify-between text-[15px] font-medium py-2">
            <div>{card.title}</div>
            <div>{card.price}</div>
          </div>
          <div className="text-[#757575] text-[15px]">{card.category}</div>
        </div>
      ))}
    </div>
  </div>
</div>
  );
}
```

Product Details (Dynamic Routes):

- Shows a product image, name, description, price (original and discounted), and a discount percentage for real-time.
- Includes a "View Product" button linking to the product's detailed page using its `_id`.



API Integration:

1. Sanity API Fetching Logic: Since you're using Sanity CMS, you need to ensure data fetching is properly managed.

- **Sanity Client Setup** (if not already set up): In your `lib` folder, configure the Sanity client:

- Fetching Data in **useEffect:**

Dynamic Routing

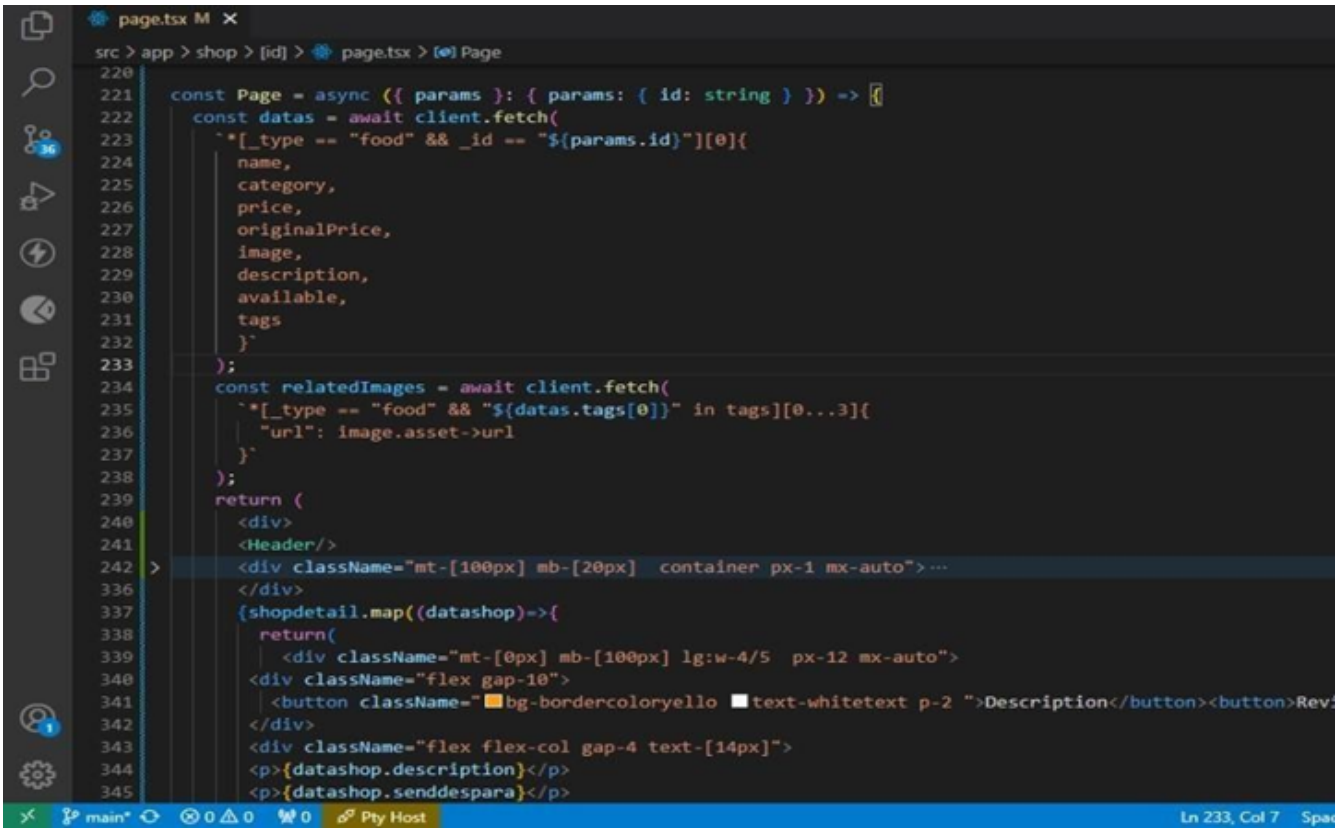
Dynamic routing is used to handle routes like /shop/:id for individual product pages.

1. Create Dynamic Route File:

In the pages directory, create a [id].tsx file inside the shop folder:

pages/shop/[id].tsx

2. Get Product Data Dynamically:



Final Checklist

Frontend Component Development	Styling and Responsiveness	Code Quality	Documentation and Submission	Final Review
✓	✓	✓	✓	✓