

Day 4 - Dynamic Frontend Components – Nike Shoes

Introduction:

On Day 4, focus on designing and dynamically frontend components for the shoes marketplace. The goal was to enhance the user experience by integrating data from sanity CMS and creating scalable and reusable components.

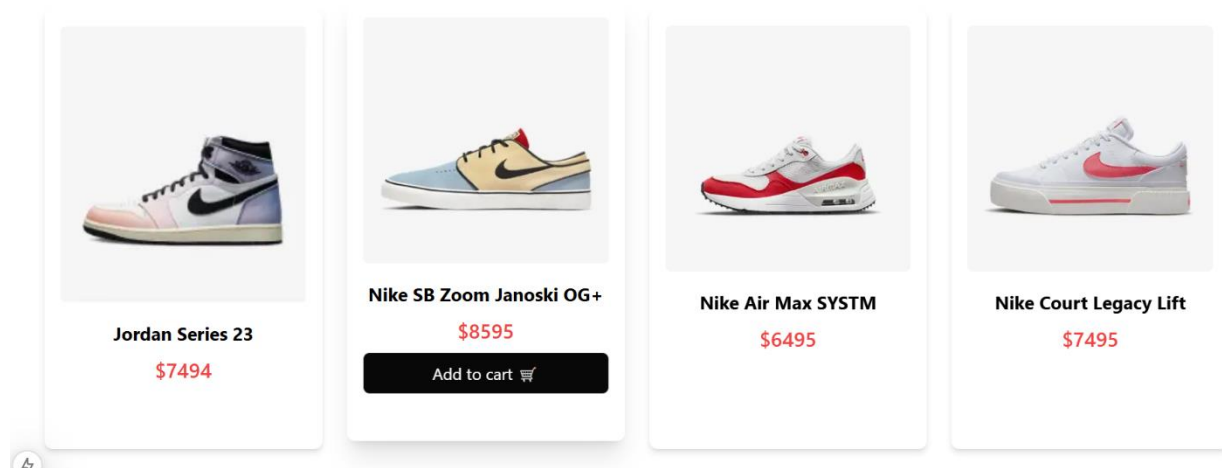
Key Objective:

- Building dynamic frontend components to display data from Sanity CMS.
- Implementing reusable and modular components.
- Understanding state management techniques.
- Applying responsive design and UI/UX best practices.


Components Developed:

- **Product Listing Component:** Displayed products dynamically in a grid layout shows product name, price.

Our Latest Products



- **Product Detail Component:** Provided details about the product name, description, price, category, color. Dynamic routing ensures each product has a unique URL for easy sharing.



Nike Air Max SC

The Nike Air Max SC combines a sporty design with all-day comfort. Its breathable mesh and leather upper ensure durability and a sleek look. The signature Max Air unit provides lightweight cushioning for a smooth ride. Perfect for casual wear, this sneaker offers both style and performance

Price: \$5995.00

Color: white

Category: Women's Shoes

Add to cart 🛒

- Search Bar Component

Men's Shoes



Nike Air Max SYSTM

\$6495

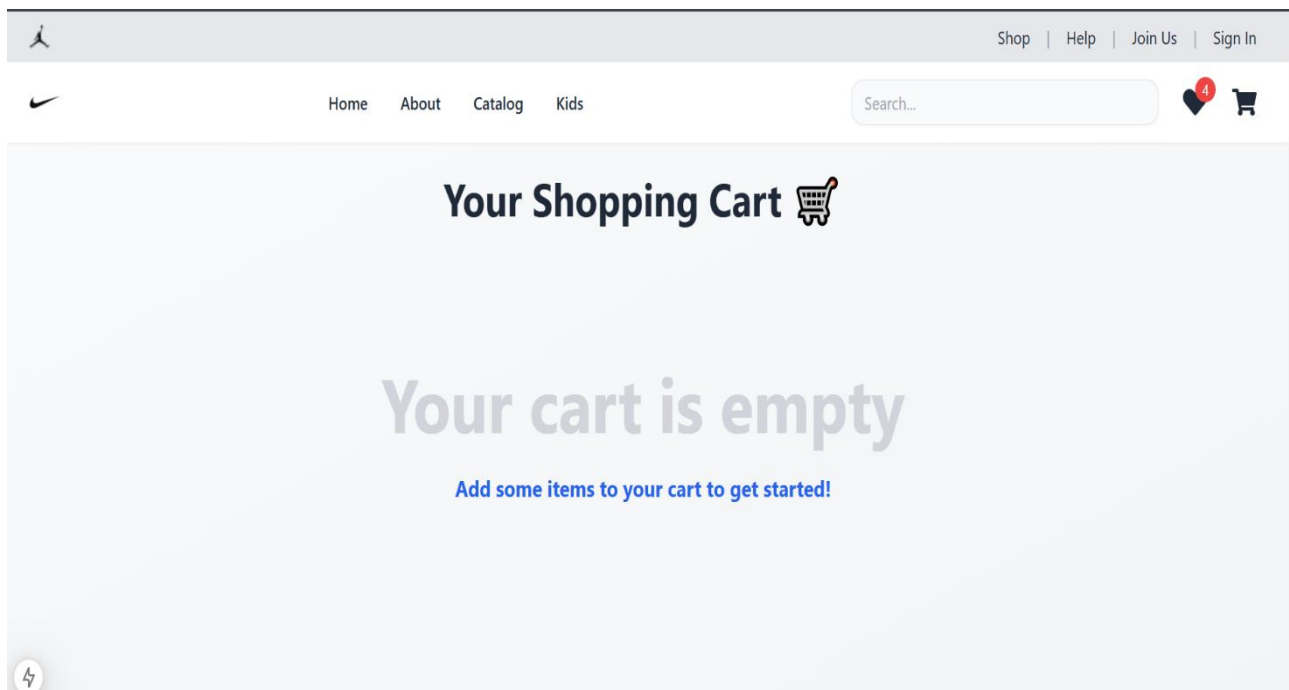
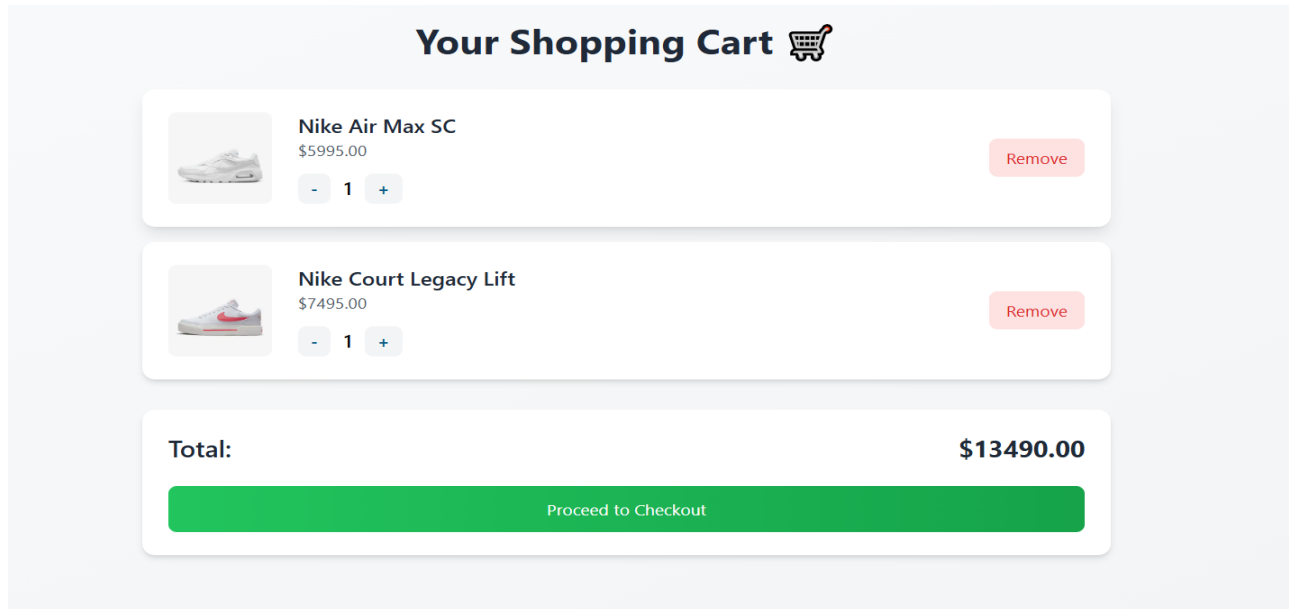


Nike Court Legacy Lift

\$7495

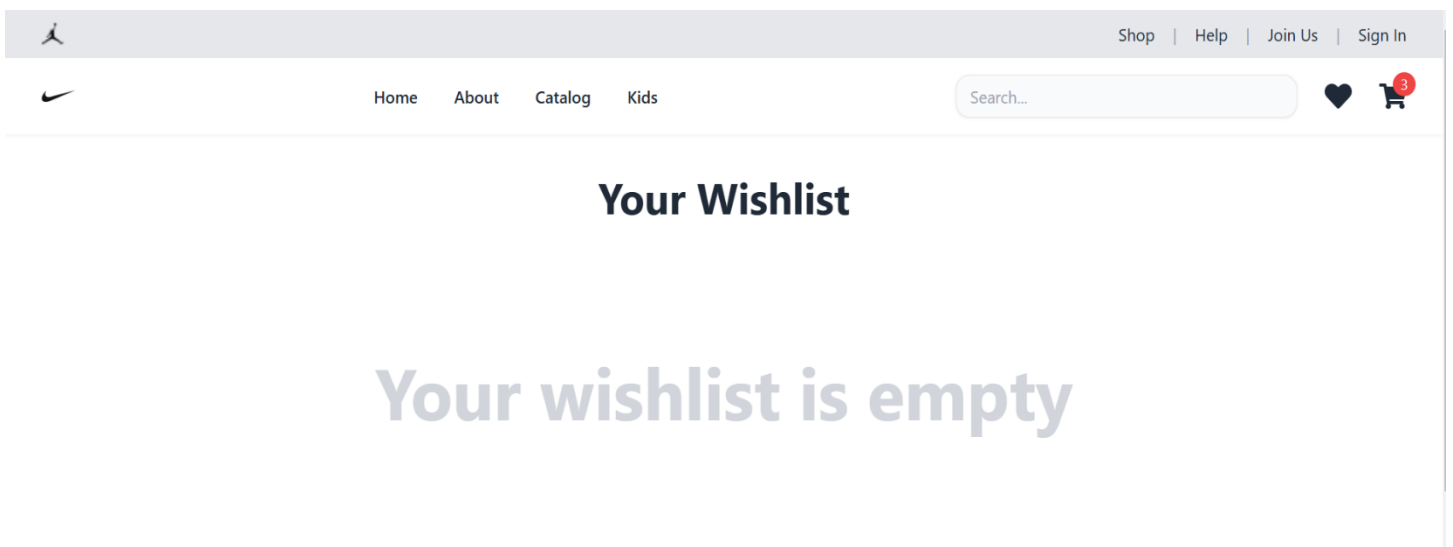
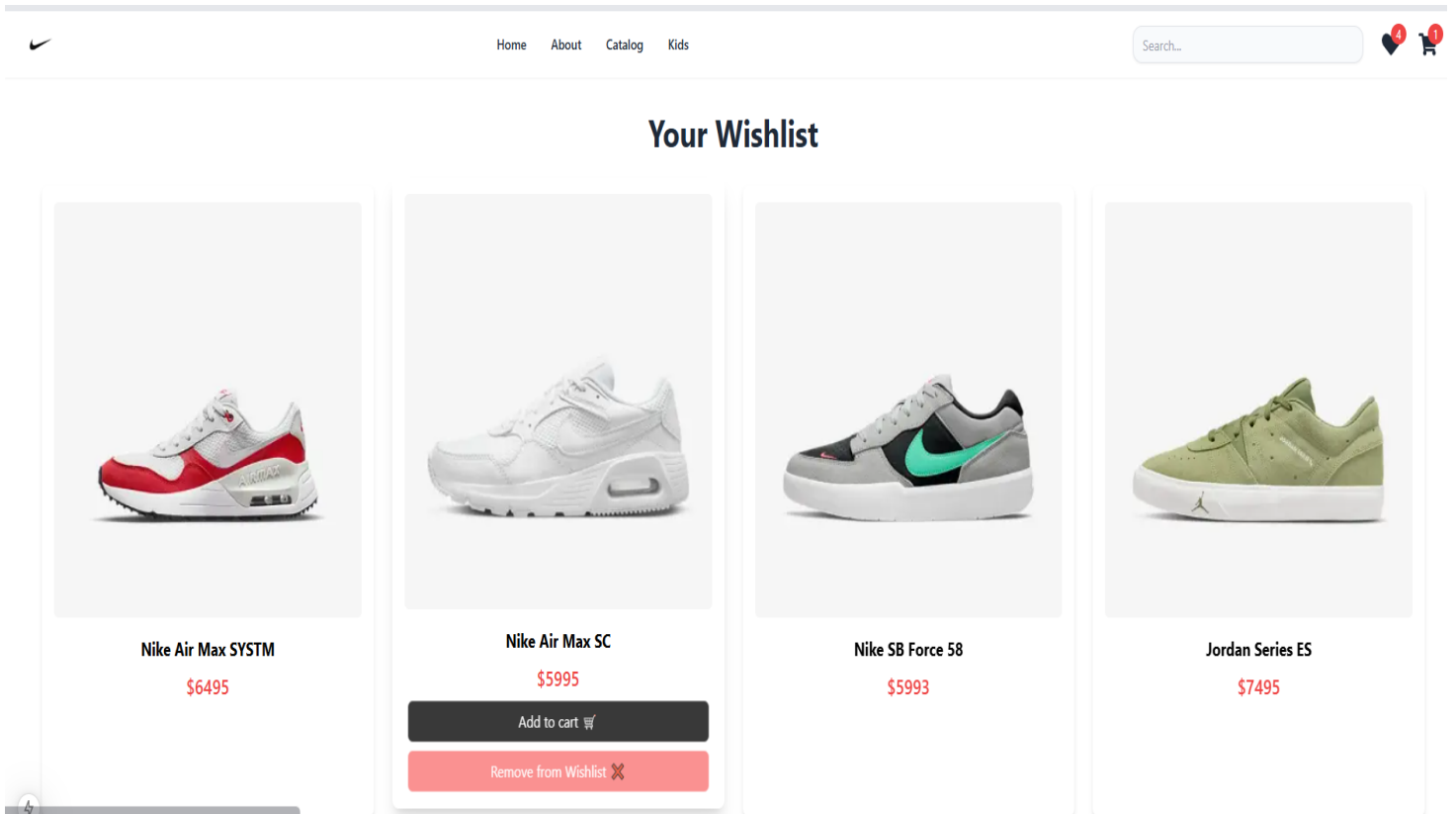
- **Add to Cart:**

User can add products to the cart, view them, and remove items dynamically.



- **Wishlist:**

User can add product to the wishlist view them, and remove items dynamically.



Checkout:

User can enter their delivery information, choose a payment method, and confirm their order to complete the purchase.

Billing Information

Enter Your Name and Address

Enter Your First Name

Enter Your Last Name

Enter Your Address

City

Zip Code

What's Your Contact Information?



Enter your Email address

Enter your phone number

Place Order

Order Summary

Subtotal:	\$13490
Discount:	-\$0
Total:	\$13490.00

	Nike Air Max SC Quantity: 1	\$5995
	Nike Court Legacy Lift Quantity: 1	\$7495

Code Snippets:

Product Listing Page:

```
const Products = () => {
  const [products, setProducts] = useState<Product[]>([]);
  const { addToCart } = useCart();
  const { addToWishlist } = useWishlist(); // Use useWishlist
  const [showAll, setShowAll] = useState(false);

  useEffect(() => {
    async function fetchProducts() {
      const fetchedProducts: Product[] = await client.fetch(allProducts);
      setProducts(fetchedProducts);
    }
    fetchProducts();
  }, []);

  const handleAddToCart = (e: React.MouseEvent, product: Product) => {
    e.preventDefault();
    addToCart(product);
    Swal.fire({
      position: "top-end",
      icon: "success",
      title: `${product.productName} added to cart`,
      showConfirmButton: false,
      timer: 1000,
    });
  };

  const handleAddToWishlist = (e: React.MouseEvent, product: Product) => {
    e.preventDefault();
    e.stopPropagation(); // Prevent Link from navigating
    addToWishlist(product);
  };
};
```

```
<div className="mx-8 px-4 py-8">
  <h1 className="text-3xl font-bold mb-6 text-gray-700">
    Our Latest Products
  </h1>
  <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-6">
    {(showAll ? products : products.slice(0, 8)).map((product) => (
      <div
        key={product._id}
        className="bg-white shadow-md p-4 rounded-lg hover:shadow-xl transition-shadow duration-300 cur
      >
        <Link href={`~/productdetails/${product.slug.current}`}>
          <div className="relative overflow-hidden rounded-md">
            {product.image && (
              <div className="relative">
                <Image
                  src={urlFor(product.image).url()}
                  alt={product.productName}
                  width={300}
                  height={300}
                  className="w-full object-cover rounded-md transform transition-transform duration-300 g
                />
                { /* Heart Icon */ }
                <div
                  className="absolute top-2 right-2 bg-white p-2 rounded-full opacity-0 group-hover:opa
                  onClick={(e) => handleAddToWishlist(e, product)}
                >
                  <FaHeart className="text-red-500" />
                </div>
              </div>
            )}
          </div>
        </Link>
      </div>
    ))}
  </div>
</div>
```

Product Details Page:

```
export default function ProductPage() {
  const { slug } = useParams() as { slug: string | string[] };

  const slugStr: string = typeof slug === "string" ? slug : slug ? slug[0] : "";

  const [product, setProduct] = useState<Product | null>(null);
  const [loading, setLoading] = useState(true);

  const getProduct = async (slug: string) => {
    try {
      const data = await client.fetch(
        groq`*[_type == "product" && slug.current == $slug][0]{
          _id,
          productName,
          _type,
          image,
          price,
          description,
          colors,
          inventory,
          category,
          "slug": slug.current
        }`,
        { slug }
      );
      setProduct(data);
      setLoading(false);
    } catch (err) {
      console.error("Error fetching product:", err);
      setLoading(false);
    }
  };
}
```

```
<div className="rounded-lg max-w-5xl w-full md:flex bg-white shadow-lg overflow-hidden">
  <div className="w-full md:w-1/2 p-8 flex items-center justify-center bg-gray-100">
    {product.image && (
      <Image
        src={urlFor(product.image).url()}
        alt={product.productName}
        width={600}
        height={600}
        className="w-full h-auto object-cover rounded-md transform transition duration-500 hover:scale-105"
      />
    )}
  </div>

  <div className="w-full md:w-1/2 p-8 flex flex-col justify-center">
    <h1 className="text-4xl font-bold text-gray-800 mb-4 tracking-wide">
      {product.productName}
    </h1>
    <p className="text-gray-600 mt-4 text-lg leading-relaxed">
      {product.description}
    </p>
    <p className="text-xl font-semibold text-red-600 mt-4">
      Price: ${Number(product.price).toFixed(2)}
    </p>
    <p className="text-sm text-gray-800 font-bold mt-2">
      Color: {product.colors}
    </p>
    <p className="text-sm text-gray-800 font-bold mt-2">
      Category: {product.category}
    </p>
  </div>
</div>
```



```

<div className="rounded-lg max-w-5xl w-full md:flex bg-white shadow-lg overflow-hidden">
  <div className="w-full md:w-1/2 p-8 flex items-center justify-center bg-gray-100">
    {product.image && (
      <Image
        src={urlFor(product.image).url()}
        alt={product.productName}
        width={600}
        height={600}
        className="w-full h-auto object-cover rounded-md transform transition duration-500 hover:scale-105"
      />
    )}
  </div>

  <div className="w-full md:w-1/2 p-8 flex flex-col justify-center">
    <h1 className="text-4xl font-bold text-gray-800 mb-4 tracking-wide">
      {product.productName}
    </h1>
    <p className="text-gray-600 mt-4 text-lg leading-relaxed">
      {product.description}
    </p>
    <p className="text-xl font-semibold text-red-600 mt-4">
      Price: ${Number(product.price).toFixed(2)}
    </p>
    <p className="text-sm text-gray-800 font-bold mt-2">
      Color: {product.colors}
    </p>
    <p className="text-sm text-gray-800 font-bold mt-2">
      Category: {product.category}
    </p>
  </div>
</div>

```

Cart Page:

```

const CartPage = () => {
  const { cartItems, removeFromCart, updateCartQuantity } = useCart();
  const router = useRouter();

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

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

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

```

Wishlist Page:

```
const FavoritesPage = () => {
  const { wishlist, removeFromWishlist } = useWishlist();
  const { addToCart } = useCart();

  // Handle Add to Cart
  const handleAddToCart = (e: React.MouseEvent, product: any) => {
    e.preventDefault();
    addToCart(product);
    Swal.fire({
      position: "top-end",
      icon: "success",
      title: `${product.productName} added to cart`,
      showConfirmButton: false,
      timer: 1000,
    });
  };

  // Handle Remove from Wishlist
  const handleRemoveFromWishlist = (e: React.MouseEvent, product: any) => {
    e.preventDefault();
    removeFromWishlist(product._id);
    Swal.fire({
      position: "top-end",
      icon: "success",
      title: `${product.productName} removed from wishlist`,
      showConfirmButton: false,
      timer: 1000,
    });
  };
};
```

```
<div className="mx-8 px-4 py-8">
  <h1 className="text-4xl font-bold mb-8 text-gray-800 text-center">
    Your Wishlist
  </h1>
  {wishlist.length === 0 ? (
    <div className="py-20">
      <h1 className="text-6xl font-bold text-gray-300 text-center">
        Your wishlist is empty
      </h1>
    </div>
  ) : (
    <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-6">
      {wishlist.map((product) => (
        <div
          key={product._id}
          className="bg-white shadow-md p-4 rounded-lg hover:shadow-xl transition-shadow duration-300"
        >
          <Link href={` /productdetails/${product.slug.current}`}>
            <div className="relative overflow-hidden rounded-md">
              {product.image && (
                <Image
                  src={urlFor(product.image).url()}
                  alt={product.productName}
                  width={300}
                  height={300}
                  className="w-full object-cover rounded-md transition-transform duration-300"
                />
              )}
            </div>
          </Link>
        </div>
      )}
    </div>
  )}
```

Checkout Page:

```
export default function CheckoutPage() {
  const [cartItems, setCartItems] = useState<Product[]>([]);
  const [discount, setDiscount] = useState<number>(0);
  const [formValues, setFormValues] = useState({
    firstName: "",
    lastName: "",
    address: "",
    city: "",
    zipCode: "",
    phone: "",
    email: "",
  });

  const [formErrors, setFormErrors] = useState({
    firstName: false,
    lastName: false,
    address: false,
    city: false,
    zipCode: false,
    phone: false,
    email: false,
  });

  useEffect(() => {
    setCartItems(getCartItems());
    const appliedDiscount = localStorage.getItem("appliedDiscount");
    if (appliedDiscount) {
      setDiscount(Number(appliedDiscount));
    }
  });

  const subtotal = cartItems.reduce(
    (total, item) => total + item.price * item.inventory,
    0
  );
  const total = Math.max(subtotal - discount, 0);

  const handleInputChange = (e: React.ChangeEvent<HTMLInputElement>) => {
    setFormValues({
      ...formValues,
      [e.target.id]: e.target.value,
    });
  };

  const validateForm = () => {
    const errors = {
      firstName: !formValues.firstName,
      lastName: !formValues.lastName,
      address: !formValues.address,
      city: !formValues.city,
      zipCode: !formValues.zipCode,
      phone: !formValues.phone,
      email: !formValues.email,
    };
    setFormErrors(errors);
    return Object.values(errors).every((error) => !error);
  };

  const handlePlaceOrder = async () => {
    if (validateForm()) {
      // ...
    }
  };
}
```

