# Hackathon Day 4

# Day 4 – Dynamic Frontend Components Report for E-Commerce

## 1. Introduction

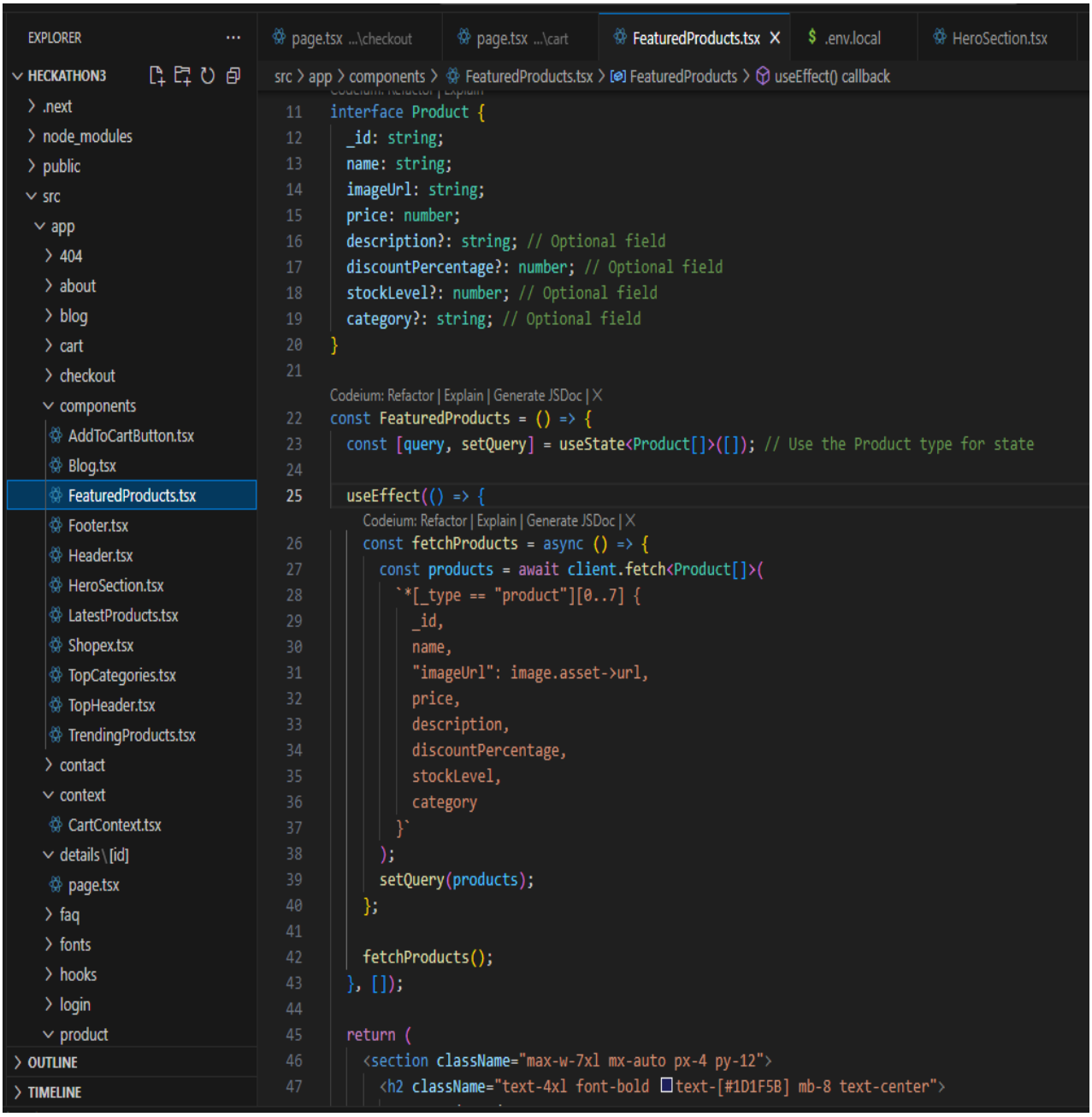
This report outlines the development of interactive and responsive frontend components for **E-Commerce**. The key focus areas include:

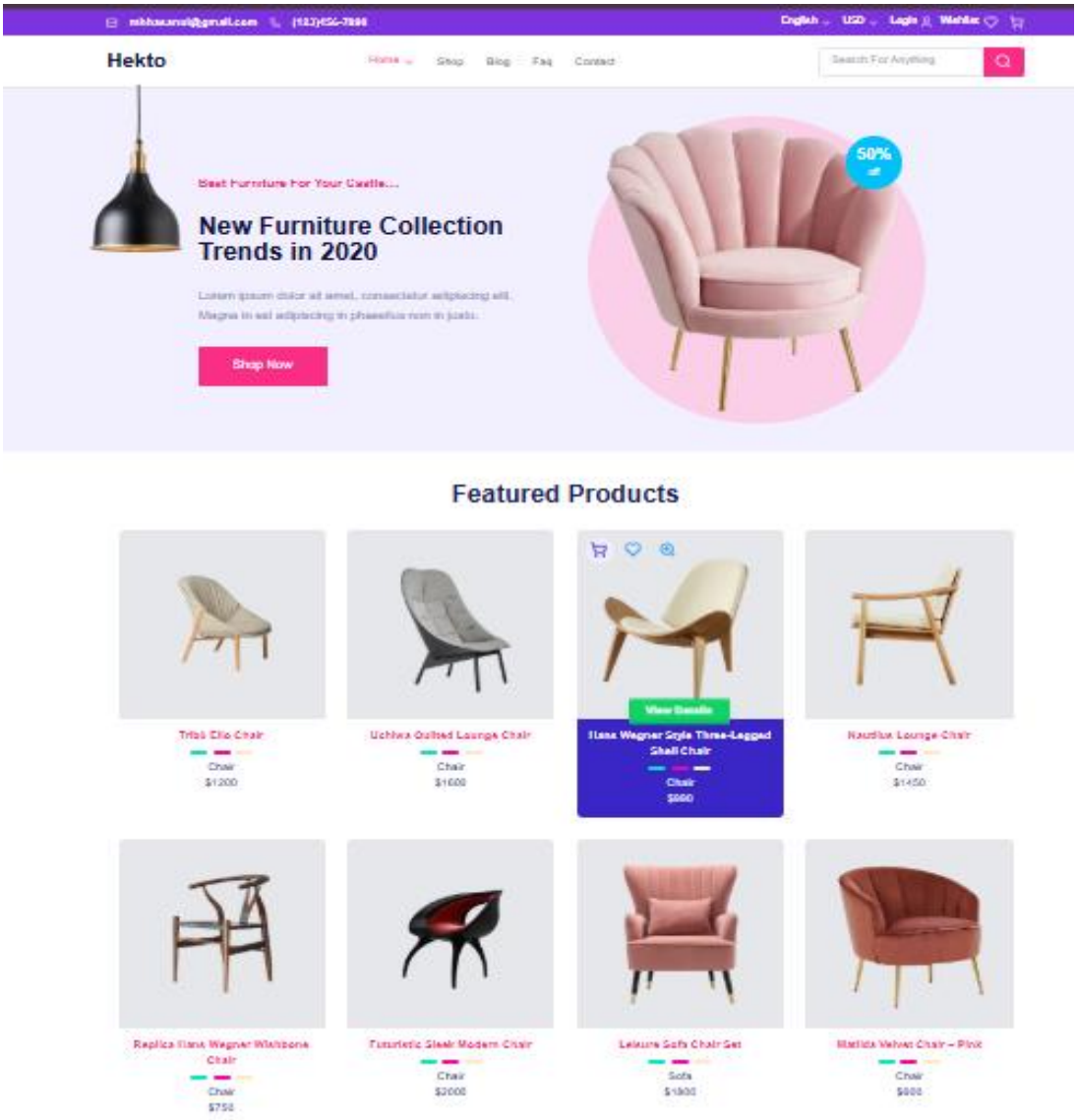
✅ **Dynamic product display**
✅ **Product cards component displayed on the home page**
✅ **Creating a structured product listing page**
✅ **Designing a detailed product view page**
✅ **Developing a functional shopping cart for order management**
✅ **Order placement on the checkout page will send data to Sanity**

## 2. Tasks Overview

### Step 1: Product Component

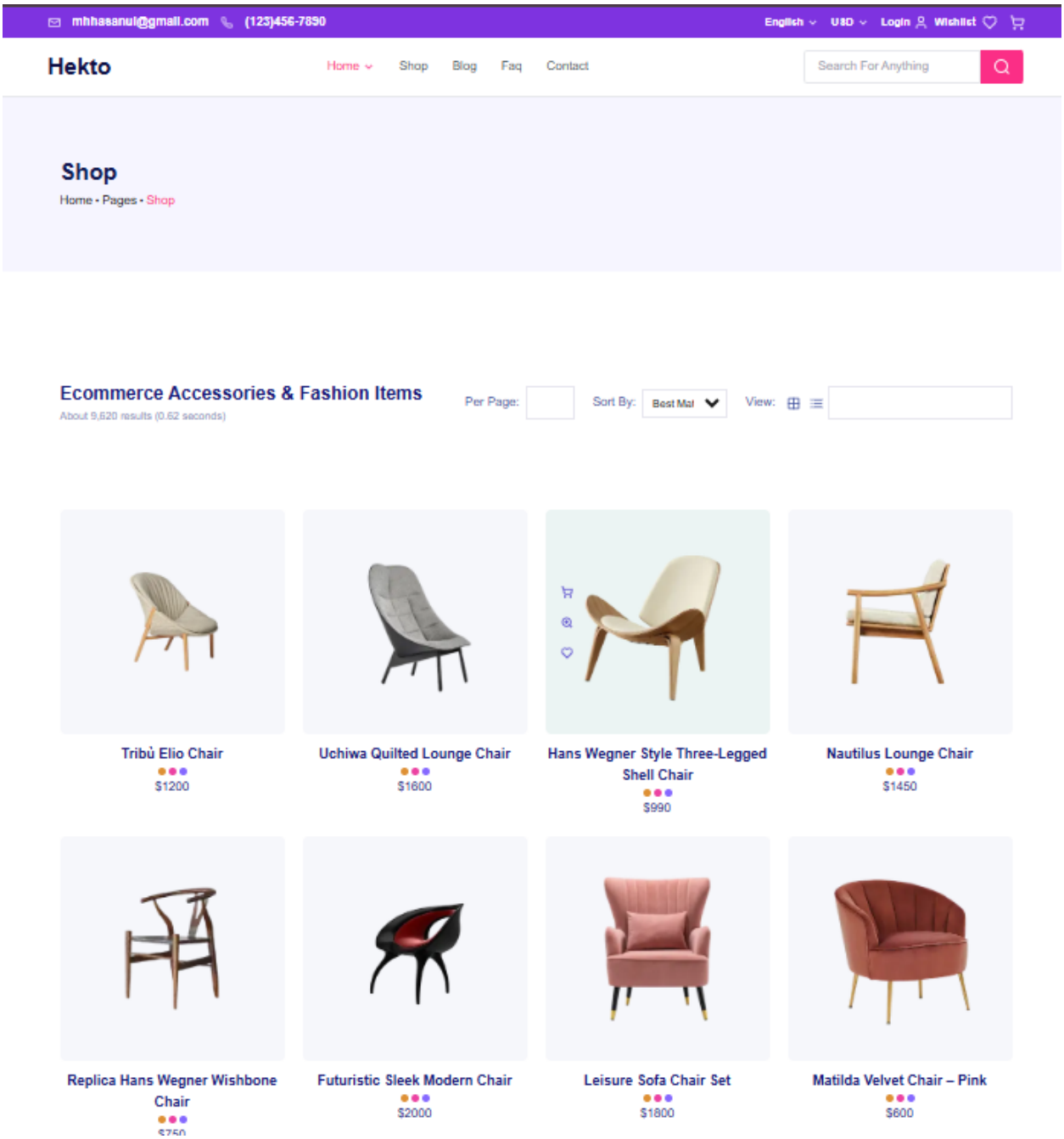
- Retrieved product data dynamically from the API.
- Fetched product data in a grid layout within a dedicated component on the home page, displaying names and prices.


EXPLORER
HECKATHON3
.next
node_modules
public
src
app
404
about
blog
cart
checkout
components
AddToCartButton.tsx
Blog.tsx
FeaturedProducts.tsx
Footer.tsx
Header.tsx
HeroSection.tsx
LatestProducts.tsx
Shopex.tsx
TopCategories.tsx
TopHeader.tsx
TrendingProducts.tsx
contact
context
CartContext.tsx
details\[id]
page.tsx
faq
fonts
hooks
login
product
OUTLINE
TIMELINE
page.tsx ...\checkout
page.tsx ...\cart
FeaturedProducts.tsx
.env.local
HeroSection.tsx
src > app > components > FeaturedProducts.tsx > FeaturedProducts > useEffect() callback
interface Product {
_id: string;
name: string;
imageUrl: string;
price: number;
description?: string; // Optional field
discountPercentage?: number; // Optional field
stockLevel?: number; // Optional field
category?: string; // Optional field
}
Codeium: Refactor | Explain | Generate JSDoc | X
const FeaturedProducts = () => {
const [query, setQuery] = useState<Product[]>([]); // Use the Product type for state
useEffect(() => {
Codeium: Refactor | Explain | Generate JSDoc | X
const fetchProducts = async () => {
const products = await client.fetch<Product[]>(
`*[_type == "product"][0..7] {
_id,
name,
"imageUrl": image.asset->url,
price,
description,
discountPercentage,
stockLevel,
category
}`
);
setQuery(products);
};
fetchProducts();
}, []);
return (
<section className="max-w-7xl mx-auto px-4 py-12">
<h2 className="text-4xl font-bold text-[#1D1F5B] mb-8 text-center">



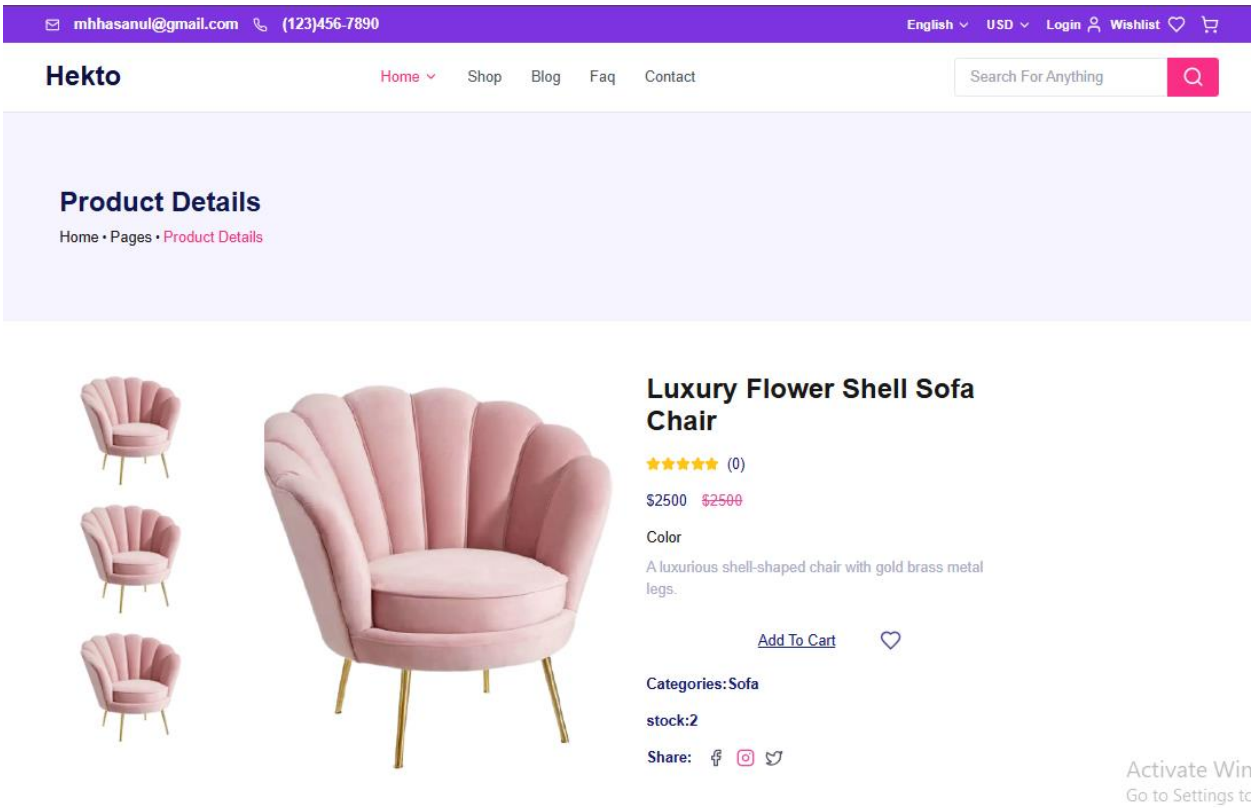

## Step 3: Product Listing

- Retrieved product data dynamically from the API.
- Showcased products in a flexible grid layout, including names and prices.



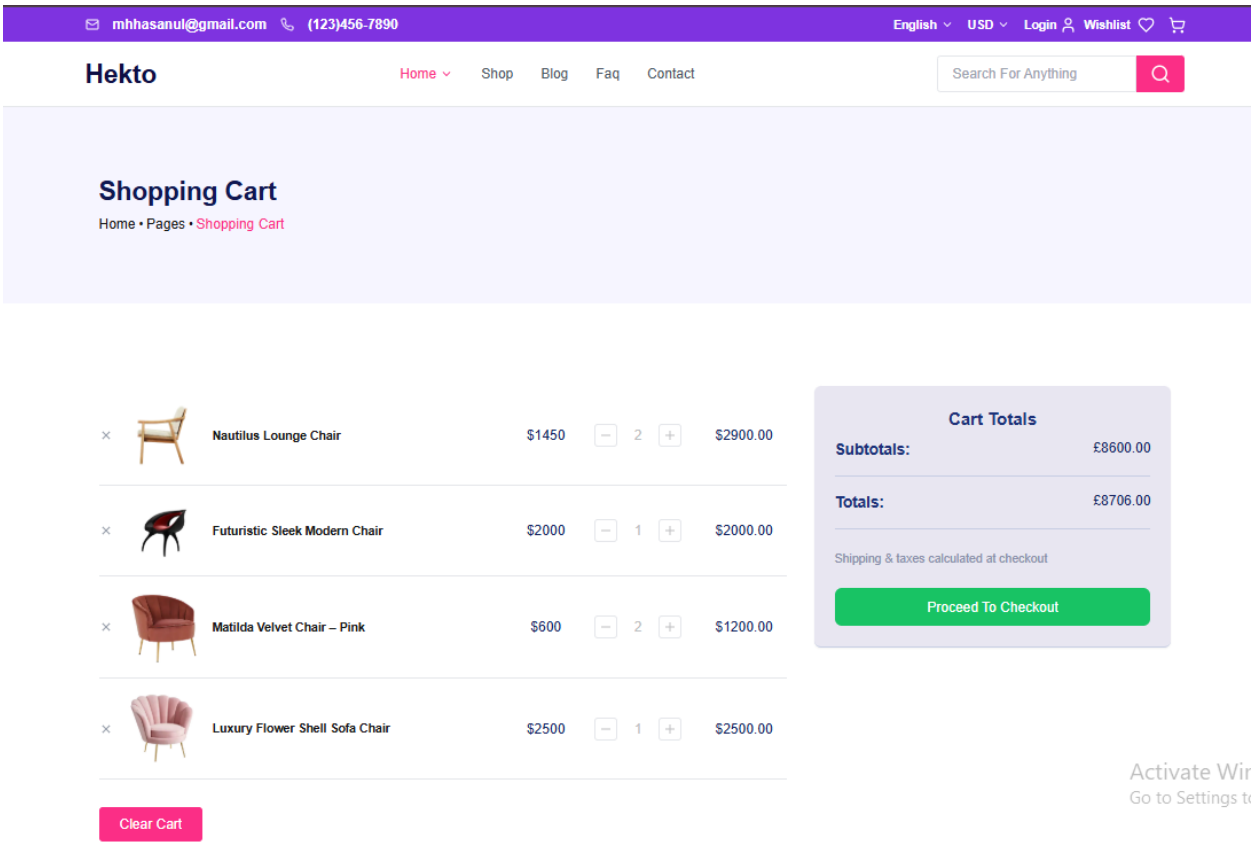

## Step 4: Product Detail Page

- Developed a dynamic product detail page with individual routing.
- Included an "Add to Cart" button for seamless shopping. When the 'Add to Cart' button is clicked, the items are successfully added to the shopping cart!



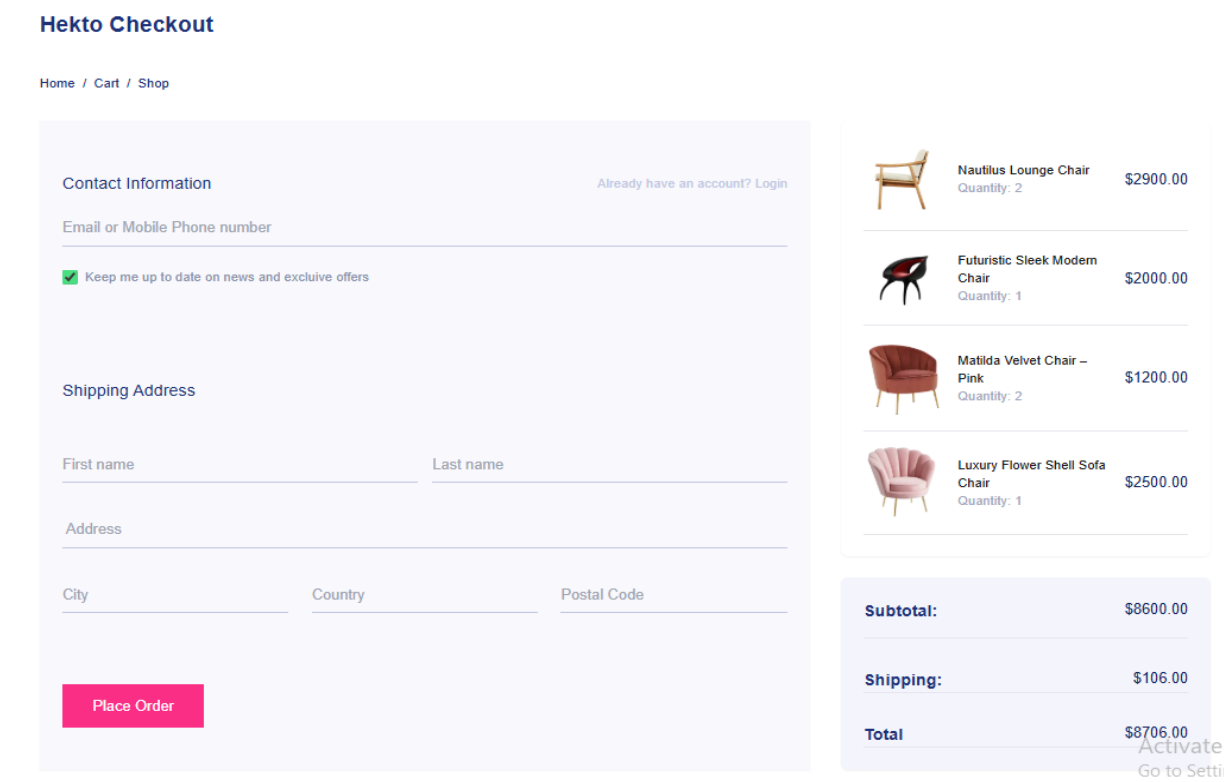

## Step 5: Shopping Cart Functionality

- Allowed users to add and remove products from the cart.
- Implemented real-time cart updates and a checkout option.

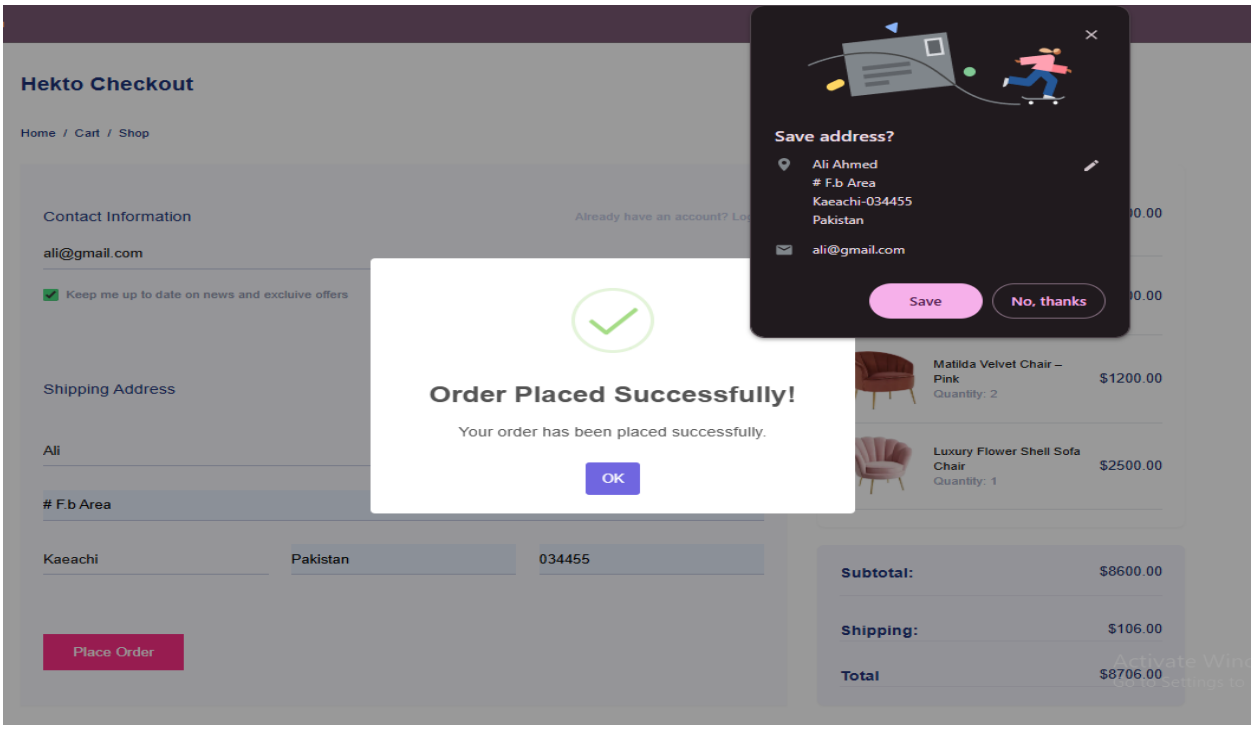


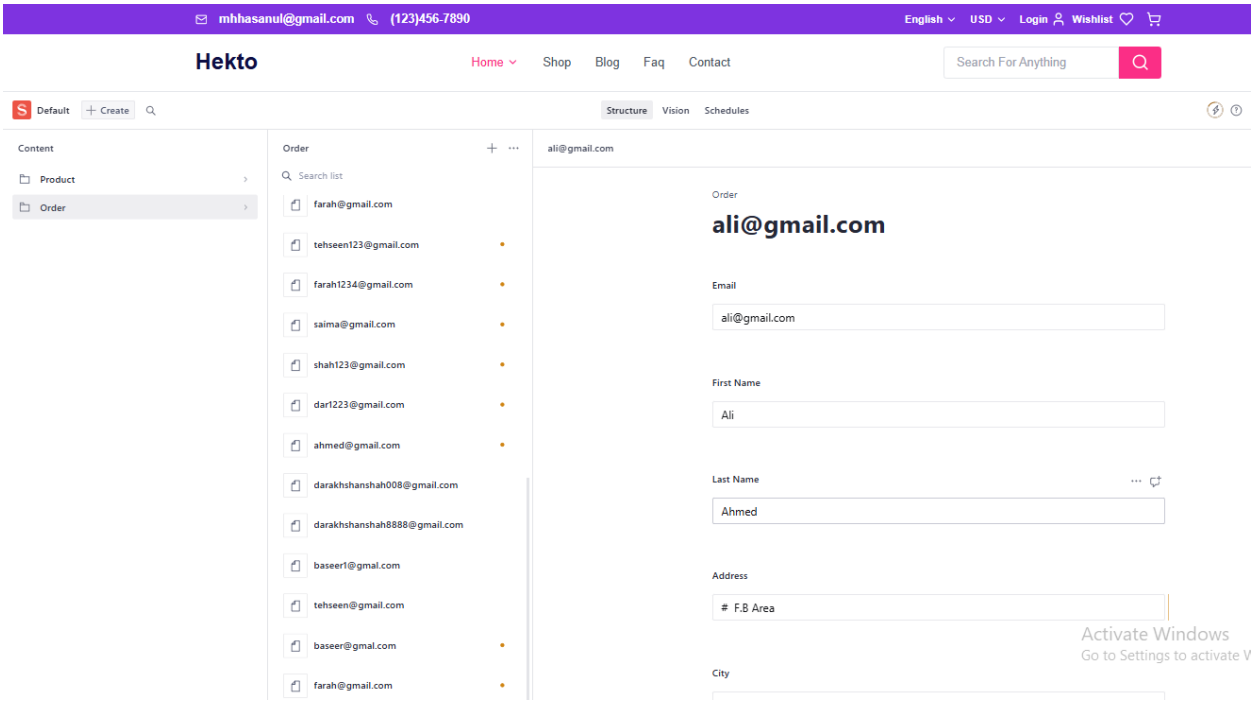

## Step 6: Successful Checkout

- Clicking the checkout button navigates the user to the checkout page.
- Users can proceed by entering their checkout details.




Order placement on the checkout page will send data to **Sanity**






Here is a structured **Self-Validation Checklist** in a table format, based on your report. This will provide a clear framework for validation and ensure all tasks are completed efficiently.

## Self-Validation Checklist for Frontend Development

### Frontend Component Development ✓ ✗

✓ Dynamic product listing

✓ Product detail page functionality

✗ Category filter component

✗ Search bar functionality

✓ Cart functionality (add/remove items)

✓ Checkout functionality

✗ Login/SignUp Page

**Styling and Responsiveness** ✓ ✗

✓ Responsive grid layout for product listings

✓ Consistent UI design across all pages

✓ Mobile responsiveness

**Code Quality** ✓ ✗

✓ Modular and reusable components

✓ Clean and organized code structure

✓ Proper use of state management and data binding

✓ Optimization for performance and loading speed

**Documentation and Submission** ✓ ✗

✓ Steps for setting up and running the application

✓ Code comments and explanations

✗ Submission before the deadline

**Final Review** ✓ ✗

✓ All functionality tested (products, cart, checkout)

✓ UI/UX reviewed for user-friendliness

✓ No bugs/issues identified in the development

**Professional Practices Emphasized** ✓ ✗

✓ Modular and reusable component design

✓ State management and dynamic data binding

✓ Responsive and user-friendly UI design

✓ Thorough documentation for code and processes

## Conclusion

The interactive frontend components were successfully developed, improving the user experience with responsive designs and dynamic cart features. Attached screenshots confirm the completed implementation.