Day 4 Submission: Building Dynamic Frontend Components for FurnitureMart.pk

Introduction

In this phase, I focused on building dynamic and reusable frontend components. These components enhance the user experience by providing seamless navigation, product browsing, cart management, and filtering capabilities.

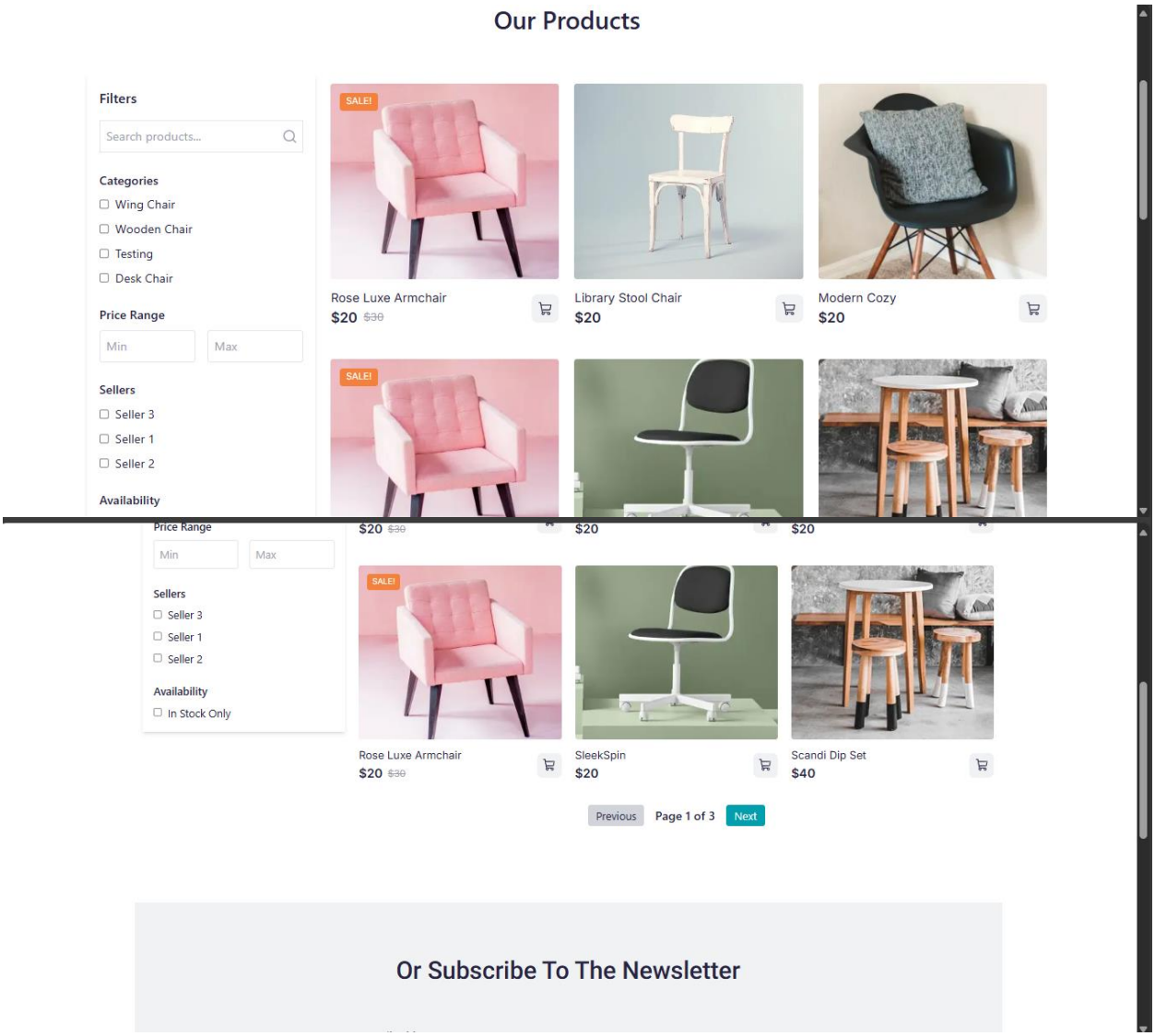
Completed Components

1. **Header and Footer**:
    - Provides consistent navigation and branding across all pages.
    - Includes links to Home, Cart, About, and other key sections.
    - Responsive and styled with Tailwind CSS.
2. **Product Card**:
    - Dynamically displays product details, including name, price, stock status, and images.
    - Designed for responsiveness and reusability in product grids.
3. **Product Detail**:
    - Displays detailed information about a selected product, such as description, price, available customization options, and an "Add to Cart" button.
    - Implements dynamic routing in Next.js to fetch product-specific data.
4. **Cart Component**:
    - Lists all items added to the cart with quantities and total price.
    - Includes functionality to update item quantities or remove items from the cart.
    - Uses state management (e.g., Context API) for dynamic updates.
5. **Toast Notifications**:
    - Provides real-time feedback for user actions like adding items to the cart, completing checkout, or encountering errors.
    - Customizable for different notification types (success, error, info).

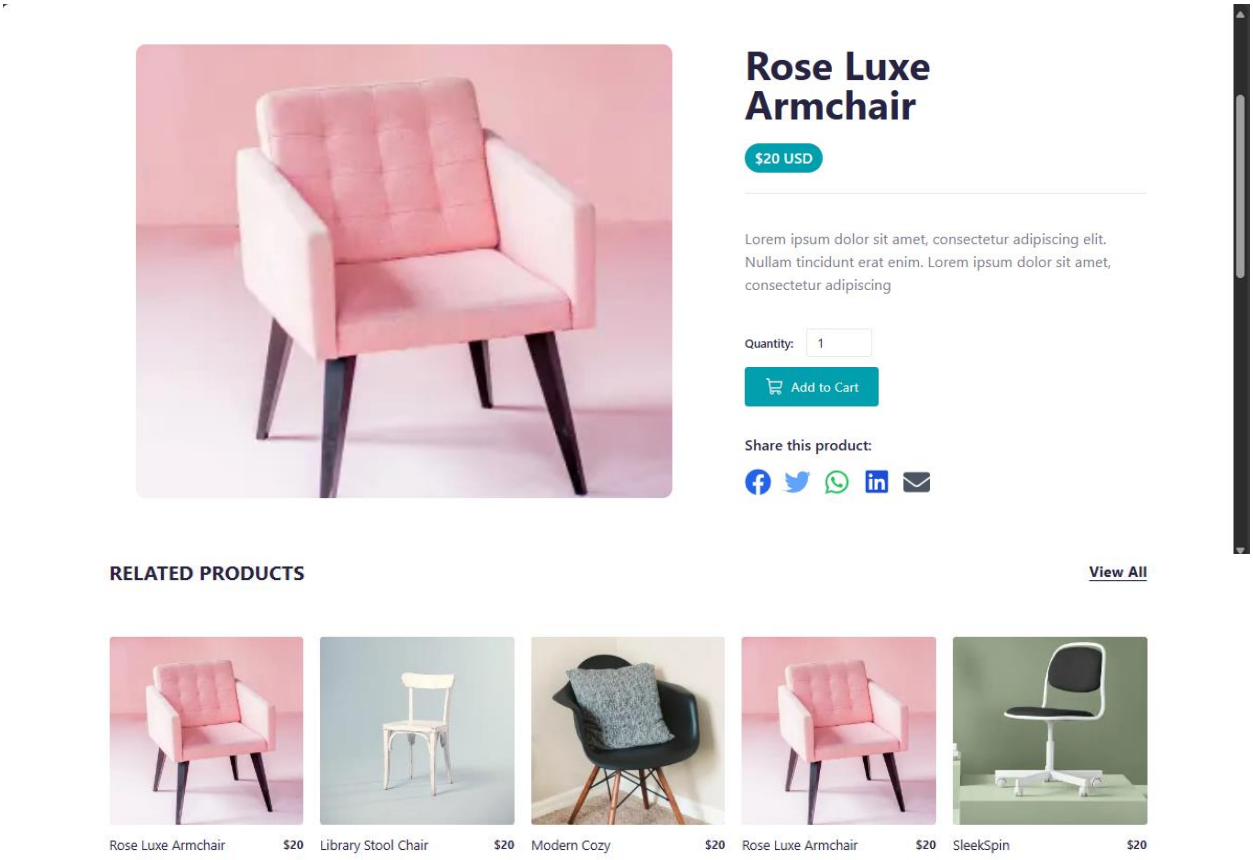
6. **Social Sharing**:
    - Enables users to share product details on social media platforms.
    - Uses dynamically generated URLs for seamless sharing.
7. **Search Bar**:
    - Allows users to search for products by name or tags.
    - Includes debounced input to optimize API calls or search logic.
8. **Filter Panel**:
    - Offers basic and advanced filtering options, such as category, price range, stock status, and tags.
    - Dynamically updates product listings based on selected filters.
9. **Pagination**:
    - Breaks down large product listings into multiple pages.
    - Includes "Next" and "Previous" buttons for easy navigation.
10. **Checkout Component**:
    - Collects user details and payment information.
    - Integrates seamlessly with the cart and backend order API.
11. **Advanced Filters**:
    - Provides multi-select options for categories, price sliders, and availability toggles.
    - Enhances the product search experience by combining multiple filters.
12. **Related Products**:
    - Displays products similar to the one being viewed, based on categories or tags.
    - Encourages users to explore and add more items to their cart.
    - Dynamically fetched from Sanity CMS and displayed on the product details page.

Screenshots:

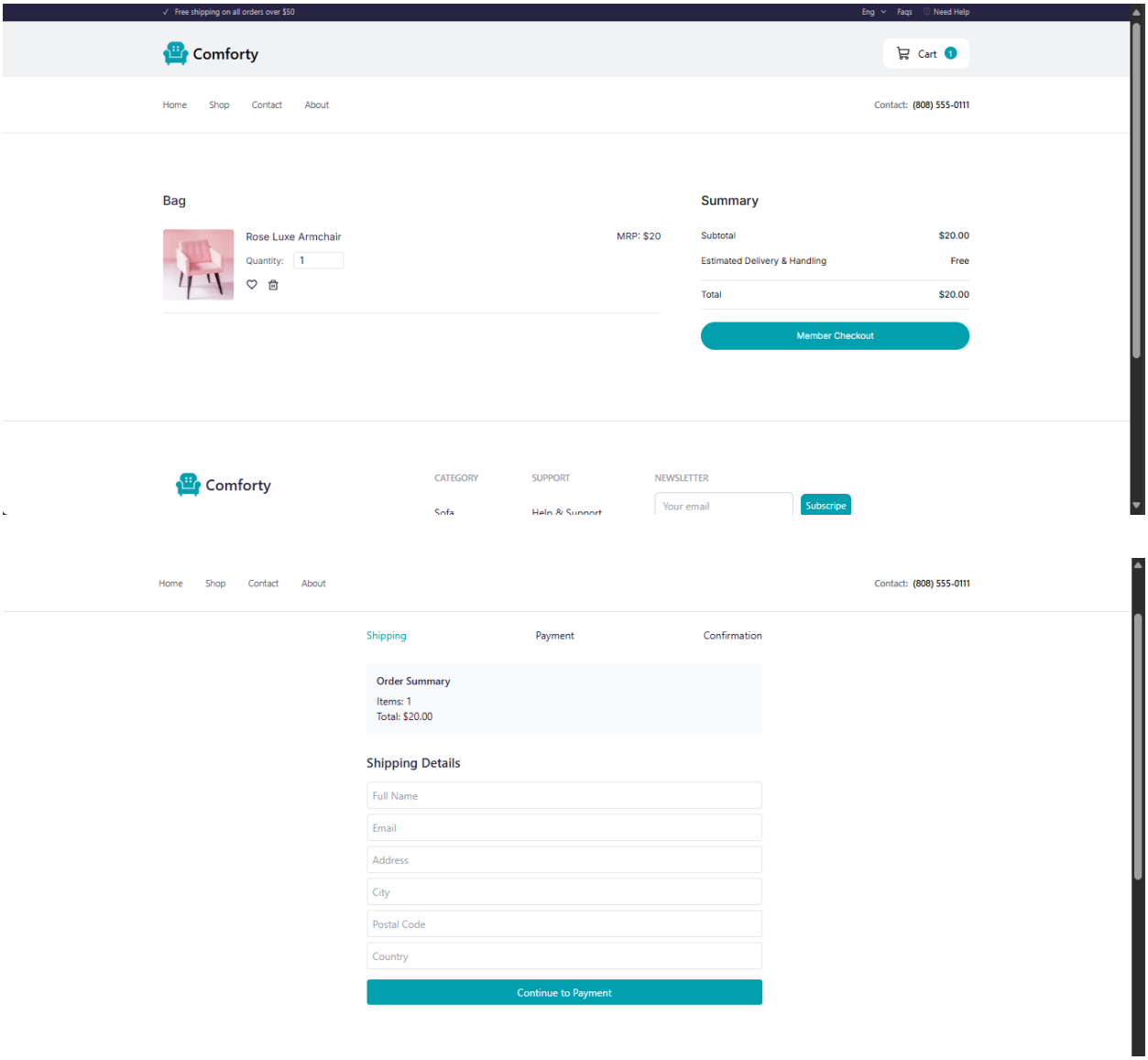
1. Product listing page with filters and pagination.



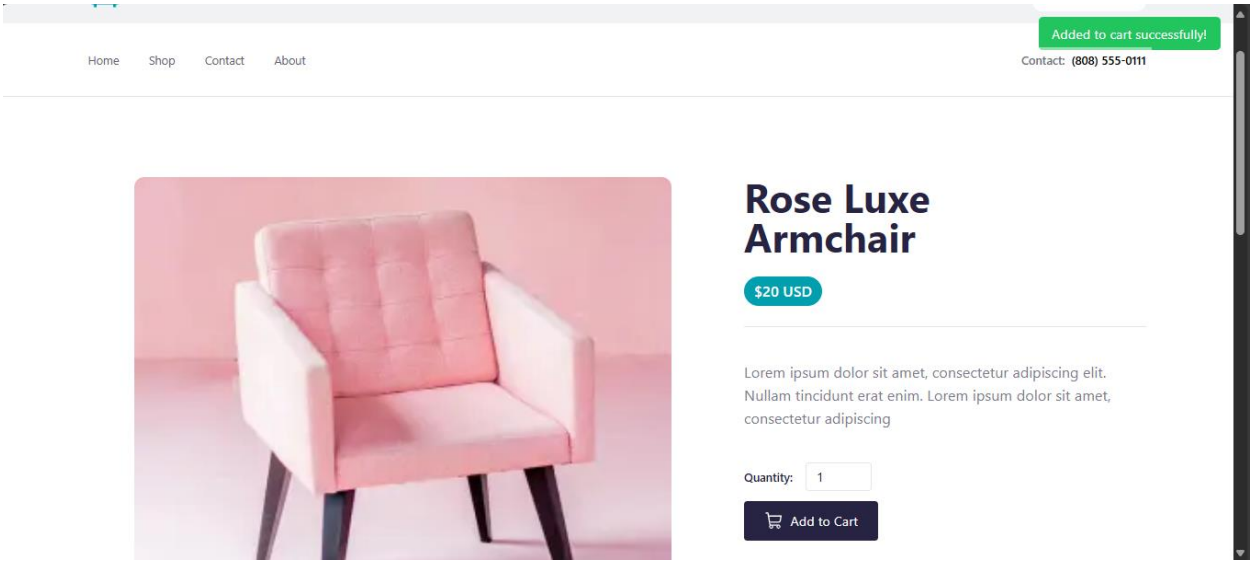
2. Product detail page with social sharing, and related products



3. Cart and checkout pages with dynamic updates.



4. Toast notifications in action.



Steps Taken

1. **Component Creation**:
    - Built reusable components for product cards, filters, pagination, and related products.
    - Used Tailwind CSS for styling to ensure responsiveness.
2. **Dynamic Data Integration**:
    - Integrated components with Sanity CMS to fetch dynamic product and category data using GROQ queries.
    - Ensured proper data mapping and error handling.
3. **State Management**:
    - Utilized Context API to manage cart state and user interactions.
    - Ensured real-time updates for cart, filter functionalities, and related products.
4. **Testing and Optimization**:
    - Tested all components across different devices and screen sizes.
    - Optimized components for performance, including lazy loading for images and debounced inputs.

Challenges Faced

- **Data Mapping Issues**:

    - Resolved mismatches between Sanity CMS schema and API response fields for smooth integration.
- **Search Optimization**:
    - Implemented debounce logic to reduce excessive API calls while maintaining responsive search functionality.
- **State Synchronization**:
    - Ensured proper state synchronization across cart, filters, product listings, and related products.

Decisions Made

- Designed all components to be reusable and modular for scalability.
- Used Context API for lightweight and efficient state management.
- Implemented Tailwind CSS for consistent styling and responsiveness.