Prepared by: Urooj Sadiq

Day 4 - Building Dynamic Frontend Components

Overview:

This document outlines the process of building **dynamic frontend components** for your marketplace application using data fetched from **Sanity CMS** or APIs. It includes practical steps for creating reusable and modular components, managing application state, and implementing responsive design to enhance the user experience.

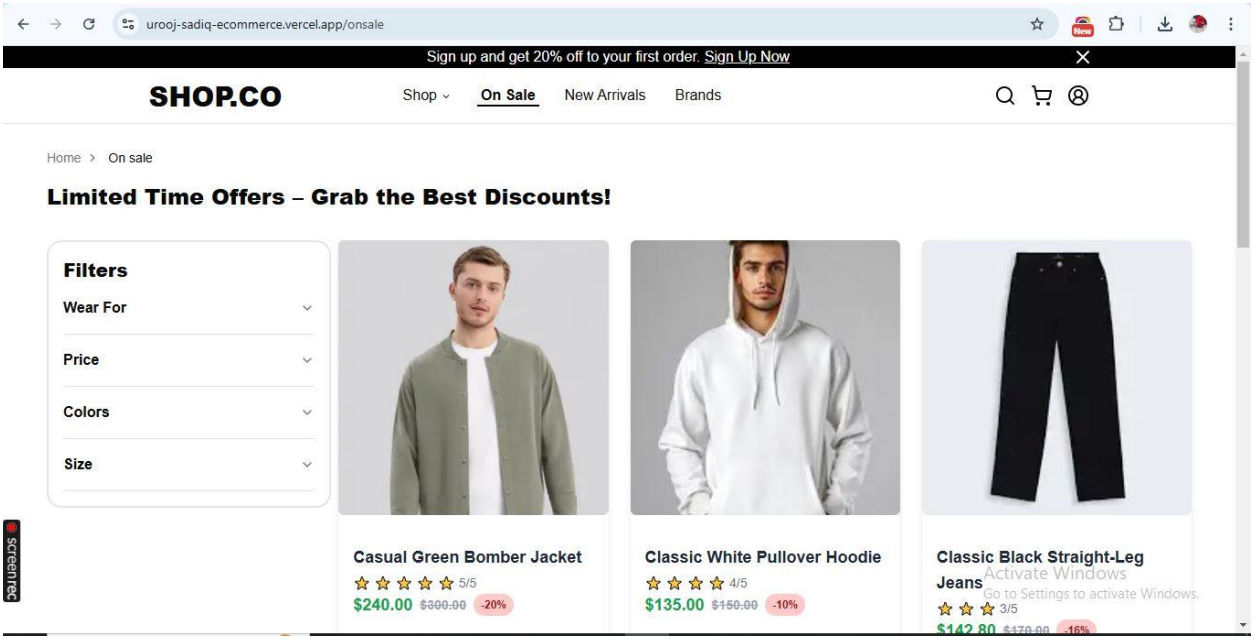
Key Components

1. Product Listing Component

The Product Listing Component dynamically displays product information in an organized grid layout. This component fetches data from Sanity CMS, processes it, and presents the products in a visually appealing manner, enabling users to easily browse through the available items.

Key Fields Included:

- Name
- Price
- Image
- Category
- Discount Percentage
- Stock Status
- New Arrival Badge
- Wear for
- Colors
- Sizes
- Ratings and Reviews

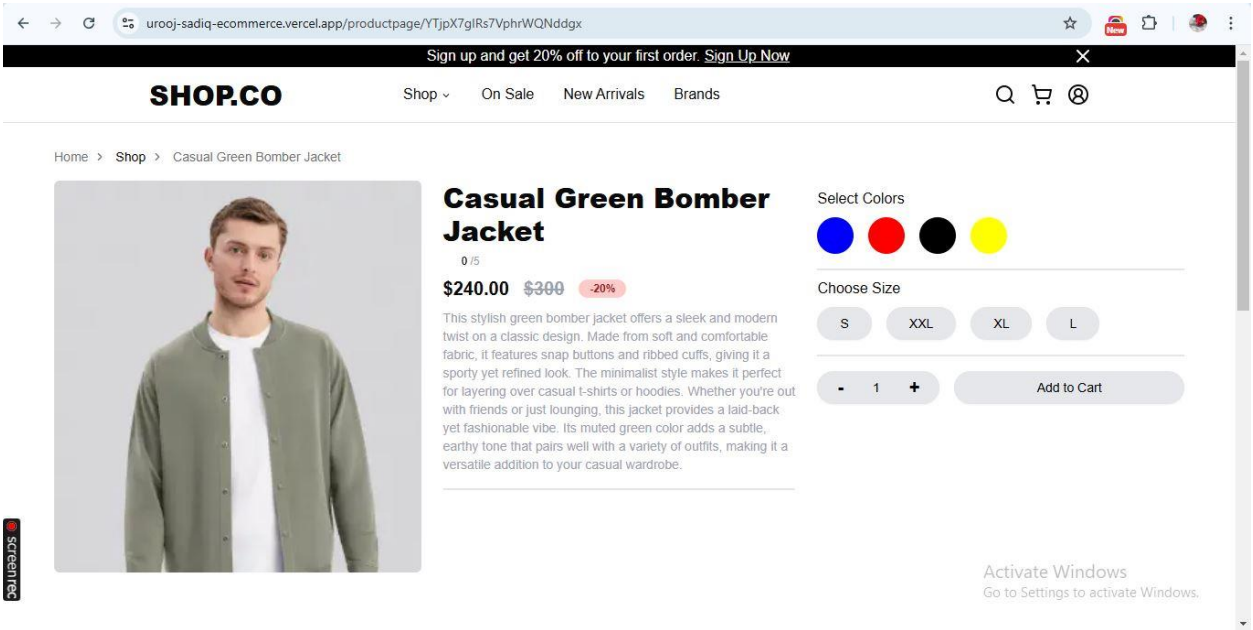


2. Product Detail Component

The Product Detail Component focuses on creating individual product detail pages, providing users with in-depth information about a selected product. This component uses dynamic routing in Next.js to render pages for each product based on its unique identifier.

Comprehensive Information:

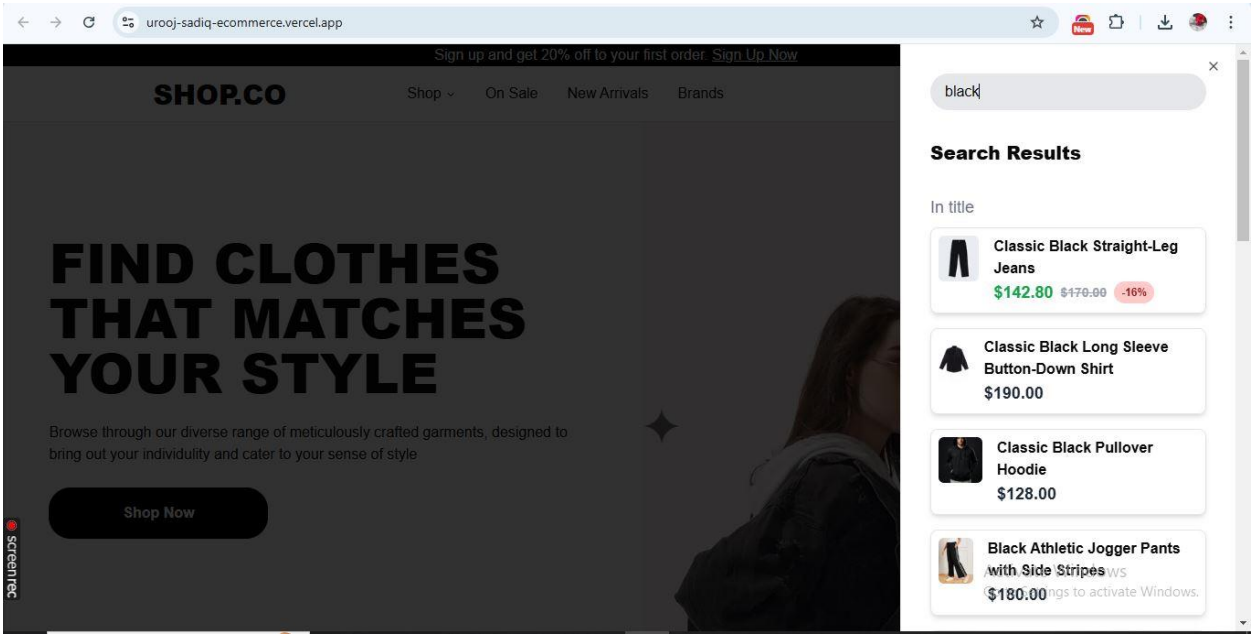
- **Description**: A detailed overview of the product.
- **Price**: Displayed with any active discounts.
- **Available Sizes**: A list of sizes the product is available in.
- **Available Colors**: A display of color options.
- **Image**: Image to showcase the product.



3. Implementing a Search Bar

The Search Bar allows users to filter products dynamically based on specific criteria such as the product's title, color, or size. This feature enhances user experience by providing quick access to desired products.

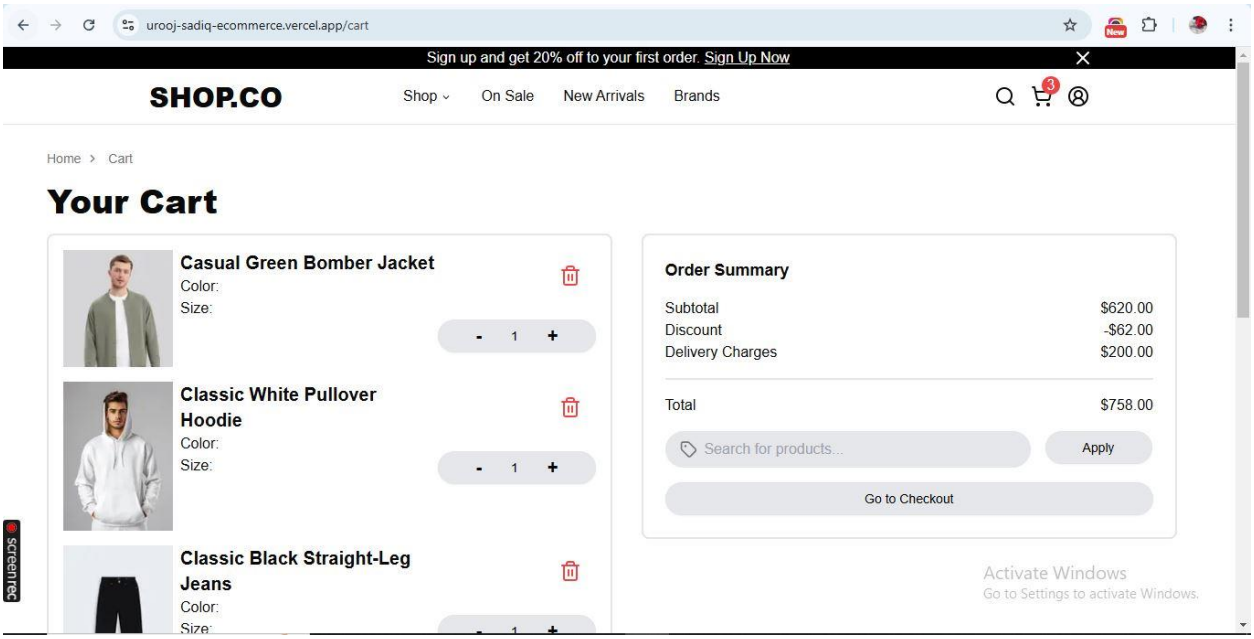
Dynamic Filtering and Live Feedback: Filters products based on user input in real time and displays results immediately as the user types.



4. Cart Component

The Cart Component allows users to manage and view the products they have added to their cart. It tracks the quantity, displays the total price, and provides an intuitive shopping experience.

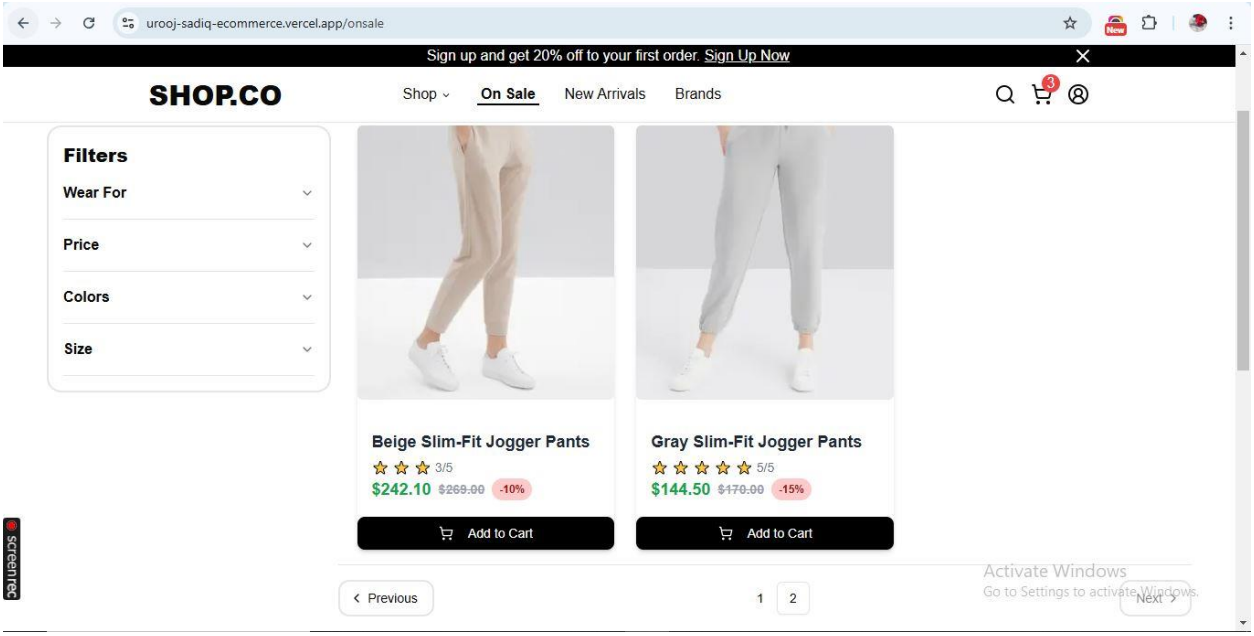
- Displays a list of added items along with their quantities and individual prices.
- Automatically calculates the total cost based on the products and quantities.
- Allows users to change the quantity of each product.
- Users can remove items from the cart.



5. Building the Pagination Component

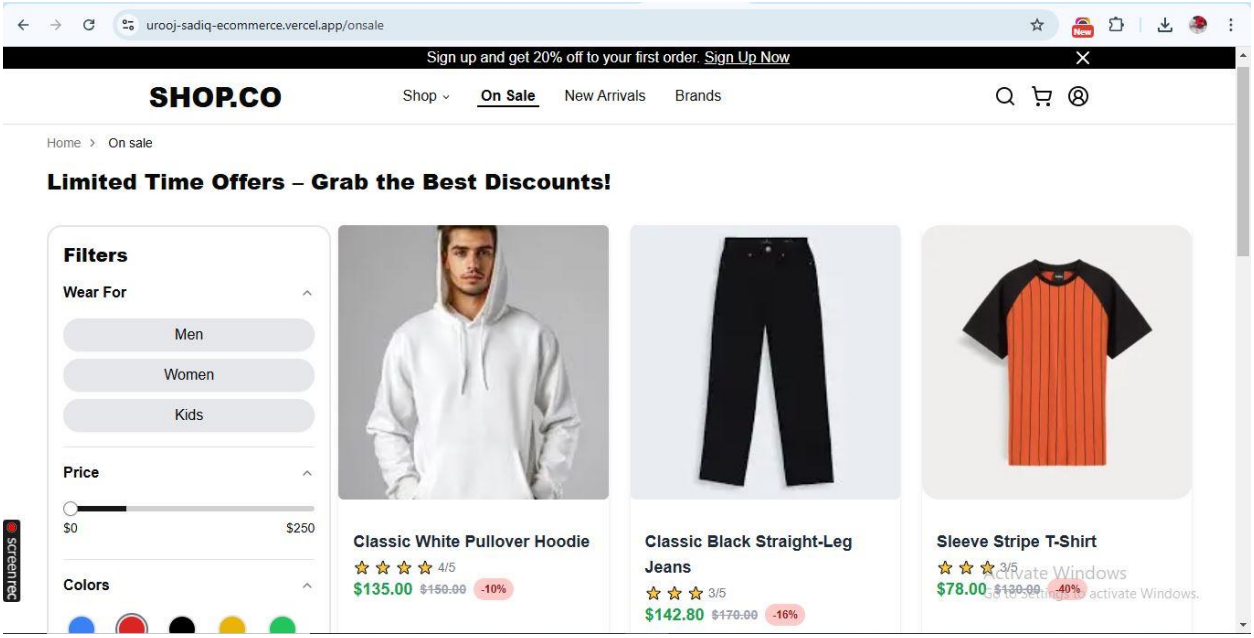
The Pagination Component is essential for e-commerce websites that display large product lists. Instead of loading all products at once, which could slow down the site, pagination breaks the list into manageable chunks. This component allows users to navigate between pages of products and efficiently browse through the inventory.

- Previous and next buttons.
- Users can jump to a specific page by clicking the page numbers.
- Automatically adjusts based on the number of products and the selected page.
- Allows you to set how many items should be displayed per page.



6. Building the Filter Panel Component

The Filter Panel Component is essential for enhancing the user experience on e-commerce websites. It provides users with advanced filtering options i.e. Wear For, Colors, Sizes and Price Range to narrow down product searches according to specific criteria such as price range, brand, and availability.



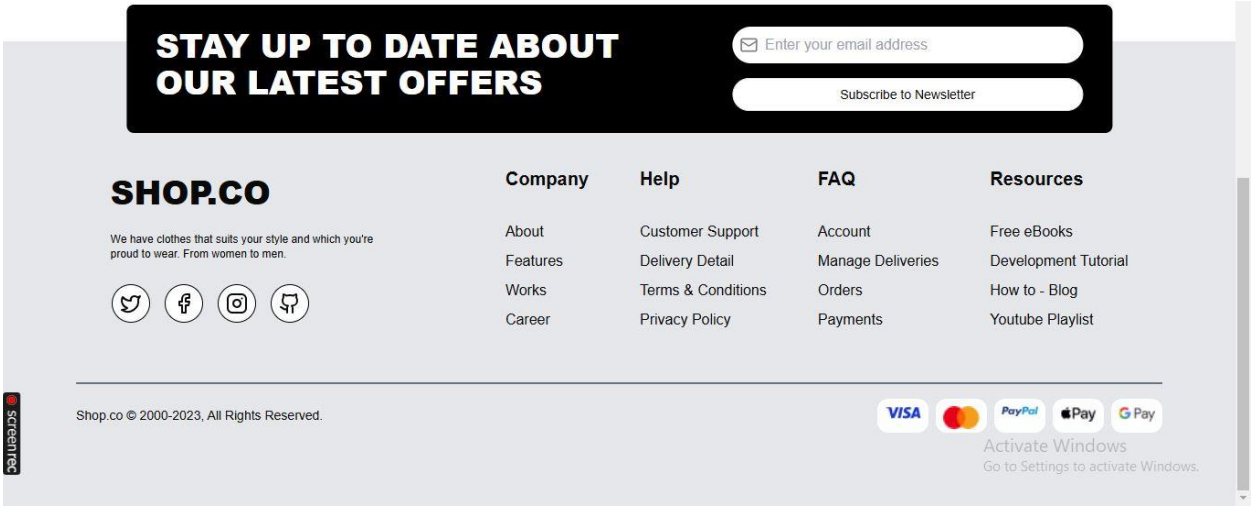
7. Header and Footer Components

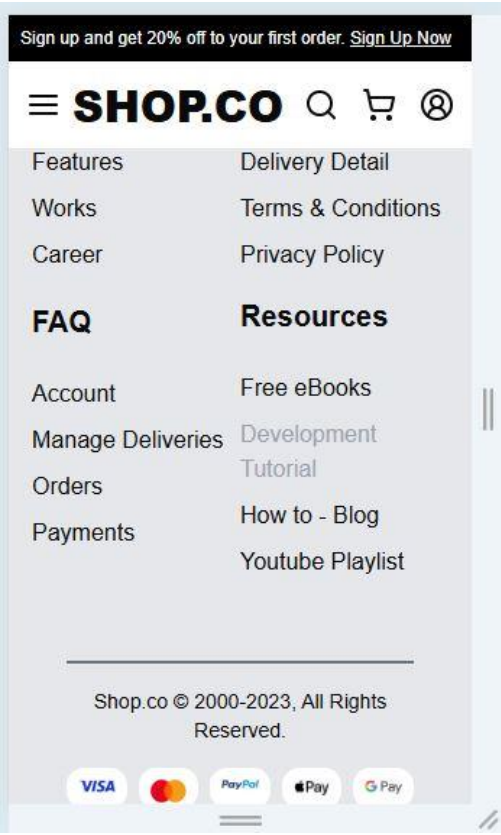
Header:

- Display the brand logo, navigation links (Home, About, Contact), a search bar, and cart icon.
- Keep responsiveness and provides easy access to key pages.

Footer:

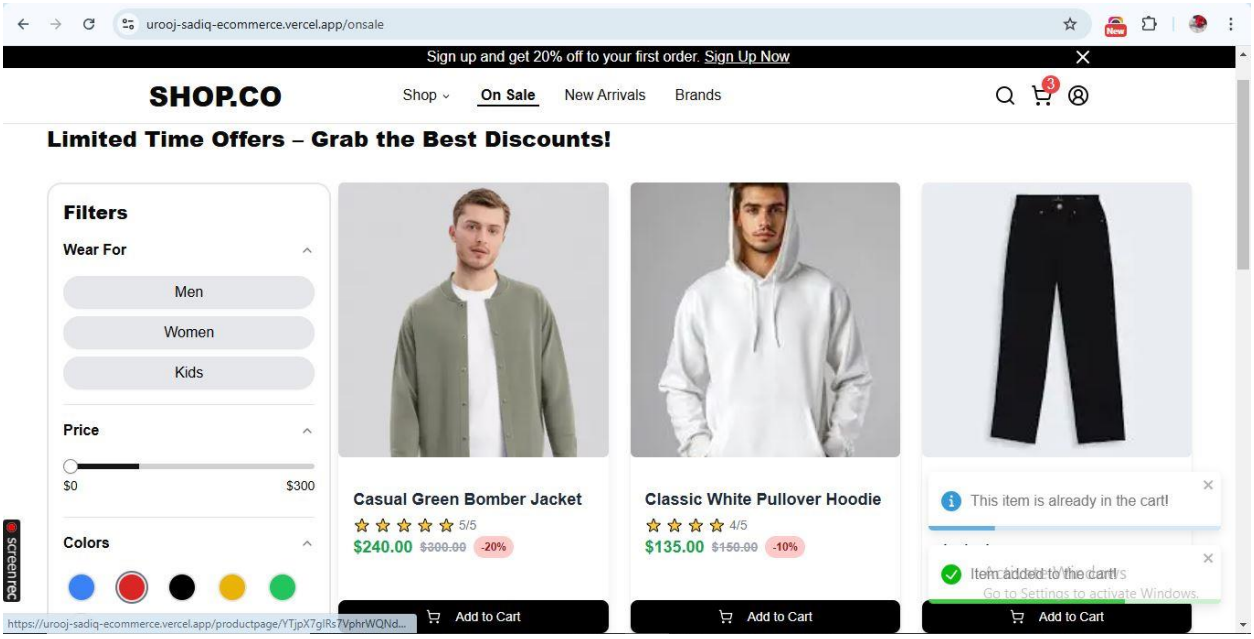
- Provide links to legal pages (Privacy Policy, Terms of Service) and social media accounts.
- Include company contact information.

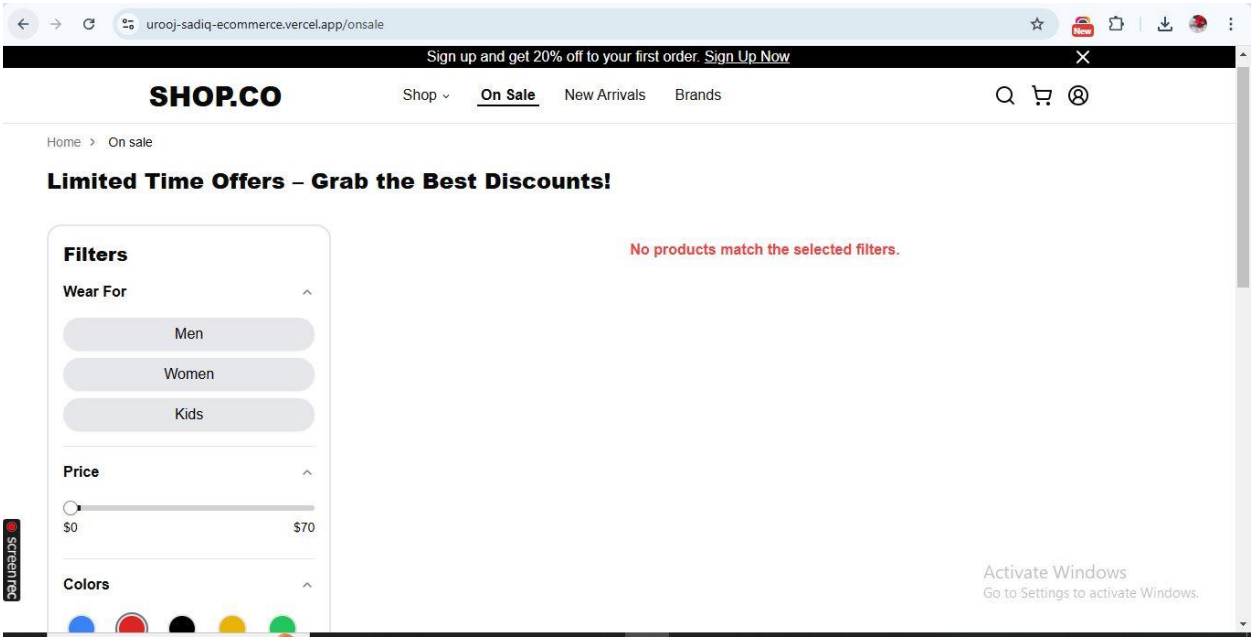




8. Notifications Component

- The Notifications Component is designed to provide real-time feedback to users, alerting them of important actions like adding items to the cart, errors, or successful purchases.
- It leverages toast notifications for brief, non-intrusive messages and modal windows for more detailed alerts. The goal is to deliver immediate feedback that enhances user experience.
- Best practices include auto-closing notifications, using distinct colors for different message types, and ensuring accessibility for all users.





Self-Validation Checklist:

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