Day 4 - Dynamic Frontend Components

Media Mart Functional Deliverables

1. Product Listing Page with Dynamic Data

- o **Description:** The product listing page dynamically displays products using a manually exported JSON dataset. Each product includes its image, name, price, and a brief description, presented in a clean and visually appealing layout to enhance user experience.

2. Individual Product Detail Pages with Accurate Routing and Data Rendering

- o **Description:** Each product detail page dynamically renders detailed information for the selected product, including the title, full description, price, and related images. Leveraging accurate dynamic routing ensures users are directed seamlessly to the correct product page based on their selection.
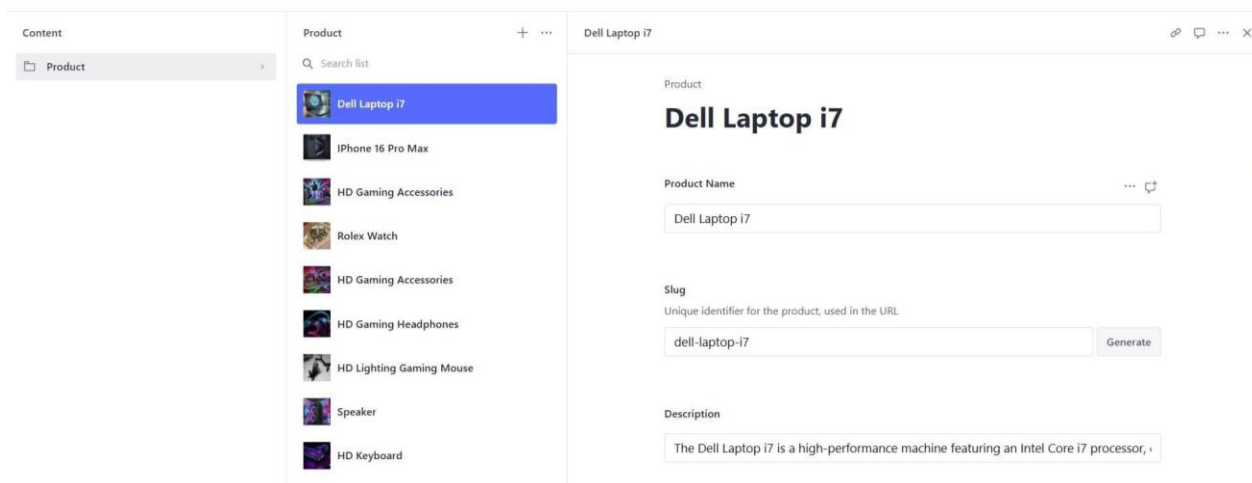
3. Working Category Filters, and Search Bar

- ➤ *Screenshots/Recordings:*
  - o Demonstration of category filters.
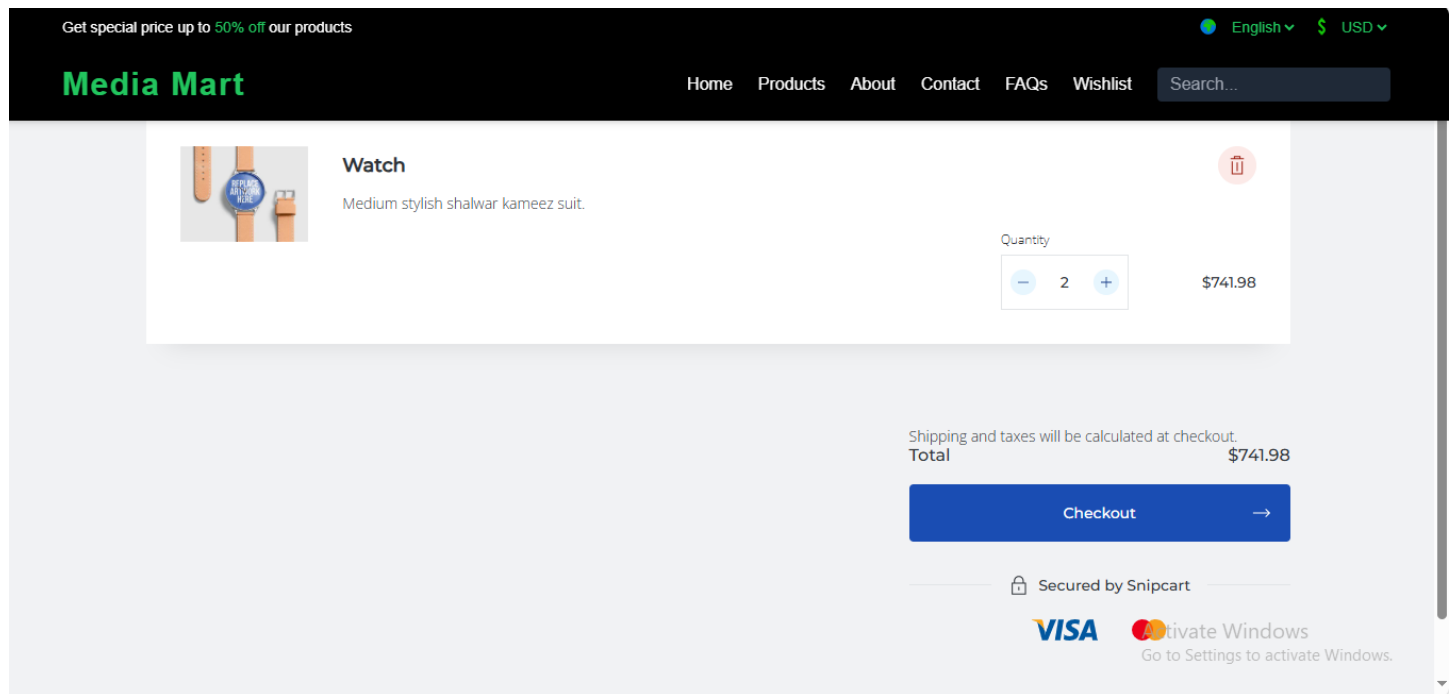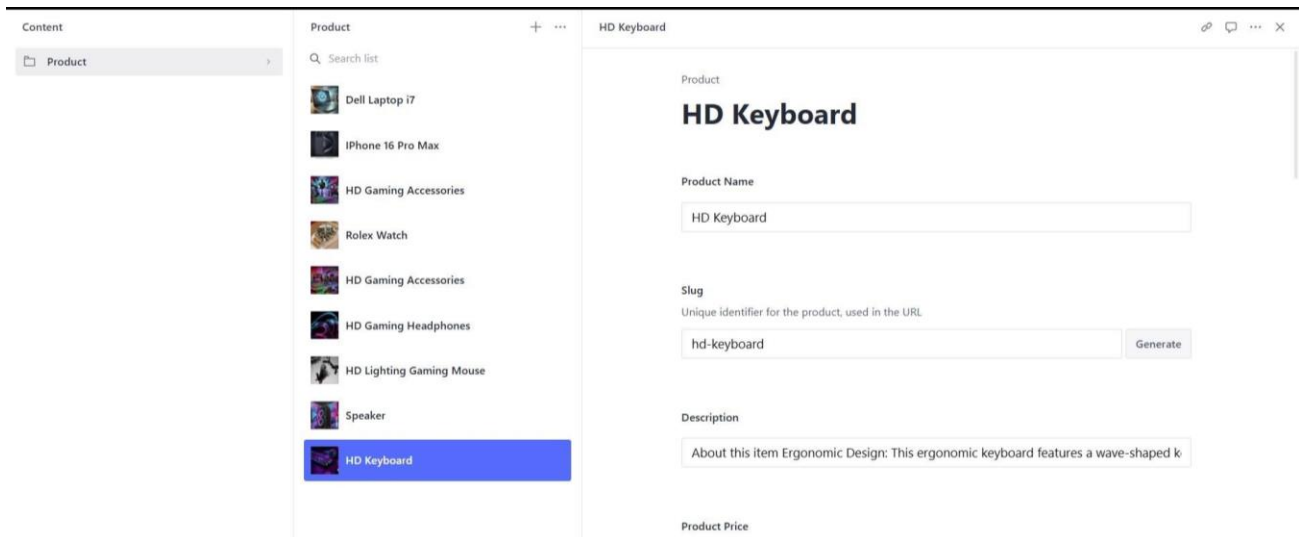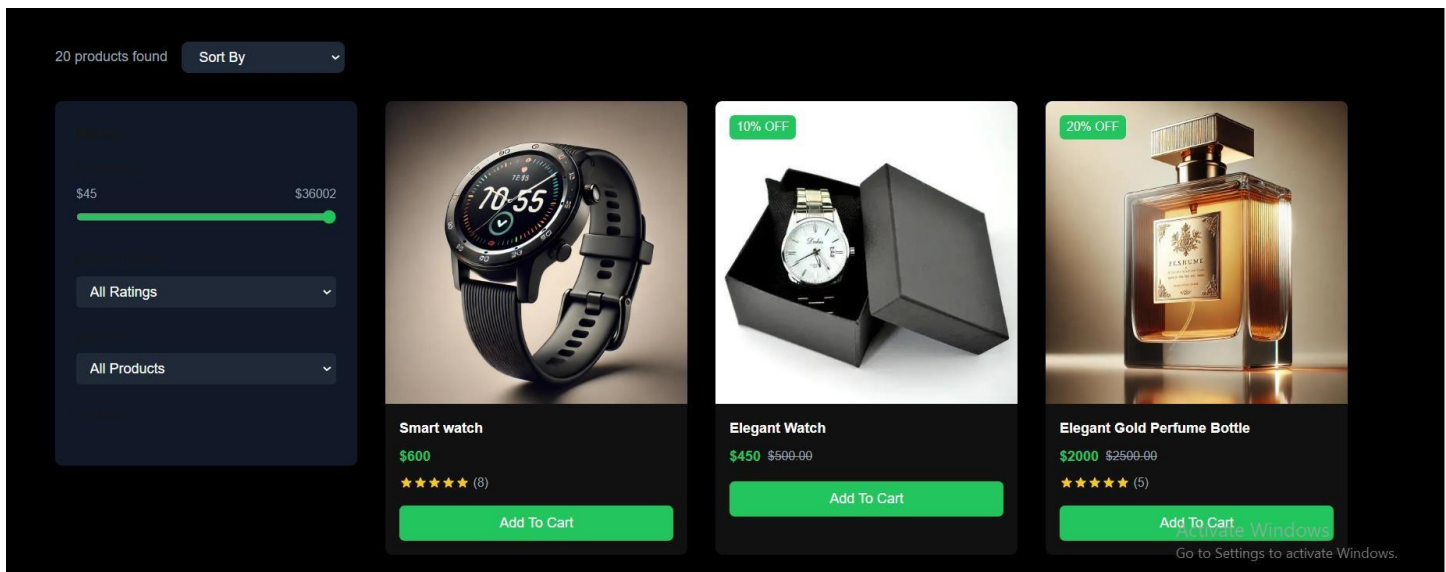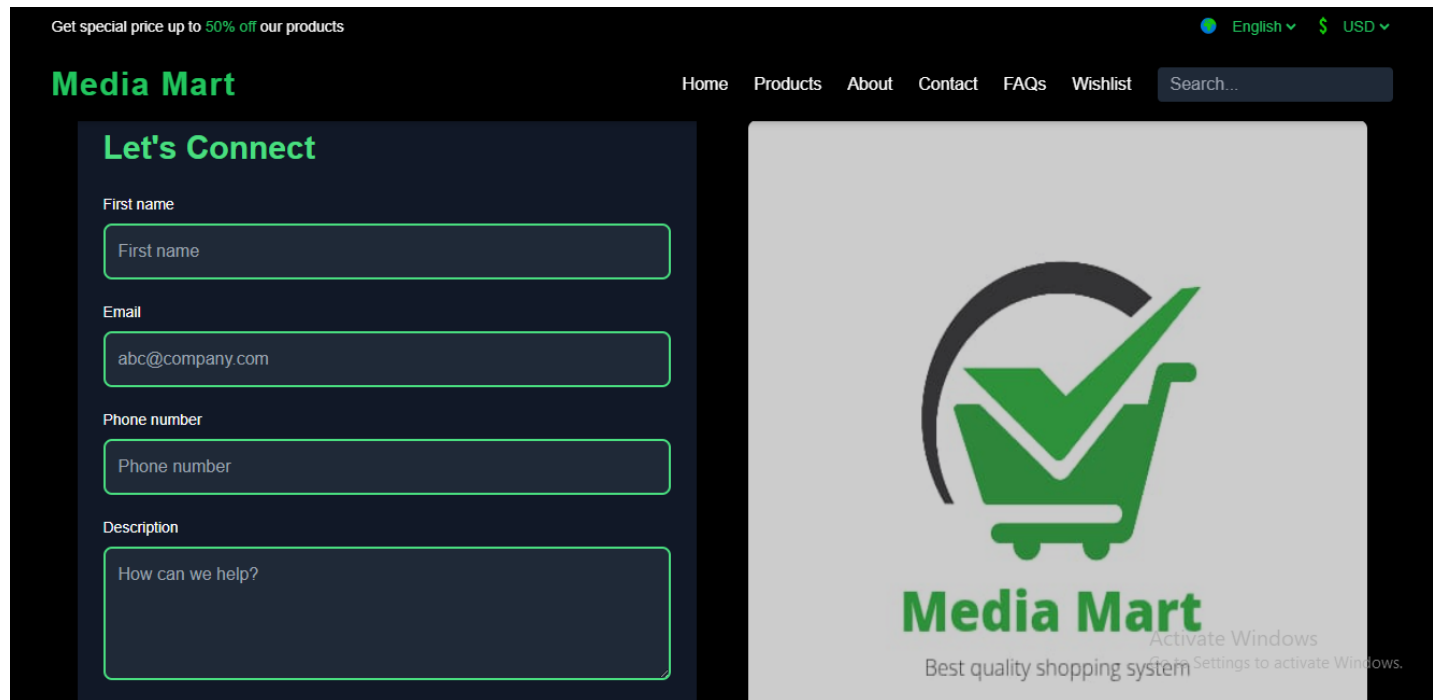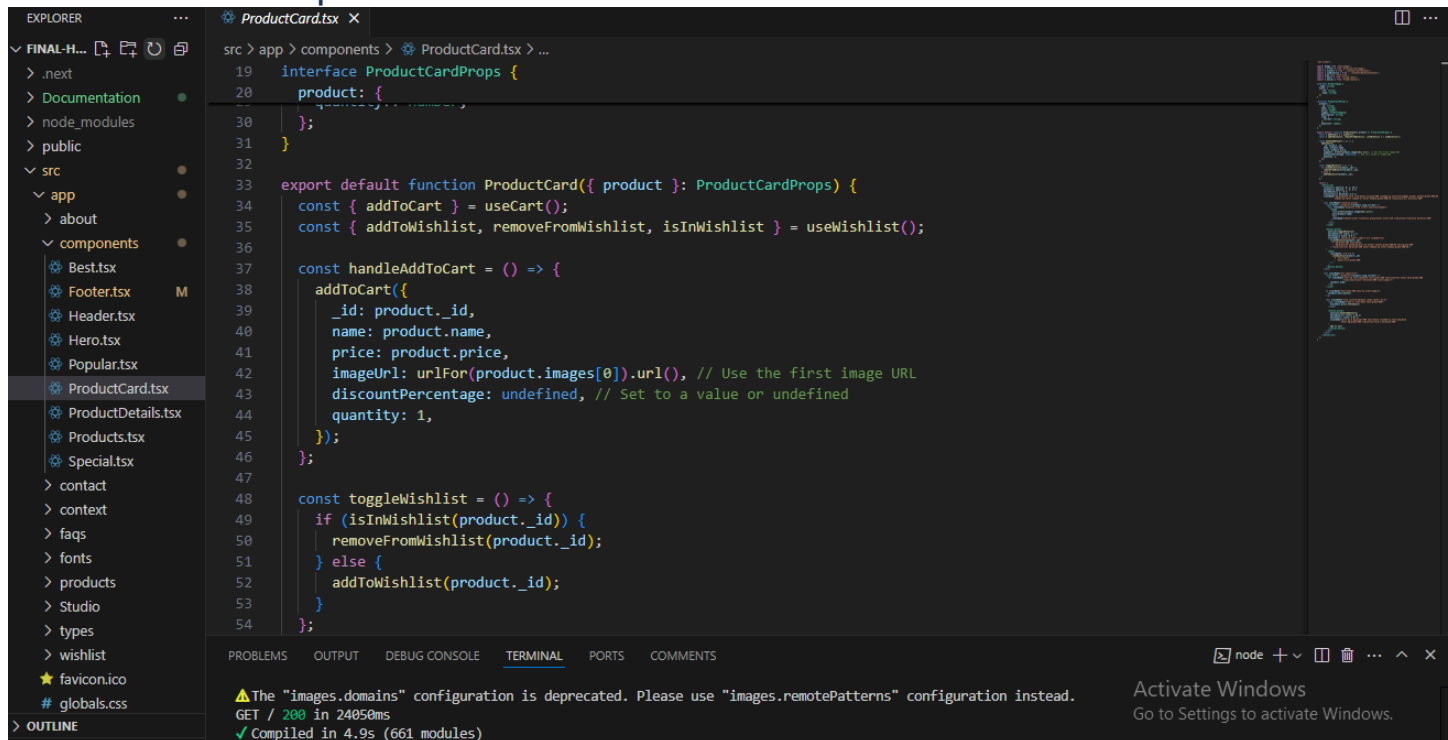 - o Search bar in action.

- **Description:**
  - o Category Filters: Enables filtering of products based on categories like
  - o "Electronics," "Home Appliances," etc., allowing users to narrow down
    their preferences.

  - o Search Bar: Offers real-time search functionality, dynamically
  - o filtering products as users type keywords.

1. Product Listing Page with Dynamic Data

- **Description:** The product listing page dynamically displays products using a manually exported JSON dataset. Each product includes its image, name, price, and a brief description, presented in an aesthetically pleasing format.

2. Individual Product Add to cart

- **Description:** Each product detail page dynamically renders data for the selected product, including the title, full description, price, and related images. Accurate dynamic routing ensures users are directed to the correct product page based on their selection.

3. Working Category Filters, Search Bar, and Pagination

- **Screenshots/Recordings:**
  - Demonstration of category filters.
  - Search bar in action.
- **Description:**
  - **Category Filters:** Filters products based on categories like "Electronics," "Home Appliances," etc.
  - **Search Bar:** Provides real-time search functionality, filtering products dynamically as users type.

Code Deliverables

Key Components

Product Card Component

Review Component

```
src > app > components > ProductDetails.tsx > ...
20    const ProductDetails: React.FC<Props> = ({ product }) => {
22
23      const handleAddToCart = () => {
24        addToCart({
25          _id: product._id,
26          name: product.name,
27          price: product.price,
28          imageUrl: urlFor(product.image[0]).url(),
29          quantity: 1,
30        });
31      };
32
33      return (
34        <div className="container mx-auto px-4 py-8 min-h-screen">
35          <div className="grid grid-cols-1 md:grid-cols-2 gap-8">
36            <motion.div initial={{ opacity: 0, y: 20 }} animate={{ opacity: 1, y: 0 }}>
37              <Image
38                src={urlFor(product.image[0]).url()}
39                alt={product.name}
40                width={500}
41                height={500}
42                className="rounded-lg"
43              />
44            </motion.div>
45            <div>
46              <h1 className="text-4xl font-bold text-green-500">{product.name}</h1>
47              <p className="text-gray-300 mt-4">{product.description}</p>
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    COMMENTS

⚠The "images.domains" configuration is deprecated. Please use "images.remotePatterns" configuration instead.
GET / 200 in 24050ms
✓ Compiled in 4.9s (661 modules)
```
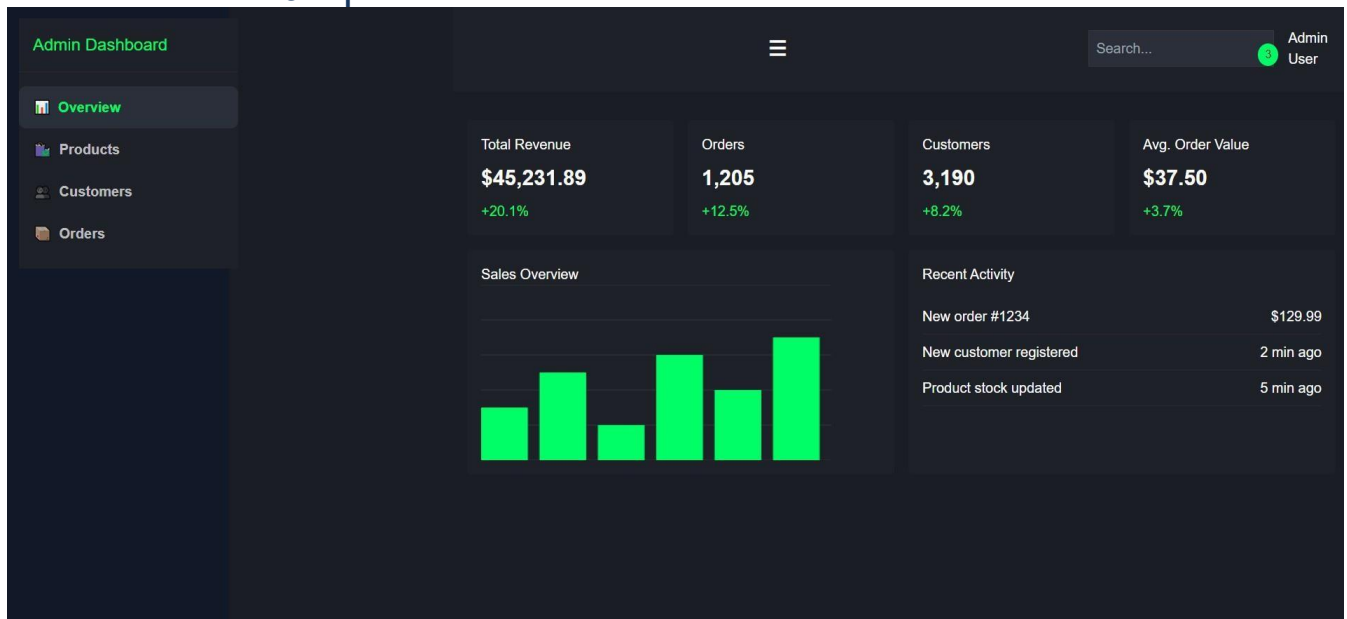
Whish List Component

```
src > app > wishlist > page.tsx > ...
32    export default function WishlistPage() {
94              </h2>
95              <p className="□text-gray-400 mb-6">
96                Add some awesome products to your wishlist!
97              </p>
98              <Link
99                href="/products"
100               className="inline-block ■bg-green-500 □text-white px-6 py-3 rounded-lg font-semibold
101                           ■hover:bg-green-600 transition-transform duration-300 shadow-lg □hover:shadow-green-500/20 transfo
102               >
103                 Explore products
104               </Link>
105             </div>
106         ) : (
107           <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-8">
108             {wishlistProducts.map((product) => (
109               <motion.div
110                 key={product._id}
111                 initial={{ opacity: 0, y: 20 }}
112                 animate={{ opacity: 1, y: 0 }}
113                 exit={{ opacity: 0, y: 20 }}
114                 className="bg-gradient-to-b □from-black □to-gray-800 rounded-lg overflow-hidden
115                            border □border-green-500/20 shadow-lg hover:shadow-2xl □hover:shadow-green-500/20
116                            transition-all duration-300 transform hover:scale-105"
117               >
118                 <Link href={`/products/${product.slug}`}>
119                   <div className="relative h-64 w-full group">
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    COMMENTS

⚠The "images.domains" configuration is deprecated. Please use "images.remotePatterns" configuration instead.
GET / 200 in 24050ms
✓ Compiled in 4.9s (661 modules)
```

.

Admin Dashboard Component

Login Page

Documentation:

Steps Taken to Build and Integrate Components

1. **Data Preparation:**
   - Created a JSON dataset with fields such as id, name, price, image, and category.

2. **Component Development:**
    - Built reusable components, including Product Card, Product List, and Search Bar.
    - Developed additional components like cart, add-to-cart, and user profile sections.

3. **Dynamic Routing:**
    - Implemented Next.js dynamic routing for product detail pages.

4. **UI/UX Enhancements:**
    - Added category filters, search functionality, and pagination.
    - Styled components using Tailwind CSS for responsiveness and consistency.

Challenges Faced and Solutions Implemented

1. **Efficiently Rendering Large Datasets:**
    - **Solution:** Implemented pagination to improve performance and load times.

2. **Maintaining Consistent Routing and Data Handling:**
    - **Solution:** Utilized Next.js dynamic routing and centralized data in a JSON file.

3. **Creating Responsive and User-Friendly Components:**

- **Solution:** Applied Tailwind CSS for styling and ensured a responsive design for various devices.

Best Practices Followed

- Adopted a modular component structure for code reusability.
- Used TypeScript for enhanced type safety and error prevention.
- Ensured responsive designs for optimal user experience on both mobile and desktop.

Components Developed for "Media Mart"

- Accordion
- Product Card
- Product List
- Search Bar
- Image Gallery
- Add-to-Cart
- Wish List
- Pagination
- Header
- Footer
- Admin Dashboard
- Login
- Payment Method(Snip Cart)

Conclusion

The **Media Mart** e-commerce platform demonstrates the power of dynamic frontend components in creating an engaging and functional user experience. By utilizing reusable components, efficient data handling, and responsive design principles, the platform successfully combines usability and performance. Key features such as dynamic routing, category filters, search functionality, and pagination ensure smooth navigation and interaction for users. The challenges encountered during development were effectively addressed through best practices, making the platform robust and scalable. **Media Mart** is a testament to the effectiveness of modern front-end technologies and thoughtful design in crafting user-centric digital solutions.

Best Regards

Web & Media Agency teams

.