Day 4 - Building Dynamic Frontend Components for Your Marketplace

Prepared by: Syed Huzaifa Ahmed Hashmi

1. Day 3 Recap

On Day 3, the focus was on API integration and data migration. Key achievements include:

- Reviewing and testing the provided API endpoints (primarily `/products`).
- Validating that the Sanity CMS schema aligns with the API data structure.
- Drafting and executing a migration script using `npm run migrate` to import product data.
- Implementing utility functions in Next.js to fetch data from Sanity.
- Displaying product data on the frontend (in basic components), though with some error handling and refinements still pending.

2. Day 4 - Building Dynamic Frontend Components for Your Marketplace

Objective

On Day 4, the goal is to design and develop dynamic, reusable, and responsive frontend components to display marketplace data fetched from Sanity CMS or APIs. This phase emphasizes:

- Modular component design.
- Effective state management.
- Responsive UI/UX best practices.
- Preparation for real-world scalability and professional project delivery.

Key Learning Outcomes

1. Building dynamic frontend components that render real-time data.
2. Creating reusable and modular UI elements.

3. Applying state management techniques for component interaction.
4. Designing responsive interfaces that work seamlessly on all devices.
5. Replicating professional workflows and preparing documentation for submission.

---

3. Key Components to Build

Product Listing & Detail

1. **Product Listing Component**:
    - **Functionality**: Dynamically renders products in a grid layout with details such as product name, price, image, and stock status.
    - **Status**: Implemented.
2. **Product Detail Component**:
    - **Functionality**: Displays detailed information on individual product pages using dynamic routing (e.g., `/products/[id]`).
    - **Status**: Implemented.

Category and Search

3. **Category Component**:
    - **Functionality**: Dynamically fetches and displays product categories; allows filtering by category.
    - **Status**: Not yet implemented.
4. **Search Bar**:
    - **Functionality**: Enables search functionality to filter products by name or tags.
    - **Status**: Not yet implemented.

Cart & Wishlist

5. **Cart Component**:
    - **Functionality**: Displays items added to the cart, their quantities, and total price; uses state management for dynamic updates.
    - **Status**: Implemented.
6. **Wishlist Component**:
    - **Functionality**: Allows users to save products for future reference; can use local storage or a global state solution.
    - **Status**: Not yet implemented.

Checkout and User

7. **Checkout Flow Component**:
    - **Functionality**: Implements a multi-step form for checkout, including billing, shipping, and mock payment details.
    - **Status**: Implemented.
8. **User Profile Component**:
    - **Functionality**: Displays user-specific details such as name, email, addresses, and order history.
    - **Status**: Not yet implemented.

Additional Functional Components

9. **Reviews and Ratings Component**:
    - **Functionality**: Allows users to view and submit product reviews and ratings.
    - **Status**: Not yet implemented.
10. **Pagination Component**:
    - **Functionality**: Breaks down large product lists into pages, with navigation controls.
    - **Status**: Not yet implemented.
11. **Filter Panel Component**:
    - **Functionality**: Provides advanced filtering options (price range, brand selection, availability toggles) for product listings.
    - **Status**: Not yet implemented.
12. **Related Products Component**:
    - **Functionality**: Suggests similar or complementary products on the product detail page.
    - **Status**: Not yet implemented.

Core Layout and UX

13. **Header and Footer Components**:
    - **Functionality**: Consistent navigation and branding across the site; includes links to key pages.
    - **Status**: Implemented.
14. **Notifications Component**:

  - **Functionality**: Displays real-time alerts (e.g., toast notifications) for user actions like adding to cart or error messages.
  - **Status**: Implemented using react-hot-toast.

Advanced and Optional Components

15. **Analytics Dashboard Component**:
    - **Functionality**: Displays key performance indicators (KPIs) such as sales, user traffic, and popular products; currently implemented via the Sanity Studio.
    - **Status**: Implemented.
16. **Product Comparison Component**:
    - **Functionality**: Allows users to compare multiple products side by side.
    - **Status**: Not yet implemented.
17. **Multi-Language Support Component**:
    - **Functionality**: Provides a language switcher for multilingual support; UI placeholder at this stage.
    - **Status**: Not yet implemented.
18. **Order Tracking Component**:
    - **Functionality**: Displays real-time order status updates, including estimated delivery times.
    - **Status**: Not yet implemented.
19. **FAQ and Help Center Component**:
    - **Functionality**: Contains a searchable FAQ and help center, along with contact options.
    - **Status**: Implemented.
20. **Subscription Management Component**:
    - **Functionality**: Manages user subscriptions and renewal dates; currently a UI component.
    - **Status**: Not yet implemented.
21. **Admin Dashboard Component**:
    - **Functionality**: Provides an interface for managing products, orders, users, and analytics; built using Sanity.
    - **Status**: Implemented.
22. **Discount and Promotion Component**:
    - **Functionality**: Enables the creation and management of discounts, coupons, and promotions; displayed dynamically on product listings.

    - **Status**: Not yet implemented.
23. **Social Media Sharing Component**:
    - **Functionality**: Allows users to share products or promotions on social media platforms; UI placeholder for now.
    - **Status**: Not yet implemented.
24. **Bulk Upload Component**:
    - **Functionality**: Enables admin users to upload large datasets (CSV/Excel) of product information; includes data validation.
    - **Status**: Not yet implemented.
25. **AI Recommendations Component**:
    - **Functionality**: Uses basic AI/LLM models to recommend products based on user preferences; currently a UI-only component.
    - **Status**: Not yet implemented.
26. **Gift Card and Voucher Component**:
    - **Functionality**: Allows the purchase and redemption of gift cards or vouchers, with balance tracking.
    - **Status**: Not yet implemented.
27. **Customer Feedback Component**:
    - **Functionality**: Provides a form for users to submit feedback about the marketplace or specific products, with aggregated display for review.
    - **Status**: Not yet implemented.
28. **Advanced Search Component**:
    - **Functionality**: Implements faceted search with multiple filters (price, ratings, brands) for precise results.
    - **Status**: Not yet implemented.

---

6. Expected Output

By the end of Day 4, the following should be achieved:

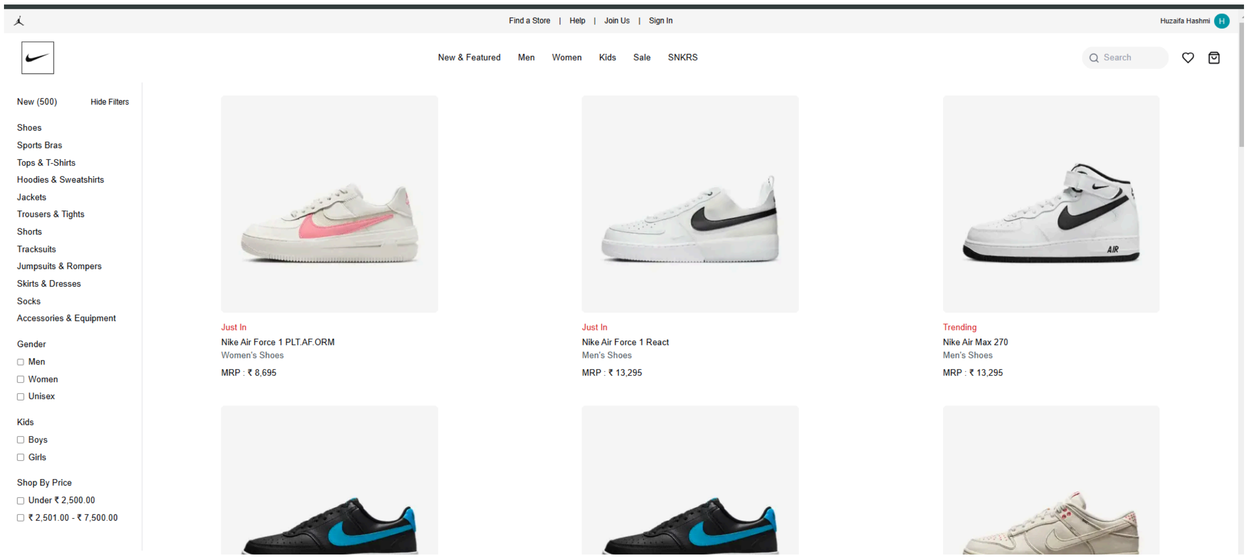
- A fully functional product listing page that dynamically renders product data.
- Individual product detail pages with dynamic routing and accurate data rendering.
- Working category filters, a search bar, and pagination (or UI placeholders for these components).
- Implemented cart and checkout flow components.
- Consistent header, footer, and notifications ensuring a unified user experience.

- Additional components (e.g., FAQ, analytics dashboard) either fully implemented or present as UI placeholders.
- All components are designed to be reusable, responsive, and modular.

---

7. Submission Requirements

- **Document Title**: "Day 4 - Dynamic Frontend Components - Fitness Marketplace"
- **What to Submit**:
    1. **Functional Deliverables**:
        - Screenshots or screen recordings showcasing:
            - The product listing page with dynamic data.



Just In
Nike Air Force 1 PLT.AF.ORM
Women's Shoes
MRP : ₹ 8,695

Just In
Nike Air Force 1 React
Men's Shoes
MRP : ₹ 13,295

Trending
Nike Air Max 270
Men's Shoes
MRP : ₹ 13,295

            - Individual product detail pages with accurate routing and data rendering.
            - Working category filters, search bar, and pagination (if implemented).
            - Other additional components were developed.
    2. **Code Deliverables**:
        - Code snippets for key components (e.g., ProductCard, ProductList, SearchBar).
        - Scripts or logic used for API integration and dynamic routing.
    3. **Documentation**:
        - A technical report summarizing the implementation process, challenges faced, and solutions implemented.
        - Include screenshots, code snippets, and a summary of best practices followed.
    4. **Repository Submission**:

        - Upload all files to your designated GitHub repository, organized under a clear folder hierarchy.

---

9. Checklist for Day 4

Self-Validation Checklist:

- **Frontend Component Development**: ✔ / ✘
- **Styling and Responsiveness**: ✔ / ✘
- **Code Quality**: ✔ / ✘
- **Documentation and Submission**: ✔ / ✘
- **Final Review**: ✔ / ✘

---

10. Professional Practices Emphasized

- **Modular and Reusable Component Design**: Ensuring that all components are maintainable and scalable.
- **Effective State Management**: Utilizing React's state and context to manage dynamic data effectively.
- **Responsive and User-Friendly UI Design**: Prioritizing a seamless user experience across all devices.
- **Thorough Documentation**: Maintaining detailed records of code, screenshots, and process logs.
- **Version Control**: Regular commits with clear messages and milestone tagging.

End Note

Day 4 focused on building dynamic frontend components to display marketplace data from Sanity CMS and other APIs. With a comprehensive implementation roadmap and adherence to best practices, the fitness marketplace project is evolving into a professional, scalable solution. Continued enhancements and refinements will ensure a robust user experience and a solid foundation for future development.

Prepared by: Syed Huzaifa Ahmed Hashmi
