DAY 4 - DEVELOPING DYNAMIC FRONTEND COMPONENTS FOR A MARKETPLACE

Hackathon Task Overview

On Day 4, the focus is on creating dynamic frontend components for a marketplace application. Participants need to develop key components that improve user interaction and experience.

1. **Product Listing & Detail Components** – Dynamically display product information.
2. **Category & Filter Panel** – Allow users to browse and filter products easily.
3. **Search Bar** – Enable quick product searches.
4. **Cart, Wishlist & Checkout Flow** – Manage user actions and simplify the purchasing process.
5. **User Profile, Reviews & Pagination** – Handle user data, feedback, and navigation efficiently.
6. **Header, Footer & Notifications** – Structure the layout and provide alerts for user interactions.

This task emphasizes **dynamic rendering, responsiveness, and user-focused design** principles.

Migrating Data to Sanity

Sanity CMS was used to organize and manage marketplace data effectively. The migration process involved transferring product details, categories, and user information into Sanity’s schema. This required defining structured data types (e.g., Products, Blogs) in **Sanity Studio**. Once the schema was set up, data was either imported or manually added, ensuring seamless integration with the Next.js frontend.

Advantages of Using Sanity

- Flexible content modeling
- Real-time updates and collaboration
- Scalable and well-structured data management

Fetching Data from Sanity in Next.js

After migrating the data to **Sanity**, the frontend components in the **Next.js** application retrieved this data using **Sanity's GROQ (Graph-Relational Object Queries)** or the official **@sanity/client** package. Queries were created to dynamically fetch marketplace data, including product listings, categories, and user reviews.

Key data fetching operations included:

- Retrieving a list of products based on their category.
- Fetching product details such as price, description, and images for the **Product Detail Component**.
- Getting filtered or paginated data for the **Search and Filter Panel** to improve navigation.

Integration Steps

1. Set up the **Sanity client** in the Next.js application by configuring the project ID and dataset.
2. Write **GROQ queries** to fetch the necessary data.
3. Implement **getStaticProps** or **getServerSideProps** in Next.js for server-side data fetching, ensuring **SEO optimization** and **dynamic rendering**.

The Product Listing Component is designed to fetch product data from Sanity CMS in real-time, ensuring updates are automatically reflected. Sanity functions as the backend system, storing key details such as the product name, price, description, images, categories, and availability status.

1. **Sanity Data Structure:**
   - Each product in Sanity is defined by fields like title, price, image, stock, and description.
   - Categories and relationships are established for better filtering and organizing of products.
2. **Fetching Products in Next.js:**
   - With Sanity's GROQ queries and the @sanity/client package, the Next.js app retrieves product data.
3. **Dynamic Rendering:**
   - The component displays product information like images, names, and prices dynamically.
   - Pagination or infinite scrolling is incorporated to help navigate large datasets.

Product Detail Page with Dynamic Data

The Product Detail Page dynamically fetches detailed information for individual products from Sanity CMS, ensuring real-time updates of unique attributes like descriptions, images, and specifications for an interactive shopping experience.

Sanity Schema for Product Details:

- Each product has detailed fields such as:
  - Title (Product Name)
  - Price (Product Cost)
  - Description (Overview)
  - Images (Product Photos)
  - Stock (Availability)
  - Colors (Product Color Options)

Dynamic Routing in Next.js:

- Next.js uses dynamic routes (e.g., `/products/[slug]`) to generate individual product detail pages.

Rendering Product Details:

- The page shows:
  - Product Title: Name of the product.
  - Image Gallery: High-quality images for visualization.
  - Price and Add to Cart Button: Interactive buying options.
  - Detailed Description: A comprehensive product overview.
  - Specifications: Additional details such as dimensions and features.

Cart Page Functionality

The Cart Page allows users to manage their selected items for purchase, updating dynamically as items are added or removed in real-time.

1. **Add to Cart:** Selected items are added to the cart with their name, price, and quantity.
2. **Update Quantity:** Users can modify the quantity directly, with prices updating automatically.
3. **Remove Items:** Items can be removed with a click.
4. **Total Price Calculation:** Real-time total cost of items in the cart is displayed.
5. **Proceed to Checkout:** Users can move to the checkout page to complete their purchase.

The cart ensures a smooth, interactive shopping flow by managing product data dynamically.

Checkout Page Functionality

The Checkout Page is where users review their orders and enter necessary details for payment and delivery, ensuring a seamless transaction process.

1. **Order Summary:**
    - Lists all cart items, including product names, quantities, prices, and the total amount.
2. **User Information Form:**
    - Collects delivery details like name, address, phone number, and email.
3. **Payment Options:**
    - Offers secure payment methods such as credit/debit cards and PayPal.
4. **Promo Code Application:**
    - Users can apply discount codes for savings.
5. **Order Confirmation:**
    - After payment, a confirmation message with order ID and delivery estimate is shown.

This page guides users through the final steps of their purchase.

Blog Section with Dynamic Data from Sanity

The Blog Section fetches and displays content such as tutorials, updates, or product articles from Sanity CMS.

1. **Sanity Data Structure:**
    - Blog posts are stored with fields like title, slug, content, author, date, and image.

Dynamic Blog Pages:

- Next.js dynamic routing is used to display individual blog posts on separate pages, allowing users to read full articles.

Customer Reviews Section with Dynamic Data

The Customer Reviews Section allows users to read and submit feedback on products, helping other customers make informed purchasing decisions. This section fetches review data dynamically from Sanity, ensuring it's always up-to-date and easily manageable.

- **Storing Reviews:** Reviews are stored in real time, typically in the review section, appearing at the bottom of the product details page.
- **Displaying Reviews:** Reviews are displayed with the user’s name, rating, and comments, giving a clear idea of their experience. Additionally, a review submission form can be provided for customers to share their feedback after purchasing the product.

By integrating dynamic content from Sanity for both blogs and customer reviews, the website stays current and offers valuable insights to users, enhancing the shopping experience.

Shipment Functionality

The Shipment Section plays a crucial role in the checkout and order fulfillment process, ensuring that products are delivered correctly and promptly. It integrates with a shipping service like ShipEngine, which manages shipping calculations, tracking, and delivery options.

- **Integration with ShipEngine:** ShipEngine is integrated into the system to provide real-time shipping rates, generate labels, and track shipments. It uses customer address data along with product weight and dimensions to calculate the most efficient shipping options. This integration also provides accurate shipping costs based on the customer’s location and delivery method, whether it’s ground, express, or international shipping.
- **Fetching Shipping Rates:** When the user enters their shipping address during checkout, the system sends this data to the ShipEngine API to fetch available shipping methods and costs. The customer can then select their preferred shipping method before proceeding with payment.
- **Shipment Tracking:** After the order is placed and the shipment is processed, customers receive tracking numbers generated by ShipEngine. This allows them to track the progress of their shipments in real-time. A tracking link is provided, offering easy access to monitor delivery status.
- **Shipment Status Updates:** Customers are kept informed about the status of their shipments. They receive notifications when their package is dispatched, in transit, and delivered. Any delays or issues are promptly communicated to ensure customers are always updated on their order’s status.

Tracking and Future Enhancements

As I continue to develop my e-commerce website, I plan to enhance its efficiency by adding more features and components in the future, insha'Allah. Goodbye for now!