Hackathon Day 4: DYNAMIC FRONTEND COMPONENTS

Objective:

On Day 4, the focus is on designing and developing dynamic frontend components to display marketplace data fetched from Sanity CMS or APIs.

This step emphasizes modular, reusable component design and real-world practices for building scalable and responsive web applications.

Product Listing:

The product listing feature dynamically displays data stored in Sanity CMS. By leveraging the flexibility of GROQ queries,

this functionality ensures the product data is fetched efficiently and rendered in real-time on the frontend.

Dynamic Routing:

The dynamic page is designed to display content that adapts based on user interactions, parameters, or external data,

providing a personalized and interactive experience. This enhances the website's usability by delivering relevant

information dynamically without the need for hard-coded static pages.

Dynamic Content:

- This page content changes based on query parameters.
- For example, a car details page dynamically displays information based on the selected car.

Data Fetching:

- Content is fetched from Sanity using API calls.

Fetching Car Data from Sanity using GROQ:

The goal of this implementation is to retrieve car data stored in Sanity CMS using the GROQ query language and display it dynamically on the website.

This approach allows content to be efficiently managed through Sanity's CMS interface while ensuring dynamic rendering on the frontend.

Implementation Overview:

- Initializing State for Data:
  - A React state variable, cards, is created to hold the fetched car data.
  - The setCards function is used to update this state whenever new data is retrieved.
- GROQ Query Language: Used to fetch structured data from Sanity CMS.
- Sanity Client: The client.fetch method executes the GROQ query and returns the result.
- React State and Hooks:
  - useState manages the fetched data.
  - useEffect triggers the data fetching process when necessary.
- Asynchronous Programming: Ensures data fetching occurs without blocking the UI.

Search Car with Category:

The search bar provides users with the ability to quickly find specific items, such as cars, within the application.

It enhances user experience by offering a fast and intuitive way to filter and locate content.

Reusable And Modular Components:

All components are designed to be reusable and modular for better maintainability.

FULLY Responsive:

The components and UI are fully responsive across different screen sizes.

CONCLUSION:

I have successfully developed and integrated key features that improve the application's functionality, scalability, and overall user experience.

Advanced Technical Skills:

- Enhanced my proficiency in modern web development tools and frameworks, including Next.js and Sanity CMS.
- Gained valuable insights into API integration, dynamic routing, and effective state management.

Improved User-Focused Design:

- Prioritized building interactive and responsive components, such as the search bar and comments section, to deliver an excellent user experience.

This project not only expanded my technical expertise but also emphasized the significance of user-focused design and modular development.
It lays a solid foundation for future improvements and demonstrates my capability to create dynamic, scalable, and interactive web applications.

Made by Aliya Manzoor