Hackathon 3 Day 3: API Integration & Migration Report Coffee Q Commerce ☕💻

Overview of Day 3

For Day 3 of Hackathon 3, we were tasked with integrating an API into our Hackathon 2 projects. In Hackathon 2, I worked on Template 5, which was a static project. The objective of Day 3 was to transform the static project into a dynamic one by incorporating the API, adjusting the schema, and migrating the data to Sanity CMS. This challenge involved integrating the API into the project, updating the product schema, and migrating product data to Sanity CMS.

Accomplishments

1. API Integration

Integrated the external API (`https://template6-six.vercel.app/api/products`) provided by the instructor into the project.
The API fetches product-related data such as product name, description, price, image URL, tags, and discount percentage.
Used the fetched data to dynamically populate product listings on the frontend, ensuring that the products are fetched and displayed from the API rather than being statically hardcoded.

2. Schema Adjustments

Updated the existing Sanity product schema to align with the API data structure.
Added the following fields to the Sanity schema to match the API data:

- `title`: The product name.
- `description`: The product's detailed description.
- `price`: The product's price.
- `tags`: Tags associated with the product for categorization.

- `productImage`: The image of the product.

These fields enable Sanity CMS to sync data properly with the API, ensuring seamless data transfer from the API to the CMS.

3. Data Migration

Wrote a custom Node.js migration script to fetch product data from the API and populate the Sanity CMS database.
The script automates the process of fetching product details, including images, and uploading them into Sanity.
Successfully mapped the image URLs from the API to Sanity's `productImage` field, automating data migration and ensuring consistency across the platform.
The product data is now dynamically displayed on the frontend, replacing the static data from Hackathon 2.

4. Frontend Display

The dynamically fetched product data is rendered on the frontend using React components.
The following sections were updated to display the API data dynamically:

- Featured Products
- Shop Products

Each section displays product images, prices, tags, and discount information fetched from the API.
The UI components were structured to ensure clean and responsive data display, making the user experience seamless.

5. Key Learnings

API Integration: Gained hands-on experience in integrating an external API into a Next.js project and rendering dynamic data in the frontend.
Schema Design: Enhanced skills in adjusting the content management schema in Sanity to align with external API data structures.

Data Migration: Learned how to write a Node.js script for automating the migration of product data into Sanity CMS, which saves time and improves efficiency.
Frontend Data Rendering: Improved understanding of React components and how to dynamically render data fetched from APIs on the frontend, building more interactive web pages.

Conclusion

Day 3 of Hackathon 3 marked a significant achievement in transitioning the e-commerce project from a static site to a dynamic, API-driven platform. The successful integration of the API and migration of data to Sanity CMS has set the stage for a fully functional, dynamic product display system. This accomplishment will serve as the foundation for further development in the coming days of the hackathon.

I am excited to continue building and refining the project, optimizing both functionality and user experience as we progress.