By Tanzeela Arshad Roll No: 00431067

API INTEGRATION AND DATA MIGRATION

Objective:

On Day 3 of the hackathon, my focus was to integrate APIs and perform data migration which was provided according to my car rental website. This step was important to ensure that the website displays real-time data dynamically while maintaining a well-structured backend. I successfully completed the tasks and displayed the data on the website as per my template.

Tasks Completed

1. API Integration

I integrated APIs into my car rental website to fetch dynamic data and enhance functionality. Below are the details:

APIs Used:

Rental car listings API

Steps Taken:

Integrated the APIs into the front-end code using the fetch() method.

Displayed fetched data dynamically on the website, ensuring it matched the design and layout of my **template.**

2. Sanity Schema Development and Integration

The hackathon organizers provided pre-defined Sanity schemas and a migration script. Here’s how I implemented them:

Steps Taken:

Integrated Sanity into the project by generating an API token from Sanity.

Defined the given Sanity schema by importing it into the project’s index.ts file.

Created a folder named scripts to organize the migration process. Inside this folder, I added a file called importTemplate7Data.mjs and copied the provided migration script into it.

Installed the required dependencies using the following command:

```
npm install @sanity/client axios dotenv
```

Used the migration script with Axios to import data into Sanity. This process successfully imported all the car rental data into the Sanity.

Outcome:

After integration, the Sanity CMS was seamlessly connected to my project, and the data from the CMS was available in the website.

3. Data Fetching and Display

Once the data was successfully imported into Sanity, I fetched and displayed it on the website:

Steps Taken:

Fetched data dynamically from the Sanity CMS using queries.

Integrated the fetched data into the front-end, ensuring it matched the layout and design of my car rental website.

Verified that the displayed data was accurate and aligned with the imported content.

Outcome:

The website now dynamically displays car rental listings and other related information fetched from Sanity CMS.

Testing and Verification

After completing API integration and data migration, I tested the functionality and reliability of the system:

Tests Conducted:

Verified the car listings displayed on the website matched API and CMS data.

Checked the performance of data fetch operations on various devices and screen sizes.

Results:

All tests were successful, and the website now dynamically displays data fetched from APIs and the Sanity CMS.

Conclusion:

Day 3 was a significant milestone in the hackathon journey. By completing API integration and data migration, I ensured that our rental website is functional, dynamic, and user-friendly. These developments have laid a strong foundation for the upcoming features and refinements I plan to implement.