Technical Prototype

Introduction:

This prototype was made to test if the recommended features were technically possible to implement, to see if they are scalable and maintainable, and if it is possible to do so with Next.js as well.

Scope and Objectives:

The scope of the technical prototype includes implementing essential features and integrating Contentful CMS. The objective is to gather feedback on the implemented features.

Methodology:

The prototype was developed after reviewing the Contentful CMS documentation and conducting a MoSCoW analysis to prioritize features.

MoSCow analysis: link

https://www.contentful.com/developers/docs/

Integration with CMS:

Contentful CMS was integrated into the project by leveraging its APIs to fetch and manage content. The integration process involved setting up Contentful spaces, defining content models, and retrieving content using Next.js server-side rendering.

Learning Process:

Throughout the development of the technical prototype, significant emphasis was placed on learning. The most important thing that had to be learned to work on

this project was Next.js server-side rendering and how contentful can be integrated to Next.js.

Next.js server-side rendering:

SSR renders the React components on the server and sends the fully-rendered HTML to the client. The client receives a ready-to-display page.

Benefits of SSR:

- Improved SEO: Search engines can crawl and index SSR pages more effectively since they receive fully-rendered HTML content.
- Faster Initial Page Load: SSR provides faster initial load times because the server sends pre-rendered HTML to the client.
- Better Performance on Low-powered Devices: Devices with limited processing power benefit from SSR as they receive pre-rendered HTML, reducing the load on the client's CPU.

Drawbacks of SSR:

- Increased Server Load: SSR requires server-side processing for each request, potentially increasing server load compared to CSR.
- Slower Subsequent Page Loads: While initial page loads are faster with SSR, subsequent page navigation might be slower compared to CSR because each page request requires server rendering.

Contentful:

I developed contentful custom app using Next.js. Contentful provides a variety of tools and resources to help build custom apps, including SDKs and documentation.

https://www.contentful.com/developers/docs/extensibility/app-framework/sdk/

Features Implemented:

Color picker:

The color picker provides the ability to change the color of a specific component, text, or background color of a page.

https://gyazo.com/8791d6b307e28706f3e86b8c3fbe2f92

https://gyazo.com/0f782697c347ce008eaad63b68c97cef

Explanation:

This video above displays a visual tool (like a digital paint palette) that lets users select colors. When a user picks a color, this tool captures their choice and applies it to different parts of a webpage. It ensures users can customize the look of their content easily, enhancing the webpage's visual appeal.

Custom dynamic feature card input field:

The custom dynamic feature card input field provides the ability to select and create a number of input fields based on a value selected in the dropdown.

https://gyazo.com/a77f2f6760e8817f22b3c7bcb26d3e01

https://gyazo.com/917c1d0cd6849f55f8f03ce8bd30e4b8

Explanation:

The above video shows how to dynamically add input fields to a webpage. For example, if they select "3" from a dropdown, the webpage will display three input boxes where users can enter information. This feature provides flexibility in collecting and displaying data, making it easier to manage content on the webpage.

Image/Lottie Inputs:

This media input field provides the ability to either add an image file or a Lottie file, and based on the uploaded file, the output is rendered.

https://gyazo.com/c0a5e9641a327ce668713fd76ff105d2

https://gyazo.com/4b328224967dc62527b0cf81d367fa8e

Explanation:

This video above displays an interactive animation (Lottie) on a webpage. Users can upload an animation file, and this code retrieves the file from a specified location. Once fetched, the animation is displayed on the webpage. This feature allows for dynamic multimedia content to be integrated into webpages easily.

Multi language:

The multi-language feature provides the ability to add multiple languages based on the user's needs.

https://gyazo.com/28b4a5de1f4e6b42cd9cb2d917552543

https://gyazo.com/96c1c4c7a1ac89f63fa458fca89777b9

Explanation:

The above video showcases how users can add multiple languages through Contentful entries and how they are displayed on the webpage.

Conclusion:

The main feedback received highlighted a couple of challenges. First, the custom dynamic feature card input field may be difficult to maintain, particularly if we need multiple instances in our system. Second, the language feature currently fetches data on the client side but needs to be rendered server-side. This might pose implementation challenges within our existing setup.

Despite these issues, the other features are promising and can be implemented effectively. I will use this prototype as a starting point and continue to develop additional features while integrating it into the main codebase, guided by the provided design.