**Rendering Patterns**

**1) Client Side Rendering (CSR)** No Pre-Rendering

**2) Server Side Rendering (SSR)** Pre-Rendering

**3) Static Site Generation (SSG)**

**No Pre-Rendering** - Simple React Application (CSR)

1) The Server Sends HTML Files to the browser

2) Broswer Receives a HTML Files with

<div id="root">

//empty

</div>

3) After the Client is hydrated with JS, our react code runs then it fills that empty space with components

**Pre-Rendering** - Next.JS Application (SSR)

1) The Server sends the HTML files to the browser

2) Browser receives a complete HTML file to display on screen.

3) After the Client is hydrated with JS, our App become inter-active

Pre-Rendering(SSR) is SEO friendly because it renders the data at the server & Gives HTML as output which is final output

**Static Site Generation (SSG)**

1) Building the whole HTML once and deploy it on server

2) When user makes request server can directly provide HTML file, which reduces the processing power of server

3) SSG is a tool user in web development to generate static website

**How static site works with data?**

1) If there is need to pull data from API, DB, etc. SSG Pulls data from various Sources such as JSON Files, DB, API & etc. THe data can include content, metadata, or any other information needed for the website.

**Use Case**

**Client-Side Rendering (CSR)** is a pattern commonly used in web development where the rendering of the user interface is performed on the client side, typically using JavaScript frameworks such as React.js.

**Server-Side Rendering (SSR)** is a pattern used in web development where the server generates the HTML content of web pages and sends the fully rendered page to the client's browser

**Static Site Generation (SSG)** is a pattern used in web development where web pages are pre-built at build time and served to the client as static files.

**Real Life Usages**

**Client-Side Rendering (CSR):**

**Gmail**

Gmail, Google's popular email service, utilizes Client-Side Rendering extensively. When users access their Gmail inbox, the initial HTML content is loaded from the server. Subsequent interactions, such as opening emails, composing new messages, or applying filters, are handled by client-side JavaScript. This allows for a seamless and responsive user experience, where content updates dynamically without requiring full page reloads. Additionally, Gmail's offline mode leverages CSR to enable users to access their emails even when they are not connected to the internet.

**Server-Side Rendering (SSR):**

**Airbnb**

Airbnb, an online marketplace for lodging and travel experiences, employs Server-Side Rendering to deliver dynamic content while ensuring optimal performance and search engine visibility. When users search for accommodations or browse listings, the server generates the HTML content for each page dynamically, incorporating real-time data such as available listings, prices, and reviews. This approach enhances the initial page load performance, facilitates SEO optimization, and provides a consistent user experience across different devices and browsers.

**Static Site Generation (SSG):**

**Next.js Documentation Site**

Next.js, a popular React.js framework, utilizes Static Site Generation for its official documentation website. The documentation content is authored using Markdown and other markup languages, and the entire site is pre-built at build time using Next.js's static site generation capabilities. This approach ensures fast page loads, optimal performance, and low infrastructure overhead. Additionally, the static files generated by Next.js can be easily deployed to a content delivery network (CDN), enabling global distribution and scalability.