Client-Side Vs Server-Side Rendering

In this lesson, we will learn about the client side and the server-side rendering & the use cases for both the approaches.

WE'LL COVER THE FOLLOWING

Client-Side Rendering - How Does A Browser Render A Web Page?

When a user requests a web page from the server & the browser receives the response. It has to render the response on the window in the form of an HTML page.

For this, the browser has several components, such as the:

- *Browser engine*
- *Rendering engine*
- *JavaScript interpreter*
- *Networking & the UI backend*
- *Data storage* etc.

I won't go into much detail but the browser has to do a lot of work to convert the response from the server into an HTML page.

The rendering engine constructs the *DOM* tree, renders & paints the construction. And naturally, all this activity needs a bit of time.

Server-Side Rendering

To avoid all this rendering time on the client, developers often render the UI on the server, generate HTML there & directly send the HTML page to the UI.

This technique is known as the *Server-side rendering*. It ensures faster rendering of the UI, averting the UI loading time in the browser window since the page is already created & the browser doesn't have to do much assembling & rendering work.

Use Cases For Server-Side & Client-Side Rendering

The server-side rendering approach is perfect for delivering static content, such as WordPress blogs. It's also good for SEO as the crawlers can easily read the generated content.

However, the modern websites are highly dependent on Ajax. In such websites, content for a particular module or a section of a page has to be fetched & rendered on the fly.

Therefore, server-side rendering doesn't help much. For every Ajax-request, instead of sending just the required content to the client, the approach generates the entire page on the server. This process consumes unnecessary bandwidth & also fails to provide a smooth user experience.

A big downside to this is once the number of concurrent users on the website rises, it puts an unnecessary load on the server.

Client-side rendering works best for modern dynamic Ajax-based websites.

Though we can leverage a hybrid approach, to get the most out of both techniques. We can use server-side rendering for the home page & for the other static content on our website & use client-side rendering for the dynamic pages.

Alright, before moving down to the database, message queue & the caching components. It's important for us to understand a few concepts such as:

- *Monolithic architecture*
- *Micro-services*
- *Scalability*
- *High availability*

- *High availability*
- *Distributed systems*
- *What are nodes in distributed systems? Why are they important to software design?*

The clarity on these concepts will help us understand the rest of the web components better. Let’s have a look one by one.