Rendering Environments

3 min

In recent years, Next.js has gained significant popularity in the web development community for its versatile and flexible rendering techniques. To grasp the capabilities of Next.js and understand why it is a preferred choice for modern web applications, one must start at the core — rendering environments.

Rendering is the process of converting code into a visual and interactive display that users can view and interact with within a web browser. This process begins when a browser

Preview: Docs Loading link description

requests

a webpage and ends with the server's response, culminating in the rendered application the user interacts with.

The rendering process is a crucial consideration for any web developer. The location of the process can affect several things, such as the overall user experience, web app performance, and search engine visibility to drive organic traffic to the site.

There are two primary rendering **environments**: server and client. **Server-side rendering (SSR)** means that the assembly of the webpage happens mainly on the server, while **Client-side rendering (CSR)** assembles mainly on the client's browser. A well-optimized web application utilizes a combination of both

Preview: Docs Loading link description

methods

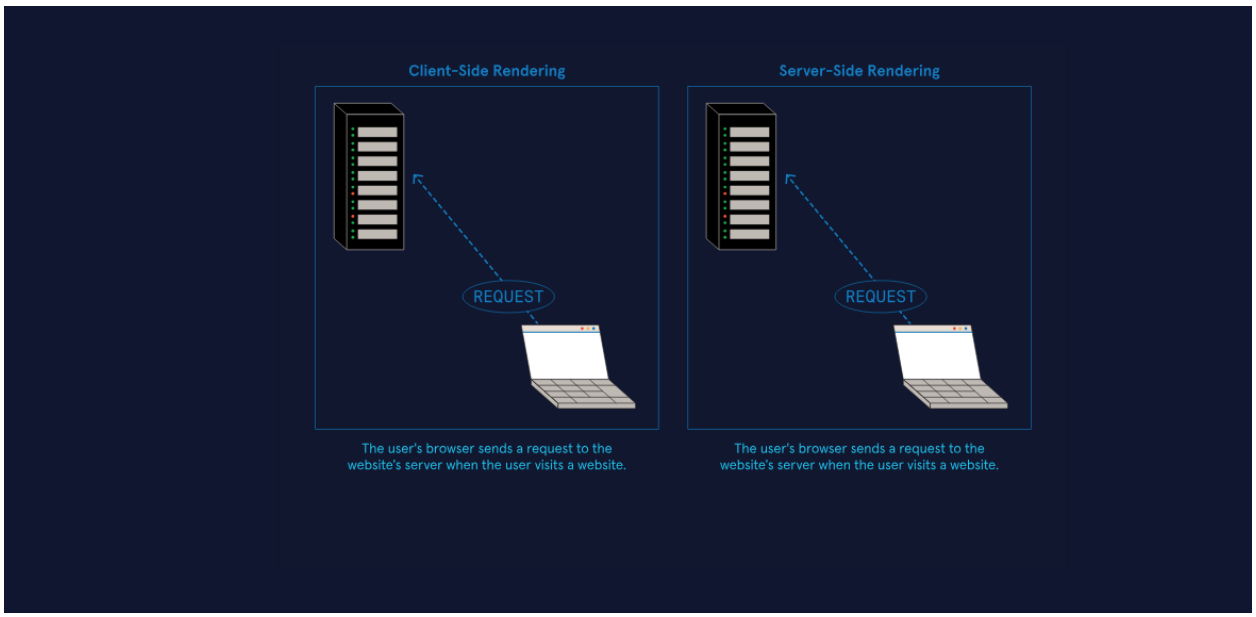
, leveraging the strength of each.

While React supports both, it lacks built-in SSR. This makes Next.js a go-to choice for developers, as it offers robust support for both SSR and CSR. With Next.js, we can specify rendering granularity down to the component level, choosing if it should be server-rendered, client-rendered, or a combination of both.

In the following two exercises, we will explore the concepts of server-side rendering and client-side rendering in detail.

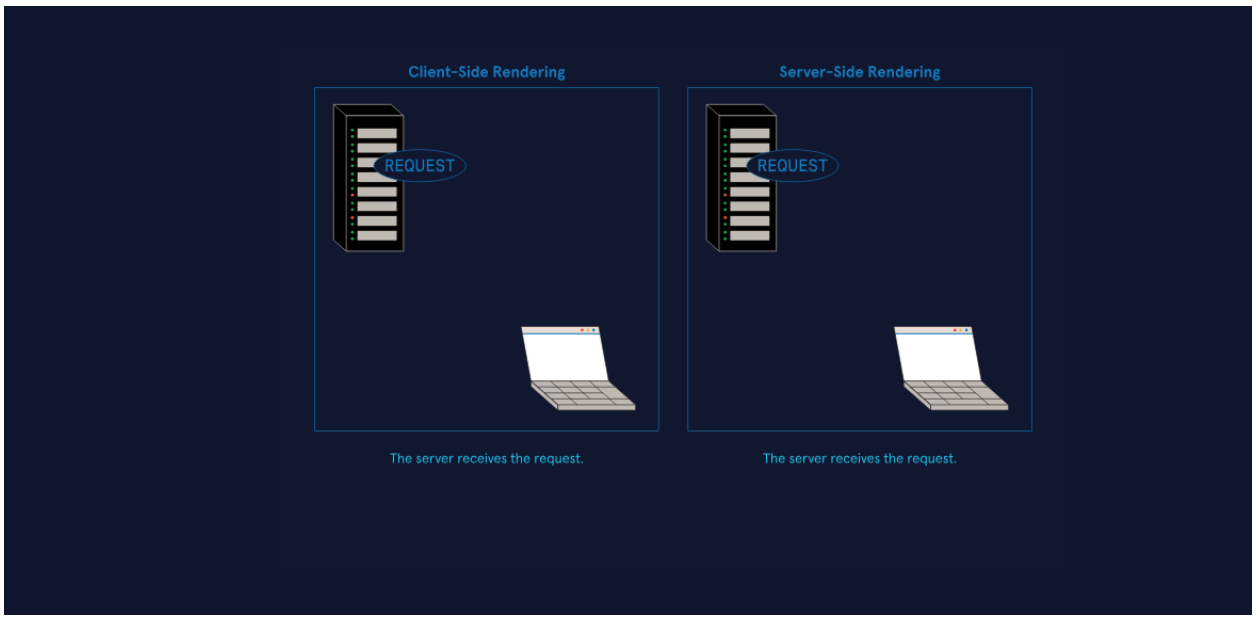
Instructions

Take a look at the diagram provided in the web browser section. Observe and compare the differences between server-side rendering and client-side rendering.



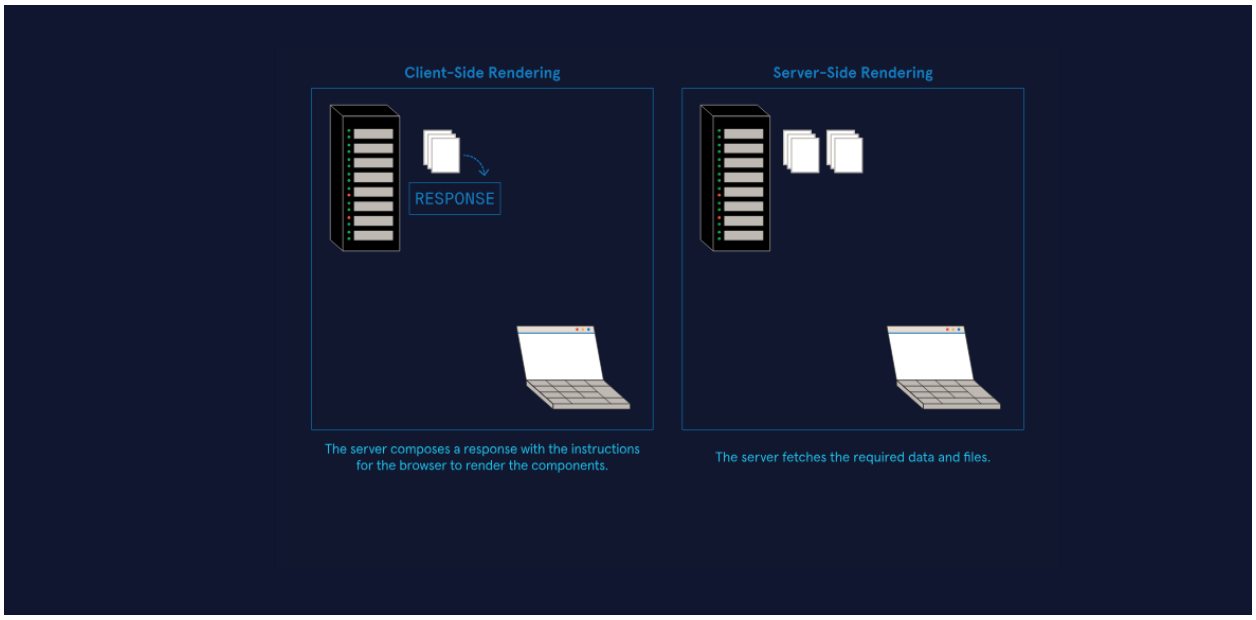
The user's browser sends a request to the website's server when the user visits a website.

The user's browser sends a request to the website's server when the user visits a website.



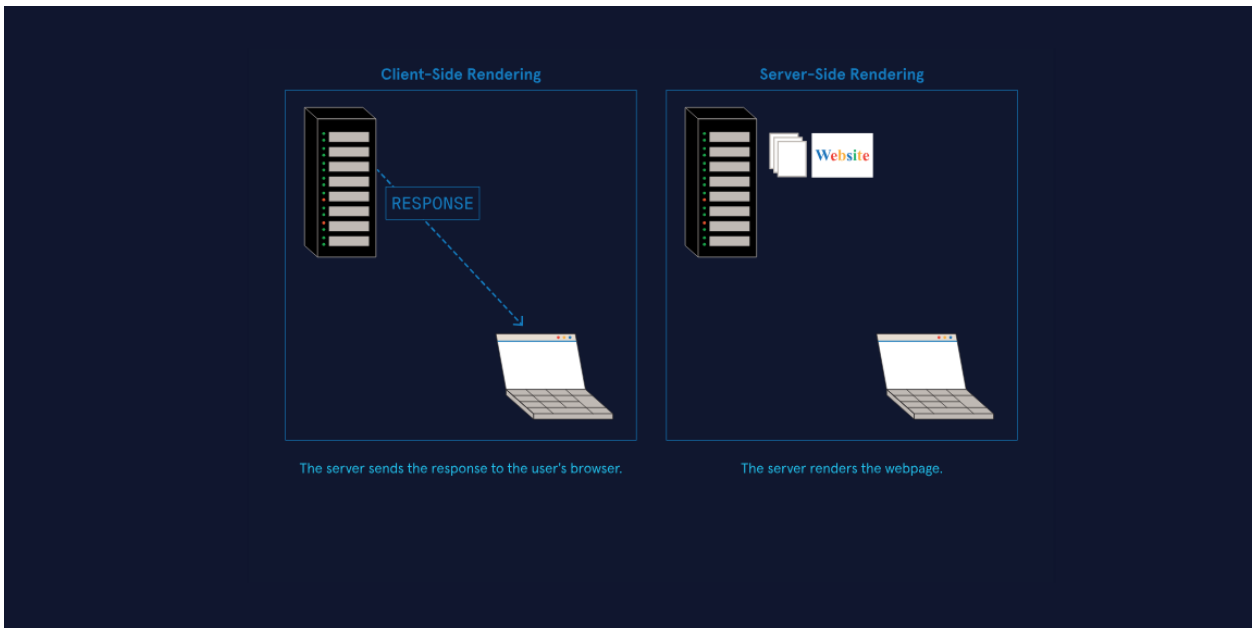
The server receives the request.

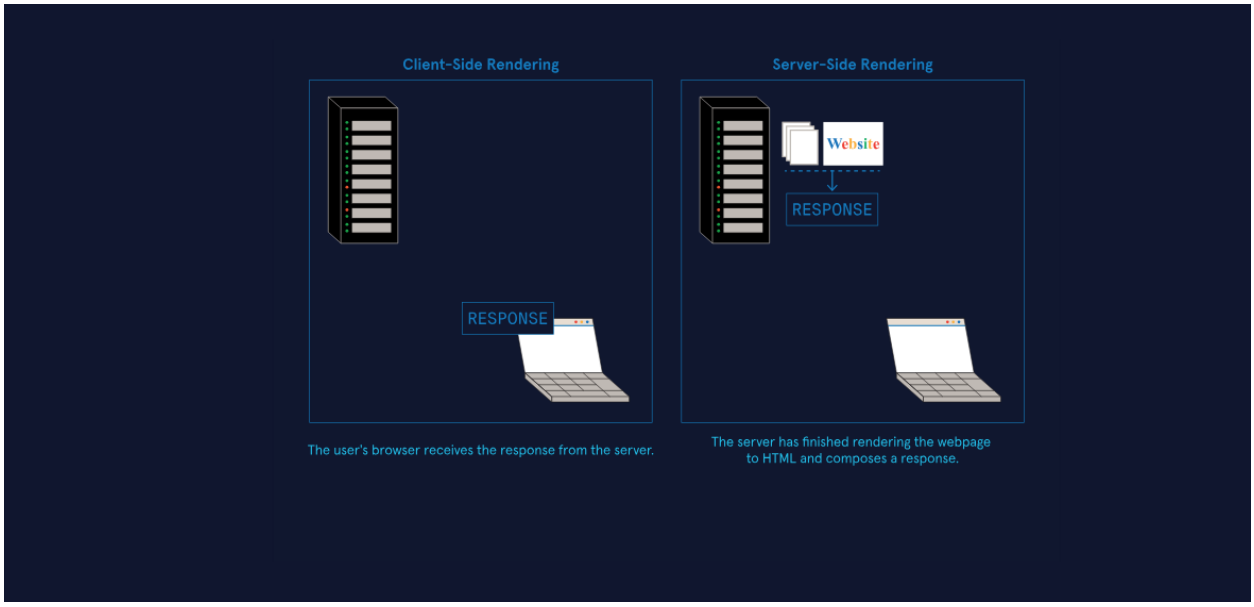
The server receives the request.

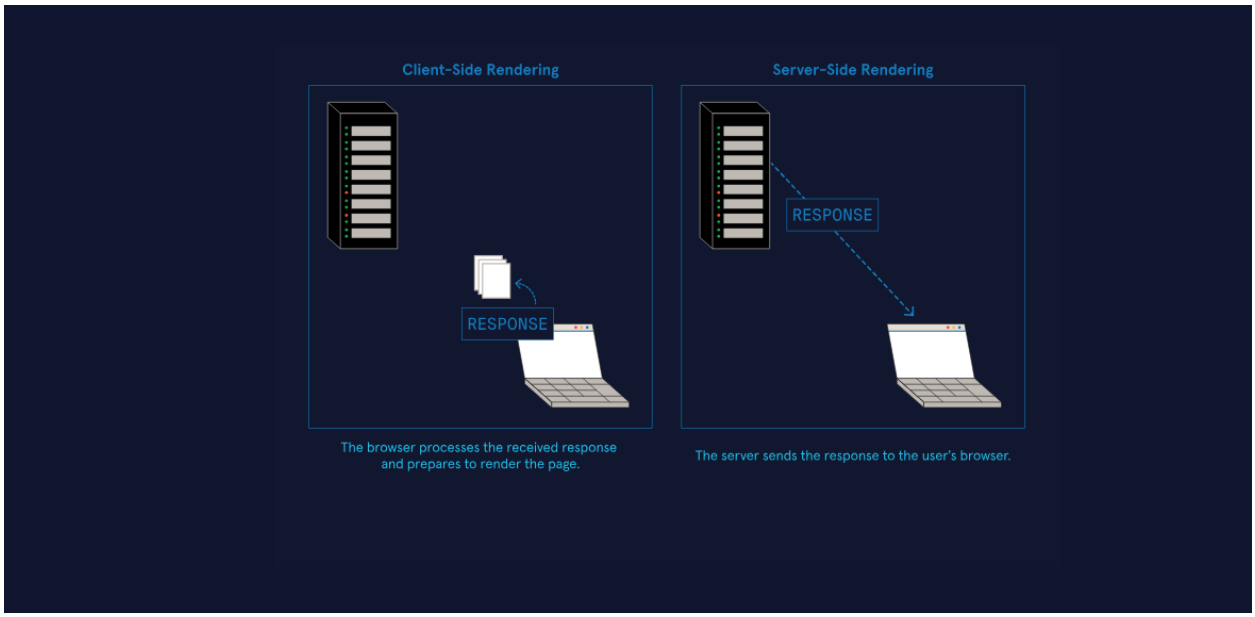


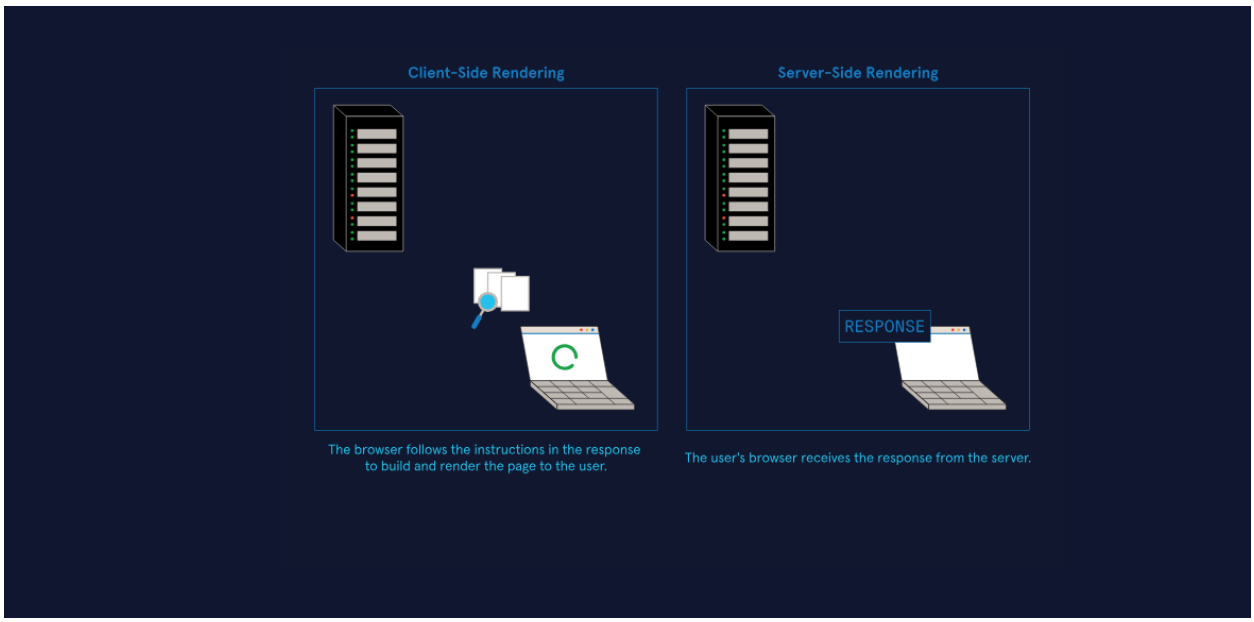
The server composes a response with the instructions for the browser to render the components.

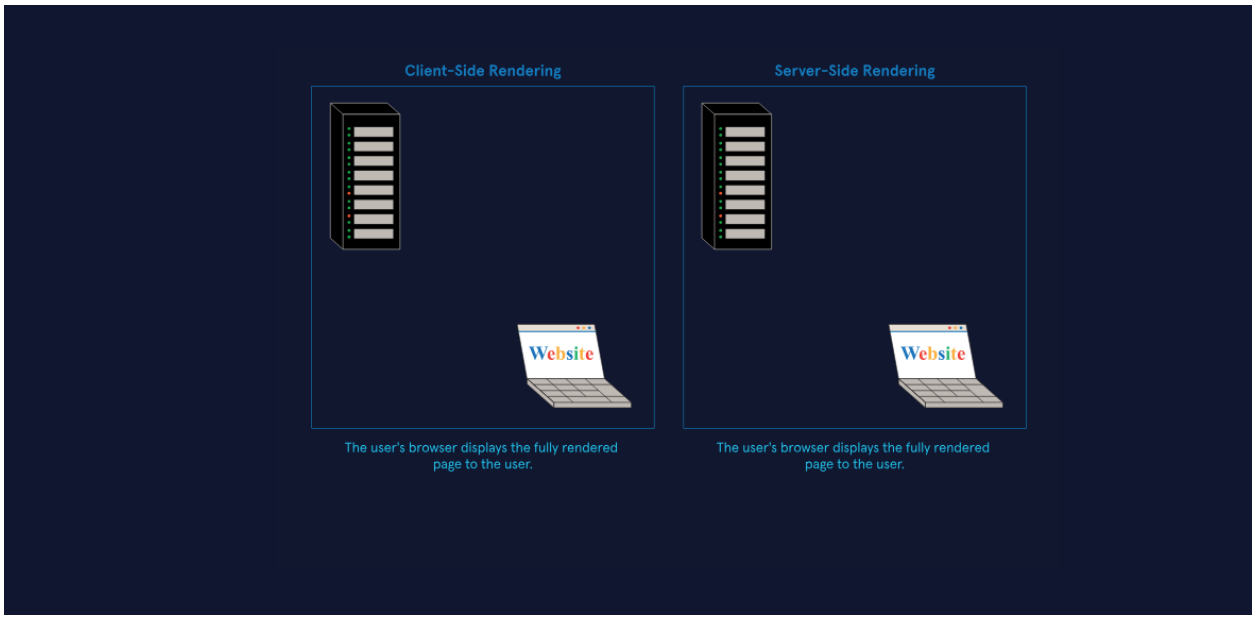
The server fetches the required data and files.


Client-Side Rendering
RESPONSE
Server-Side Rendering
Website
The server sends the response to the user's browser.
The server renders the webpage.


Client-Side Rendering
RESPONSE
Server-Side Rendering
Website
RESPONSE
The user's browser receives the response from the server.
The server has finished rendering the webpage
to HTML and composes a response.


Client-Side Rendering
RESPONSE
Server-Side Rendering
RESPONSE
The browser processes the received response
and prepares to render the page.
The server sends the response to the user's browser.


Client-Side Rendering
Server-Side Rendering
RESPONSE
The browser follows the instructions in the response to build and render the page to the user.
The user's browser receives the response from the server.


Client-Side Rendering
Server-Side Rendering
Website
Website
The user's browser displays the fully rendered page to the user.
The user's browser displays the fully rendered page to the user.