Difference between import ReactDOM from "react-dom/client"; and import ReactDOM from "react-dom";

While both import statements import modules from the React DOM library, the first one specifically targets the client-side rendering APIs, whereas the second one imports the entire ReactDOM package, which encompasses both server-side and client-side rendering functionalities. The choice between them depends on your specific requirements and the features you need to utilize in your application.

Server-side rendering (SSR) and client-side rendering (CSR) are two different approaches to how web pages are generated and displayed to users:

1. Server-side Rendering (SSR): In SSR, the HTML content of a web page is generated on the server in response to a user request. The server processes the request, executes any necessary code (such as fetching data from a database or running business logic), and generates a complete HTML page. This HTML page is then sent to the client (web browser), where it is rendered and displayed to the user. The user receives a fully-formed page ready for display without any additional processing required on the client side.

2. Client-side Rendering (CSR): In CSR, the HTML content of a web page is generated dynamically on the client side, usually by JavaScript code running in the user's web browser. When a user requests a page, the server sends a minimal HTML document along with JavaScript code. The browser then downloads and executes this JavaScript code, which fetches data from APIs and dynamically updates the HTML structure of the page based on the data received. This process is often referred to as "hydration." The user sees a blank page initially, which is then populated with content as the JavaScript executes and updates the DOM (Document Object Model) in the browser.

Key Differences:

- Initial Load Time:

- SSR typically results in faster initial load times because the server sends a pre-rendered HTML page to the client, which can be displayed immediately.

- CSR may have slower initial load times because the browser needs to download and execute JavaScript code before rendering the page, especially if the page relies heavily on client-side rendering for content generation.

- SEO (Search Engine Optimization):

- SSR is generally better for SEO because search engine crawlers can easily parse the fully-formed HTML content sent by the server.

- CSR can pose challenges for SEO because search engine crawlers may have difficulty parsing dynamically generated content rendered client-side.

- Performance:

- SSR may offer better perceived performance for users with slower internet connections or less powerful devices since they receive a fully-rendered page from the server.

- CSR can provide smoother interactions and faster subsequent navigations within the application once the initial JavaScript is loaded, as it can fetch and update content dynamically without requiring full page reloads.

Both SSR and CSR have their advantages and are suitable for different use cases depending on factors such as performance requirements, SEO considerations, and the nature of the application being developed.

**::after pseudo selector**

The ::after pseudo-element should work on an image element (<img>), but there are certain considerations to keep in mind:

1. Content requirement: The ::after pseudo-element is used to insert content after the content of the selected element. Since an <img> element doesn't have any inherent content (like text), you need to ensure there's a parent element wrapping the <img> to which you can apply the ::after pseudo-element.
2. Positioning context: Ensure that the parent element of the <img> has a non-static positioning context (i.e., relative, absolute, or fixed). This is necessary for the ::after pseudo-element to be positioned correctly relative to its parent.

**How to apply css property with setState() show and hide**

 <div

className={`mt-2 text-sm transition-opacity duration-300 ease-in-out ${

              show ? "opacity-100" : "opacity-0"

            }`}

* We are writing JSX that’s why we are using {} in className
* We are using template literals (``) because we are using a state variable to add in a string on a condition. Whenever we want to alter/change our string depending upon any variable we use template literals