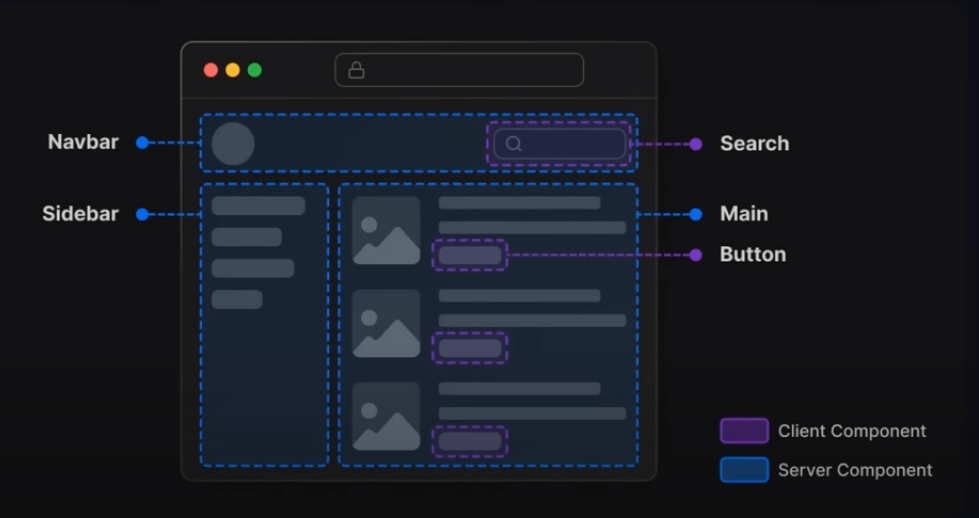
**Server & Client Components**

****

React 18 and next 13 introduced new ways to rendering components. There are two environments where your application code can be rendered.

**Note:**

* By default all components created in next.js within the app folder are react server components, which means that next.js leverages server-side rendering to enhance the initial page loading speed resulting in improved SEO and user experience.   
  For Ex: By default the Home() component in page.js file is the server-side components.

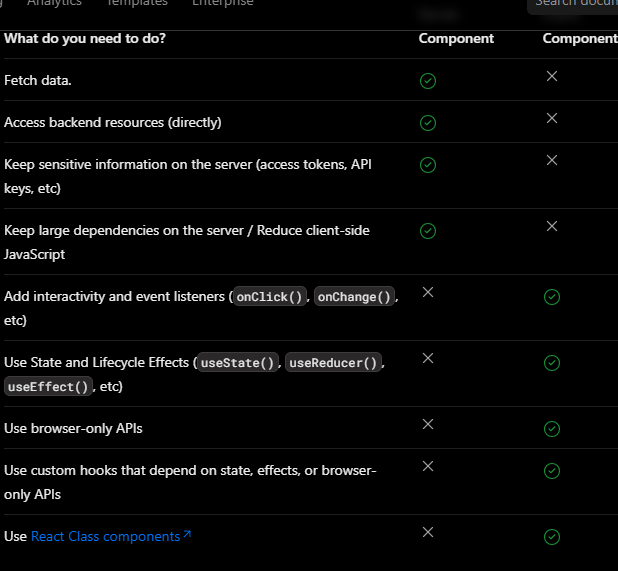
Now in case you want to turn that server-side component by default into a client-side one you need to add the **“use client “** directive to the top of your page to turn it into a client-side component.

Using both the client components and the server components allows us to leverage the benefits of server-side rendering while still utilizing react capabilities for building Dynamic and Interactive user interfaces.

* Whenever you’re utilizing state or hooks like useState in react or other client-side Management Solutions it is important to declare the component as a client-side components otherwise you’re gonna run into errors.  
  State Management in react is primarily handled on the client side where the component state is managed and updated within the browser.

**When should I use server components and when should I transform them to client ones?**

Ans:

  
*More Info:* [*https://nextjs.org/docs/getting-started/react-essentials#when-to-use-server-and-client-components*](https://nextjs.org/docs/getting-started/react-essentials#when-to-use-server-and-client-components)

**Note:**

* You can’t import a server component into a client component.
* You can’t pass a server component as prop or children to a client component.