**1. What is CDN? Why do we use it?**

=> A content delivery network (CDN) refers to a geographically distributed group of servers which work together to provide fast delivery of Internet content.

A CDN allows for the quick transfer of assets needed for loading Internet content including HTML pages, javascript files, stylesheets, images, and videos.

\* Is a CDN the same as a web host?

> While a CDN does not host content and can’t replace the need for proper web hosting, it does help cache content at the network edge (Edge computing is a networking philosophy focused on bringing computing as close to the source of data as possible in order to reduce latency and bandwidth use.), which improves website performance.

\* Benefits of using a CDN:

1. Improving website load times - By distributing content closer to website visitors by using a nearby CDN server (among other optimizations), visitors experience faster page loading times. As visitors are more inclined to click away from a slow-loading site, a CDN can reduce bounce rates and increase the amount of time that people spend on the site. In other words, a faster a website means more visitors will stay and stick around longer.

2. Reducing bandwidth costs - Bandwidth consumption costs for website hosting is a primary expense for websites. Through caching and other optimizations, CDNs are able to reduce the amount of data an origin server must provide, thus reducing hosting costs for website owners.

3. Increasing content availability and redundancy - Large amounts of traffic or hardware failures can interrupt normal website function. Thanks to their distributed nature, a CDN can handle more traffic and withstand hardware failure better than many origin servers.

4. Improving website security - A CDN may improve security by providing DDoS (Distributed Denial of Service) mitigation, improvements to security certificates, and other optimizations.

Source: <https://www.cloudflare.com/learning/cdn/what-is-a-cdn/>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**2. Why is React known as React?**

- Both React and ReactDOM collectively constitutes the React Library.

- React uses CDN, because of which the websites are loaded fast.

- We can inject React in our code by using CDN Links.

Why you should not use CDN is present in the same Facebook page you have given.

If you prefer to use your own text editor, you can also download this HTML file, edit it, and open it from the local filesystem in your browser. It does a slow runtime code transformation, so don't use it in production.

To say it clearly,

**>>Using CDN:**

As Facebook said, it does a slow runtime code transformation in the browser.

That is your code doesn't execute right-away.

First your code should be converted to JavaScript so that browsers can execute it, as JSX is not supported.

After the conversion to JavaScript, then the browsers execute it.

***On client's browser:***

JSX -> JavaScript -> Execute

**Source**: <https://stackoverflow.com/questions/41885560/why-should-i-not-use-cdn-for-react-babel>

- React gives us advantage of using JSX. JSX is a syntactic sugar. You can write less code with JSX.

- React uses has Virtual DOM which uses Diff Algorithm which is used for Reconciliation.

- React at the end of the day is JavaScript which Facebook/Meta engineers wrote for us and they have given us superpowers, just like we have Window object in JavaScript.

- We get superpowers of react from the React Library, and the superpowers can be put in the code using several ways, one such way is using CDN (inject React). CDN is the faster way of fetching that code.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**3. What is “crossorigin” in script tag?**

The **crossorigin** attribute provides support for [CORS](https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS), defining how the element handles cross-origin requests, thereby enabling the configuration of the CORS requests for the element's fetched data. Depending on the element, the attribute can be a CORS settings attribute.

Valid on the [<audio>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/audio), [<img>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img), [<link>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/link), [<script>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/script), and [<video>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/video) elements

**Cross-Origin Resource Sharing** ([CORS](https://developer.mozilla.org/en-US/docs/Glossary/CORS)) is an [HTTP](https://developer.mozilla.org/en-US/docs/Glossary/HTTP)-header based mechanism that allows a server to indicate any [origins](https://developer.mozilla.org/en-US/docs/Glossary/Origin) (domain, scheme, or port) other than its own from which a browser should permit loading resources. CORS also relies on a mechanism by which browsers make a "preflight" request to the server hosting the cross-origin resource, in order to check that the server will permit the actual request. In that preflight, the browser sends headers that indicate the HTTP method and headers that will be used in the actual request.

Source: <https://developer.mozilla.org/en-US/docs/Web/HTML/Attributes/crossorigin>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4. What is difference between React and ReactDOM?

React is not just limited to browsers, it works for Mobile devices for rendering and lot more that is the reason for 2 files.

Development files:

react@18: @18 shows the latest version of react. Assume this one as core library of React

react-dom@18: This is the web version of react. This library gives access to Document Object Model (DOM).

Suppose we want to write some React code which goes into the browser, it happens in the react-dom file.

<script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>

<script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>