Namaste React:

1. What is CDN?

A (CDN), content delivery network is a network of interconnected servers that speeds up webpage loading for data-heavy applications.

A CDN improves efficiency by introducing intermediary servers between the client and the website server.

These CDN servers manage some of the client-server communications. They decrease web traffic to the web server,

reduce bandwidth consumption, and improve the user experience of your applications.

1. What is cross origin?

Cross origin attribute is used to share the resources from one domain to another domain.

cross origin attribute sets the mode of the request to an HTTP CORS Request.

CORS is used to manage cross-origin requests.

CORS stands for Cross-Origin Resource Sharing, and is a mechanism that allows resources on a web page to be requested from another domain outside their own domain.

It defines a way of how a browser and server can interact to determine whether it is safe to allow the cross-origin request. (for sharing the resources from other domains.)

1. Why is React known as React?

React known as React because It "reacts" quickly to changes without reloading the whole page.

It uses the virtual DOM to efficiently update parts of a webpage.

It's built around components that "react" and update.

1. What is Emmet?

Emmet is an essential development tool that helps you quickly create repetitive structures like lists, tables, or ordered elements with minimal typing.

1. Difference between a Library and Framework?

|  |  |  |
| --- | --- | --- |
| Points | Library | Framework |
| **Inversion of Control (IoC):** | Libraries do not impose any inversion of control. Developers retain control over the flow of their application, and they call the library when needed. | In a framework, the control flow is inverted, meaning the framework calls your code. Your code is essentially plugged into the framework, and the framework dictates the flow of control. |
| Architecture: | Libraries are typically less opinionated about the overall structure of your code. They provide specific functionalities but leave the organization and architecture of your application largely up to you. | Frameworks often come with a predefined architecture and design patterns. They guide the developer on how to structure their code, making certain decisions on behalf of the developer. |
| Extensibility: | Libraries are more modular and can be used selectively. You can choose specific components of a library that suit your needs without being bound to a specific architecture. | Frameworks are typically more integrated and comprehensive. They provide a set of tools, conventions, and abstractions to handle a broad range of tasks. Extending the functionality of a framework often involves adhering to its established patterns. |
| Learning Curve: | Libraries usually have a more straightforward learning curve because they focus on providing specific functionalities without enforcing a particular structure. | Due to their opinionated nature and comprehensive features, frameworks might have a steeper learning curve. However, once mastered, they can streamline development. |
| Examples: | React (JavaScript): Initially conceived as a library for building user interfaces but often referred to as a framework due to its component-based architecture. | Django (Python): A web framework that follows the MVC (Model-View-Controller) pattern.  Spring (Java): A framework for building Java-based enterprise applications |
|  |  |  |

In summary, frameworks are more comprehensive and opinionated, dictating the overall structure and flow of your application.

Libraries, on the other hand, are more modular and allow developers more freedom in organizing their code while providing specific functionalities.

1. What is difference between React and ReactDOM

React: Manages the core logic of components, state, and the virtual DOM. It's platform-independent and can be used for various environments beyond the web.

React DOM: Handles the interaction between React and the browser's DOM (Document Object Model). It's specific to web applications and is responsible for rendering React components into the actual web page.

React:

Core Library: React is a JavaScript library for building user interfaces. It provides the fundamental building blocks for creating components, managing state, and handling UI updates.

Platform-Agnostic: React is designed to be platform-agnostic, meaning it can render UI components not only in web browsers but also on mobile devices using React Native or other platforms.

Component Model: React introduces a component-based architecture, where UI elements are encapsulated into reusable components, making it easier to maintain and scale applications.

Virtual DOM: React uses a virtual DOM to efficiently update the UI. It compares the virtual DOM with the actual DOM and only applies the necessary changes, resulting in improved performance.

React DOM:

DOM Rendering: React DOM is a package specifically designed for rendering React components in the DOM (Document Object Model) of a web browser.

Integration with Web Browsers: React DOM provides methods for mounting React components into HTML elements, updating them, and handling events within the browser environment.

Entry Point: React DOM serves as the entry point for React applications targeting web browsers. It includes functionality for rendering components, managing their lifecycle, and interacting with the DOM.

Separation of Concerns: By separating the rendering logic into React DOM, react can remain platform-agnostic, allowing developers to use different renderers for various environments.

In summary, react is the core library for building user interfaces, providing a component-based architecture and virtual DOM for efficient updates.

React DOM, on the other hand, is a package specifically for rendering React components in web browsers, handling interactions with the DOM and providing integration with the browser environment.

1. What is difference between react.development.js and react.production.js files via CDN?

react.development.js: This file is larger in size and includes extra warnings, error messages, and development-specific tools.

It's intended for use during the development phase of your application.

react.production.js: This file is smaller in size and optimized for production use. It excludes the extra warnings and debugging tools, making it more efficient

and suitable for deployment to a live, production environment

When you're actively developing your React application, you might use the development version for helpful warnings and debugging information.

When you're ready to deploy your application to production, you switch to the production version to optimize performance and reduce file size.

1. What is async and defer? What is async and defer?

✦ Async vs Defer Attributes in JavaScript

• HTML parsing and scripts with the async tag are fetched from the network asynchronously

• Scripts are executed as soon as they are available in the browser, and then HTML parsing continues

✦ Defer attribute delays script execution until HTML parsing is complete.

• HTML parsing continues while scripts are fetched in parallel.

• Scripts are executed only after HTML parsing is fully complete.