1. What is Emmet?

Emmet is a shorthand syntax and plugin (built into VS Code and many editors) that lets you write HTML/CSS much faster using abbreviations that expand into full markup.

Example:
Typing `ul>li*3` expands to:

```
<ul>
  <li></li>
  <li></li>
  <li></li>
</ul>
```

Emmet is a productivity toolkit that uses abbreviations to generate HTML/CSS boilerplate instantly. It speeds up repetitive markup creation and is built into modern editors like VS Code.

2. Difference between a Library and Framework?

Library:

- You call it when you need it.
- *You* control the flow.
- Example: React (it handles UI updates but you decide when/how to use it).

Framework:

- It calls *your* code.
- The framework controls the overall flow (“inversion of control”).
- Example: Angular (it decides structure, lifecycle, routing, etc.).

Short version:
Library = you’re in charge.
Framework = the framework is in charge.

3. What is CDN? Why do we use it?

A **CDN (Content Delivery Network)** is a globally distributed network of servers that deliver static assets (JS, CSS, images, videos) from locations closest to the user.

Why we use it:

- **Lower latency:** assets load faster because they come from a nearby server.
- **Reduced load on origin server:** improves scalability and reliability.
- **Better performance & SEO:** faster pages improve user experience and Core Web Vitals.
- **Built-in caching & DDoS protection** (depending on the provider).

Short version:
CDN = faster, more reliable delivery of static content by serving it from servers close to the user.

4. Why is React known as React?

React is called **"React"** because its core idea is to **react** to changes in data and update the UI efficiently.

When state changes, React automatically re-renders the necessary parts of the UI—*reacting* to that change—using its virtual DOM and diffing algorithm.

Short version:
It's named "React" because it reacts to state changes and updates the UI in a fast, predictable way.

5. What is crossorigin in script tag?

`crossorigin` in a `<script>` tag tells the browser **how to handle cross-origin requests** for that script, mainly for **CORS** and **error reporting**.

There are two common values:

1. `crossorigin="anonymous"`

- Sends the request *without* credentials (cookies, tokens).
- Needed when loading third-party scripts with **subresource integrity (SRI)**.
- Allows proper error reporting if the server allows it via CORS.

2. `crossorigin="use-credentials"`

- Sends credentials (cookies, authorization headers).
- Only works if the server explicitly allows it in CORS headers.

Short version:
`crossorigin` **defines whether a script loaded from another domain should include credentials and enables proper CORS behavior and error reporting.**

6. What is the difference between React and ReactDOM?

React

- The core library for building UI components.
- Handles component logic, state, hooks, reconciliation, etc.
- Framework-agnostic (doesn't depend on the browser).

ReactDOM

- The library that connects React to the browser DOM.
- Handles rendering components into actual DOM nodes (`createRoot`, `render`).
- Provides DOM-specific methods.

Short version:
React = component logic.
ReactDOM = renders those components into the browser DOM.

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

react.development.js

- Includes warnings, error messages, and dev tools helpers.
- Bigger file size.
- Slower because it does extra checks.
- Used during development for better debugging.

react.production.js

- All warnings and dev-only checks removed.
- Minified and optimized.
- Much smaller and faster.
- Used in production for performance.

Short version:
Development = helpful warnings + bigger + slower.
Production = no warnings + smaller + faster.

8. What is async and defer?

Both `async` and `defer` improve page performance by loading scripts **without blocking HTML parsing**, but they behave differently.

`async`

- Script is downloaded **in parallel** with HTML parsing.
- **Executes immediately** once downloaded (may interrupt HTML parsing).
- Good for independent scripts (e.g., analytics).

Order NOT guaranteed.

`defer`

- Script is downloaded **in parallel** with HTML parsing.
- **Executes only after HTML is fully parsed**.
- Execution order **is preserved**.

Best for scripts that depend on the DOM or on each other.

Short version:

`async` **= load in parallel + execute ASAP (no order guarantee).**
`defer` **= load in parallel + execute after HTML parsing (order preserved).**