1) What is react and why we use react

=> Developed and Maintained by facebook

=> It helps developers create reusable UI components

for example buttons, forms, cards etc..

=> It updates and displays only the parts of the webpage that change, without reloading the entire page

Why

Virtual DOM -> Improves Speed

Components -> Reusable UI parts

JSX -> Write HTML inside JS

one way data flow -> Easier to debug -> one direction (from parent to child)

Ecosystem -> many tools available

↳ React Router (for navigation)

=> Redux / Context API (for state management)

=> Next.js (for server-side rendering)

Where

1. Single Page Applications -> Facebook, Instagram
2. Dynamic websites/ dashboards
3. Mobile Apps (with React Native)
4. Reusable component-based API
5. Frontend (for API) Backend

=> Build fast, interactive and reusable user interfaces for both web and mobile apps

2) What is the difference between HTML using webpage and react using webpage.

Feature	HTML Webpage	React Webpage
1. Type	Static	Dynamic single Page Application (SPA)
2. Language Used	HTML, CSS, JS	JSX (JavaScript + HTML) together
3. Structure	Multiple .html files (Home.html, about.html)	only one main index.html everything handled by react

4. Rendering	Browser renders full page every time	React renders only the changed part using virtual DOM
5. Page Reload	Yes, reloads on every change	No reload - updates instantly
6. Speed	Slower for large apps	Faster and smoother performance.
7. Reusability	Code repeated in every page	Components can be reused easily
8. Data Handling	Manual DOM manipulation using JS	Automatic state & props handling
9. User Experience	Feels like separate pages	Feels like one continuous app.
10. Learning curve	Easy (basic HTML knowledge)	Needs to learn JSX, components, state etc..
11. Best For	Small, static sites (portfolio, Blog)	Big, interactive apps (Dashboard, chat, E-commerce)
12. Example file setup	index.html, about.html, contact.html.	one index.html + many React components like App.js, Navbar.js etc..
13. Maintenance	Harder (code duplication)	Easier (centralized & reusable components)

3) What is the difference between react and angular.

Feature	React	Angular
Developed by	Meta (facebook)	Google
Type	Javascript library	Full-fledged framework
Languages used	Javascript (with JSX)	Typescript (superset of Js)
DOM type	Virtual DOM (faster)	Real DOM (slower for big apps)

Data Binding	one way data binding (data flows from parent -> child)	Two way data binding (model <-> view sync automatically)
Learning Curve	Easier to learn	Harder / steeper (many built-in concepts)
Performance	Faster (due to virtual DOM)	slightly slower for large apps (due to Real DOM checks)
Routing / State	Needs external libraries like Redux, zustand, context API	Built-in with RxJS and services
Reusability	Component-based (reusable UI parts)	Also Component-Based but tightly structured
App size	Generally smaller.	Slightly larger due to full framework features
Best for	lightweight, fast, flexible apps	Enterprise level, large, structured applications
Community Support	Very large and active	Large, but not as active as React's

4) What is typescript

=) Typescript is a superset of Javascript - which means its built on top of JS, and adds extra features, mainly type checking

=) It was developed by Microsoft.

Typescript = Javascript + Type safety

That means all valid Javascript code also works in Typescript - but Typescript helps you catch errors early and write cleaner, safer code.

How it works

1. You write code in a .ts file
2. Typescript compiler converts it to js
3. Browser runs the Javascript

Why we use

=> Fewer runtime errors

=> Easier to maintain large projects

=> Cleaner, more readable code

=> used in modern frameworks like Angular, Next.js, Next JS

=> Typescript is Javascript with added type safety and better developer tools

5) What is

Variable

A variable is a container used to store data or values in a program.

Declaration

=> When you create a variable (memory creation phase)

=> variable keyword variable name

Initialization

When you assign a value to the declared variable.

Lexical Scope

Lexical scope means a variable can be accessed only inside the block or function where it is defined and its inner scopes, but not outside

=> Inner() can use variables and can access outer()

Block scope

A block means anything inside { }

Variables declared with let or const are block-scoped they exist only inside that block.

Functional Scope

Variables declared with var are function scoped- they are visible throughout the entire function.

Hoisting

Hoisting means variables and functions are moved to the top of the scope before code runs

Closure

A closure happens when an inner function remembers variables from its outer function, even after the outer function has finished executing.

=> Even after outer() finishes, the inner() function remembers count - this is a closure.

6) Why Node.js comes in Web Development

=> What is Node.js

=> Node.js is not a framework or library. It's a runtime environment that lets you run JS outside the browser. (for example on a server)

=> Normally, JS runs only inside the browser. With Node.js you can also run it on your computer or server, just like python or Java.

=> Backend Development

=> Connect frontend and database

=> Package management

=> Fast and lightweight

Which came first.

Tool	Release Year	Developed by	Type
Node.js	2009	Ryan Dahl	Javascript runtime (Backend)
Angular	2010 (AngularJS), rewritten as Angular 2+ in 2016	Google	Frontend framework
React	2013	Meta (Facebook)	Frontend library

Eg: modern web Development

Today, developers often use them together:

- Node.js + Express -> backend
- React or Angular -> frontend
- MongoDB + SQL -> database.

Node.js -> runs JavaScript on server

Angular -> builds frontend (Google)

React -> builds frontend (Facebook)

Concept	Role	Year
Node.js	Backend runtime + npm support	2009
Angular	Frontend framework	2010
React	Frontend library	2013

7. CRA

Folders and files

-> node_modules
.bin, .cache, @babel ....

=> public
-> favicon.ico
=> index.html
-> logo192.png
-> logo512.png
=> manifest.json -> {—}
=> robots.txt

VITE => src
-> App.css
-> App.js
-> App.test.js
-> index.css
-> index.js
-> logo.svg
-> reportWebVitals.js
-> setupTests.js

=> .gitignore
=> package.json
=> package-lock.json
=> README.md.

VITE

=> node_modules
.bin, .vite, .vite-temp

=> public
-> vite.svg

=> src
-> assets -> react.svg
-> App.css

-> App.jsx

->index.css

->main.jsx

=> .gitignore

=> eslint.config.js

=> index.html

=> package.json

=> package-lock.json

=> README.md

=> vite.config.js