**THÀNH PHẦN FRONTEND - WEB APP - MOBILE WEB- HYBRID APP**

1. **HTML (Structure Layer – Tầng cấu trúc)**
2. **CSS (Presentation Layer – Tầng trình bày)**
3. **JavaScript (Logic Layer – Tầng hành vi)**
4. **DOM, CSSOM, Render Tree & Rendering Pipeline**
5. **Frontend Application Logic (SPA / CSR / SSR / SSG)**
6. **Asset & Build System**
7. **Frontend Security & Network Layer**
8. **Runtime & Rendering Environment**
9. **State Management & Data Layer**
10. **UI Components & Design System**
11. **Storage & Caching Layer**
12. **Dev Tools & Testing**

**1. HTML (Structure Layer – Tầng cấu trúc)**

* Nhiệm vụ: Xác định cấu trúc và nội dung của trang (DOM tree).
* Ví dụ: <div>, <header>, <p>, <button>, v.v.
* Output: Tạo ra DOM (Document Object Model) – cấu trúc cây mà trình duyệt hiểu và render.

Mọi thứ bạn thấy trên trang web (text, hình, layout) đều bắt nguồn từ HTML.

**2. CSS (Presentation Layer – Tầng trình bày)**

Nhiệm vụ: Quy định giao diện, bố cục, màu sắc, animation, responsive layout.

Bao gồm:

* CSS cơ bản (style.css)
* CSS preprocessor (SASS, LESS)
* CSS-in-JS (Styled Components, Emotion)
* Framework (Tailwind, Bootstrap, MUI,…)

CSS tác động lên CSSOM (CSS Object Model) → kết hợp với DOM để tạo Render Tree.

**3. JavaScript (Logic Layer – Tầng hành vi)**

Nhiệm vụ: Xử lý logic, event, state, API call, render động, routing.

Vai trò:

* Tương tác DOM (thêm/xóa node, cập nhật UI)
* Gọi dữ liệu từ backend (AJAX / Fetch / GraphQL)
* Điều khiển reflow / repaint
* Xử lý logic giao diện (form validation, animation, routing,...)

JavaScript là cầu nối giữa người dùng ↔ giao diện ↔ dữ liệu.

1. **DOM, CSSOM, Render Tree & Rendering Pipeline**

Trình duyệt tạo ra 4 “thế giới” song song:

* DOM Tree – từ HTML
* CSSOM Tree – từ CSS
* Render Tree – kết hợp DOM + CSSOM
* Layout + Paint + Composite – quá trình trình duyệt vẽ ra UI thực tế

JavaScript có thể tác động vào đây gây:

* Reflow (Layout lại) → tốn CPU
* Repaint (Vẽ lại pixel) → tốn GPU

**5. Frontend Application Logic (SPA / CSR / SSR / SSG)**

Tuỳ loại dự án:

* CSR (Client-side rendering): React, Vue, Angular
* SSR (Server-side rendering): Next.js, Nuxt
* SSG (Static Site Generation): Astro, Next.js, VitePress

Mục tiêu: Tối ưu tốc độ tải, SEO và trải nghiệm người dùng.

**6. Asset & Build System**

* Bundler / compiler: Vite, Webpack, esbuild, Parcel
* Transpiler: Babel, SWC, TypeScript
* Minifier: Terser, CSSNano
* Static analyzer: ESLint, Prettier

Giúp tối ưu hiệu năng, code splitting, lazy loading, tree shaking,…

**7. Frontend Security & Network Layer**

* HTTPS, CORS, CSP (Content Security Policy)
* XSS / CSRF protection
* Token storage (JWT, Cookie, LocalStorage, SessionStorage)
* HTTP caching & service workers

**8. Runtime & Rendering Environment**

* Browser Engine: Blink (Chrome), WebKit (Safari), Gecko (Firefox)
* Rendering Pipeline: Parse → DOM → CSSOM → Layout → Paint → Composite

**9. State Management & Data Layer**

* Local UI state (React useState, Vue reactive)
* Global store: Redux, Zustand, Recoil, MobX
* Server state: React Query / TanStack Query, SWR

**10. UI Components & Design System**

* Atomic components: Button, Input, Card, Modal,...
* Design system: Material UI, Ant Design, Shadcn/UI
* Theming, typography, spacing, responsive grid

**11. Storage & Caching Layer**

* LocalStorage, SessionStorage
* IndexedDB, Cache API
* Service Worker (PWA offline cache)

**12. Dev Tools & Testing**

* DevTools (Elements, Network, Performance)
* Unit test: Jest, Vitest
* E2E test: Cypress, Playwright
* Accessibility test: Axe, Lighthouse

Tóm tắt sơ đồ:

HTML → DOMCSS → CSSOMDOM + CSSOM → Render TreeRender Tree → Layout → Paint → Composite → UI

↑

JavaScript