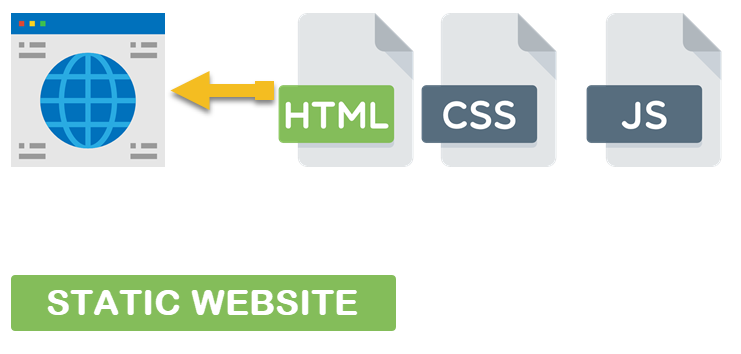
# Chương 1: TỔNG QUAN VỀ FRONTEND

1. **Frontend là gì:**

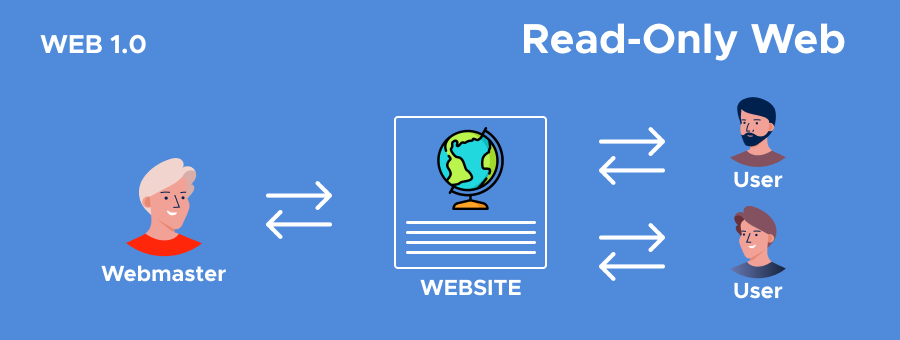
* Front-end là phần giao diện người dùng của một trang web hoặc ứng dụng di động mà người dùng có thể nhìn thấy và tương tác trực tiếp. Nó bao gồm tất cả các yếu tố trực quan như bố cục, màu sắc, phông chữ, hình ảnh, nút bấm, menu, v.v.
* Nói cách khác, front-end là "mặt tiền" của một trang web hoặc ứng dụng, là thứ đầu tiên mà người dùng nhìn thấy và trải nghiệm. Nó đóng vai trò quan trọng trong việc thu hút người dùng và tạo ấn tượng ban đầu tốt đẹp.
* Công việc chính của lập trình viên front-end là sử dụng các ngôn ngữ lập trình như HTML, CSS và JavaScript để xây dựng và duy trì giao diện người dùng. Họ cũng cần có kiến thức về các công nghệ web khác như frameworks, libraries và tools để tạo ra các trang web và ứng dụng web hiện đại, hiệu quả và tương thích với nhiều trình duyệt khác nhau.

1. **Lịch sử phát triển:**

* Lĩnh vực front-end đã trải qua nhiều giai đoạn phát triển từ khi ra đời cho đến nay, với sự ra đời của các công nghệ mới và sự thay đổi trong nhu cầu của người dùng. Dưới đây là một số mốc quan trọng trong lịch sử phát triển của front-end:
* Giai đoạn đầu (1990 - 1995):
* HTML là ngôn ngữ duy nhất được sử dụng để xây dựng trang web.
* Trang web chủ yếu là văn bản và hình ảnh tĩnh.
* Tương tác của người dùng bị hạn chế.



* Giai đoạn Web 1.0 (1995 - 2004):
* JavaScript được giới thiệu, cho phép tạo ra các trang web tương tác hơn.
* CSS được phát triển để tách biệt nội dung và giao diện của trang web.
* Các trình duyệt web bắt đầu hỗ trợ nhiều tính năng mới như frames và cookies.



* Giai đoạn Web 2.0 (2004 - 2010):
  + Sự ra đời của các ứng dụng web (web application) như Gmail, Facebook và YouTube.
  + AJAX (Asynchronous JavaScript and XML) được sử dụng để tạo ra các ứng dụng web động và tương tác hơn.
  + Các frameworks JavaScript đầu tiên như jQuery ra đời, giúp đơn giản hóa việc viết mã JavaScript.

A blue background with arrows pointing to a globe

Description automatically generated

* Giai đoạn Web 3.0 (2010 - nay):
  + Sự bùng nổ của các thiết bị di động như smartphone và tablet.
  + HTML5 và CSS3 được phát triển để hỗ trợ các tính năng mới cho thiết bị di động.
  + Các frameworks JavaScript hiện đại như React, Angular và Vue.js ra đời, giúp xây dựng các ứng dụng web phức tạp và hiệu quả hơn.
  + Mobile-first trở thành xu hướng thiết kế web, ưu tiên trải nghiệm người dùng trên thiết bị di động.

A blue background with icons and a person wearing virtual reality glasses

Description automatically generated

## Thời kì desktop applications:

* Thời kỳ Desktop Applications (Ứng dụng Máy Tính) bắt đầu từ những năm 1970 với sự ra đời của các máy tính cá nhân như Apple II, Commodore PET và IBM PC. Các ứng dụng máy tính thời kỳ này thường được viết bằng các ngôn ngữ lập trình như C, Pascal và BASIC, và được phân phối trên đĩa mềm hoặc CD-ROM.
* Đặc điểm nổi bật của thời kỳ Desktop Applications:
  + Tính độc lập: Ứng dụng máy tính chạy độc lập trên máy tính của người dùng, không cần kết nối internet.
  + Giao diện đơn giản: Giao diện người dùng (GUI) của các ứng dụng máy tính thời kỳ này thường đơn giản và dễ sử dụng.
  + Chức năng hạn chế: Các ứng dụng máy tính thời kỳ này thường có chức năng hạn chế so với các ứng dụng web hiện đại.
  + Cài đặt và gỡ cài đặt: Ứng dụng máy tính cần được cài đặt trên máy tính của người dùng trước khi sử dụng và có thể được gỡ cài đặt nếu không còn cần thiết.
* Lợi ích của Desktop Applications:
  + Hiệu suất cao: Ứng dụng máy tính thường có hiệu suất cao hơn so với các ứng dụng web, vì chúng không cần tải dữ liệu từ máy chủ.
  + Bảo mật: Ứng dụng máy tính được cài đặt trên máy tính của người dùng, giúp bảo vệ dữ liệu khỏi những kẻ tấn công bên ngoài.
  + Không phụ thuộc vào internet: Ứng dụng máy tính có thể hoạt động mà không cần kết nối internet.
* Tuy nhiên, Desktop Applications cũng có một số hạn chế:
  + Khả năng tương thích: Ứng dụng máy tính chỉ có thể hoạt động trên hệ điều hành và nền tảng cụ thể.
  + Cập nhật: Cập nhật ứng dụng máy tính thường phức tạp hơn so với cập nhật ứng dụng web.
  + Chi phí: Ứng dụng máy tính thường có giá cao hơn so với ứng dụng web.

## Thời kì web applications:

* Thời kỳ Web Applications (Ứng dụng Web) được đánh dấu bởi sự ra đời và phát triển mạnh mẽ của các ứng dụng web, thay thế cho các trang web tĩnh truyền thống. Nó bắt đầu từ khoảng đầu những năm 2000 và kéo dài đến nay, với nhiều giai đoạn phát triển và biến đổi.
* Một số mốc phát triển quan trong trong thời kỳ Web Applications:
  + Đầu những năm 2000: Xuất hiện các ứng dụng web đầu tiên như Gmail, Yahoo Mail, Hotmail.
  + Giữa những năm 2000: Phát triển mạnh mẽ của các ứng dụng web 2.0 như Facebook, Twitter, YouTube.
  + Cuối những năm 2000: Nổi lên các ứng dụng web di động (mobile web applications) cho phép người dùng truy cập trên điện thoại thông minh và máy tính bảng.
  + Hiện nay: Web Applications tiếp tục phát triển với tốc độ nhanh chóng, ứng dụng vào nhiều lĩnh vực như thương mại điện tử, giáo dục, y tế, v.v.
* Đặc điểm nổi bật của thời kỳ Web Applications:
  + Sự tương tác: Ứng dụng web cho phép người dùng tương tác trực tiếp với trang web, thay vì chỉ xem nội dung tĩnh.
  + Tính năng động: Ứng dụng web có thể cập nhật dữ liệu và thay đổi giao diện theo thời gian thực, mang lại trải nghiệm người dùng mượt mà và sống động hơn.
  + Tính phức tạp: Ứng dụng web có thể thực hiện nhiều chức năng phức tạp như xử lý dữ liệu, thanh toán trực tuyến, chat trực tiếp, v.v.
  + Khả năng tùy chỉnh: Ứng dụng web thường có thể được tùy chỉnh theo nhu cầu của người dùng, giúp nâng cao trải nghiệm người dùng.
* Lợi ích của Web Applications:
  + Tiết kiệm chi phí: Ứng dụng web không cần cài đặt phần mềm, có thể truy cập từ mọi nơi trên thế giới chỉ cần có kết nối internet.
  + Dễ dàng cập nhật: Ứng dụng web có thể được cập nhật dễ dàng và nhanh chóng mà không cần người dùng phải tải xuống phiên bản mới.
  + Khả năng mở rộng: Ứng dụng web có thể dễ dàng mở rộng để đáp ứng nhu cầu ngày càng tăng của người dùng.
  + Tính tương thích: Ứng dụng web có thể hoạt động trên nhiều trình duyệt web và thiết bị khác nhau.
* Tuy nhiên, Web Applications cũng có một số hạn chế:
  + Phụ thuộc vào kết nối internet: Ứng dụng web cần có kết nối internet để hoạt động.
  + Bảo mật: Ứng dụng web có thể dễ bị tấn công hơn so với phần mềm được cài đặt trên máy tính.
  + Hiệu suất: Hiệu suất của ứng dụng web có thể bị ảnh hưởng bởi tốc độ kết nối internet.

## Thời kỳ JavaScript

* Thời kỳ JavaScript bắt đầu từ khi ngôn ngữ lập trình JavaScript được phát triển vào năm 1995 bởi Brendan Eich tại Netscape. Kể từ đó, JavaScript đã trải qua nhiều giai đoạn phát triển và trở thành một trong những ngôn ngữ lập trình phổ biến nhất trên thế giới, đóng vai trò quan trọng trong việc xây dựng các trang web và ứng dụng web hiện đại.
* Một số mốc quan trọng trong thời kỳ JavaScript:
  + 1995: Ra đời JavaScript, ban đầu được gọi là LiveScript.
  + 1997: JavaScript được ECMA International chuẩn hóa thành ECMAScript.
  + 1999: Internet Explorer hỗ trợ JavaScript.
  + 2005: XMLHttpRequest (XHR) được giới thiệu, cho phép JavaScript truy cập dữ liệu từ máy chủ mà không cần tải lại trang.
  + 2006: jQuery ra đời, giúp đơn giản hóa việc viết mã JavaScript.
  + 2008: Node.js ra đời, cho phép JavaScript chạy trên máy chủ.
  + 2010: HTML5 và CSS3 được phát triển, mở ra nhiều khả năng mới cho JavaScript.
  + 2013: React ra đời, trở thành một trong những thư viện JavaScript phổ biến nhất.
  + 2015: Angular ra đời, một framework JavaScript khác được ưa chuộng.
  + 2016: Vue.js ra đời, một framework JavaScript nhẹ và linh hoạt.
  + Hiện nay: JavaScript tiếp tục phát triển với tốc độ nhanh chóng, được sử dụng để xây dựng các ứng dụng web phức tạp, single-page applications (SPA), mobile applications và nhiều hơn nữa.
* Lý do JavaScript trở nên phổ biến:
  + Dễ học và sử dụng: JavaScript có cú pháp tương đối đơn giản và dễ học, ngay cả với những người mới bắt đầu.
  + Linh hoạt: JavaScript có thể được sử dụng để thực hiện nhiều tác vụ khác nhau, từ thao tác DOM (Document Object Model) đến xử lý dữ liệu và tạo hiệu ứng đồ họa.
  + Cộng đồng lớn: JavaScript có cộng đồng người dùng và nhà phát triển lớn, giúp bạn dễ dàng tìm kiếm sự trợ giúp và tài nguyên khi cần thiết.
  + Hỗ trợ đa nền tảng: JavaScript được hỗ trợ bởi tất cả các trình duyệt web phổ biến, cho phép bạn xây dựng các ứng dụng web có thể hoạt động trên nhiều thiết bị khác nhau.
* Tác động của JavaScript:
  + Thay đổi cách thức hoạt động của web: JavaScript đã biến các trang web trở nên năng động và tương tác hơn, cho phép người dùng thực hiện nhiều tác vụ mà không cần tải lại trang.
  + Mở ra cánh cửa cho các ứng dụng web phức tạp: JavaScript đã giúp tạo ra các ứng dụng web phức tạp như Gmail, Facebook, Netflix và nhiều hơn nữa.
  + Thúc đẩy sự phát triển của các công nghệ mới: JavaScript đã thúc đẩy sự phát triển của các công nghệ mới như Node.js, React, Angular và Vue.js, mở ra nhiều khả năng mới cho các nhà phát triển web.

## Thời kỳ của CSS:

* Thời kỳ CSS bắt đầu từ khi ngôn ngữ CSS được phát triển vào năm 1996 bởi Håkon Wium Lie tại Netscape. Kể từ đó, CSS đã trải qua nhiều giai đoạn phát triển và trở thành một ngôn ngữ quan trọng trong việc thiết kế giao diện người dùng (UI) cho các trang web và ứng dụng web.
* Các mốc quan trong trong thời kì CSS:
  + 1996: Ra đời CSS, ban đầu được gọi là Cascading Style Sheets.
  + 1997: CSS được W3C (World Wide Web Consortium) chuẩn hóa thành CSS1.
  + 1999: CSS2 được W3C phát hành, bổ sung nhiều tính năng mới cho CSS.
  + 2000s: CSS được sử dụng rộng rãi hơn trong thiết kế web, giúp tạo ra các trang web đẹp mắt và dễ sử dụng hơn.
  + 2009: CSS3 được W3C phát hành, giới thiệu nhiều tính năng mới như bố cục flexbox, grid layout, media queries và animations.
  + 2010s: CSS preprocessors như Sass và LESS được sử dụng phổ biến, giúp đơn giản hóa việc viết mã CSS.
  + Hiện nay: CSS tiếp tục phát triển với tốc độ nhanh chóng, với nhiều tính năng mới được bổ sung và các công cụ hỗ trợ ngày càng tiên tiến.
* Lý do CSS trở nên phổ biến:
  + Dễ học và sử dụng: CSS có cú pháp tương đối đơn giản và dễ học, ngay cả với những người mới bắt đầu.
  + Linh hoạt: CSS có thể được sử dụng để tạo ra nhiều kiểu dáng khác nhau cho các trang web, từ bố cục trang đến màu sắc, phông chữ và hiệu ứng.
  + Tiết kiệm thời gian: CSS giúp tiết kiệm thời gian cho các nhà phát triển web bằng cách cho phép họ định dạng nhiều trang web cùng một lúc.
  + Tăng hiệu suất: CSS giúp cải thiện hiệu suất trang web bằng cách giảm thiểu số lượng mã HTML cần thiết để định dạng trang.
* Tác động của CSS:
  + Thay đổi cách thức thiết kế web: CSS đã thay đổi cách thức thiết kế web bằng cách tách biệt nội dung và giao diện của trang web.
  + Tạo ra các trang web đẹp mắt và dễ sử dụng: CSS giúp tạo ra các trang web đẹp mắt và dễ sử dụng hơn, mang lại trải nghiệm tốt hơn cho người dùng.
  + Mở ra cánh cửa cho các thiết kế web phức tạp: CSS đã mở ra cánh cửa cho các thiết kế web phức tạp hơn, với nhiều hiệu ứng và tương tác hơn.

## Cách mạng jQuery:

* jQuery, ra đời vào năm 2006, đánh dấu một bước ngoặt trong lịch sử phát triển front-end, mang đến cuộc cách mạng cho cách thức xây dựng và tương tác với các trang web. Trước jQuery, việc viết mã JavaScript cho các tác vụ thông thường như thao tác DOM, xử lý sự kiện và AJAX có thể tốn thời gian và phức tạp. jQuery xuất hiện như một giải pháp đơn giản và hiệu quả, giúp đơn giản hóa việc viết mã JavaScript và nâng cao năng suất của các nhà phát triển web.
* Tác động của jQuery đối với việc phát triển frontend:
  + Đơn giản hóa mã JavaScript: jQuery cung cấp một bộ API phong phú với các hàm được viết sẵn, giúp các nhà phát triển thực hiện các tác vụ JavaScript thông thường một cách nhanh nhất.
  + Tăng tốc độ phát triển: Việc sử dụng jQuery giúp tiết kiệm thời gian viết mã, cho phép các nhà phát triển xây dựng trang web nhanh hơn và hiệu quả hơn.
  + Tăng tính tương thích: jQuery hỗ trợ nhiều trình duyệt web khác nhau, giúp đảm bảo website hoạt động tốt trên nhiều thiết bị và nền tảng.
  + Tạo ra cộng đồng lớn: jQuery sở hữu cộng đồng người dùng và nhà phát triển rộng lớn, cung cấp nguồn tài nguyên phong phú và hỗ trợ tích cực.
* Lợi ích nổi bật của jQuery:
  + Dễ học và sử dụng: Cú pháp jQuery đơn giản và dễ hiểu, giúp người mới bắt đầu dễ dàng tiếp cận và sử dụng.
  + Linh hoạt và mạnh mẽ: jQuery cung cấp nhiều hàm và phương thức để thực hiện đa dạng các tác vụ front-end.
  + Hỗ trợ đa dạng: jQuery hỗ trợ nhiều trình duyệt web và thư viện JavaScript khác, giúp dễ dàng tích hợp với các dự án hiện có.
  + Cộng đồng lớn và hoạt động: Cộng đồng jQuery luôn sẵn sàng hỗ trợ và chia sẻ kiến thức, giúp giải quyết vấn đề và nâng cao kỹ năng.
* Một số hạn chế của jQuery:
  + Kích thước: jQuery có kích thước tương đối lớn, có thể ảnh hưởng đến tốc độ tải trang.
  + Có thể dẫn đến mã lộn xộn: Việc sử dụng jQuery quá mức có thể dẫn đến mã JavaScript khó đọc và bảo trì.
  + Cập nhật ít thường xuyên: jQuery không được cập nhật thường xuyên như các thư viện JavaScript hiện đại khác.

## Thời kỳ mobile applications:

* Thời kỳ Mobile Applications (Ứng dụng Di động) đánh dấu một bước ngoặt trong lịch sử công nghệ, thay đổi hoàn toàn cách thức chúng ta sử dụng điện thoại thông minh và máy tính bảng. Từ những ứng dụng đơn giản ban đầu, ứng dụng di động đã phát triển thành những công cụ mạnh mẽ, đa dạng, đóng vai trò quan trọng trong mọi khía cạnh của cuộc sống.
* Lịch sử phát triển:
  + Khởi đầu:
    - Cuối những năm 1990s: Xuất hiện những ứng dụng di động đầu tiên trên nền tảng Java ME, Symbian OS và BlackBerry OS.
    - 2007: Apple ra mắt iPhone và App Store, mở ra kỷ nguyên mới cho ứng dụng di động iOS.
    - 2008: Google ra mắt Android và Google Play Store, thúc đẩy sự phát triển mạnh mẽ của ứng dụng di động đa nền tảng.
  + Phát triển bùng nổ:
    - 2010s: Nhu cầu sử dụng ứng dụng di động tăng cao, dẫn đến sự bùng nổ về số lượng và chất lượng ứng dụng.
    - Xuất hiện nhiều công nghệ mới như HTML5, CSS3 và JavaScript, giúp việc phát triển ứng dụng di động trở nên dễ dàng và hiệu quả hơn.
    - Các cửa hàng ứng dụng chính thức như App Store và Google Play Store trở thành trung tâm phân phối ứng dụng di động lớn nhất thế giới.
  + Hiện tại và tương lai:
    - Ứng dụng di động đã trở thành một phần không thể thiếu trong cuộc sống của chúng ta, với hàng tỷ người sử dụng trên toàn thế giới.
    - Trí tuệ nhân tạo (AI), Machine Learning (ML), Thực tế ảo (VR), Thực tế tăng cường (AR), Blockchain,... được ứng dụng rộng rãi vào ứng dụng di động, mang đến những trải nghiệm mới mẻ và đột phá.
    - Xu hướng phát triển tập trung vào ứng dụng di động đa nền tảng, bảo mật cao, thân thiện với môi trường và hỗ trợ người khuyết tật.
* Tác động to lớn:
  + Thay đổi cách thức kết nối: Ứng dụng di động giúp chúng ta dễ dàng kết nối với bạn bè, gia đình và đồng nghiệp mọi lúc mọi nơi.
  + Mở ra cơ hội kinh doanh mới: Ứng dụng di động tạo ra nhiều cơ hội kinh doanh mới cho các doanh nghiệp, giúp tiếp cận khách hàng tiềm năng và tăng doanh thu.
  + Cải thiện chất lượng cuộc sống: Ứng dụng di động giúp cải thiện chất lượng cuộc sống trong nhiều lĩnh vực như y tế, giáo dục, giao thông vận tải,...
  + Thúc đẩy đổi mới sáng tạo: Ứng dụng di động là nền tảng cho nhiều ý tưởng sáng tạo mới, thúc đẩy sự phát triển của khoa học kỹ thuật.
* Một số thách thức:

Bảo mật và quyền riêng tư: Việc sử dụng ứng dụng di động tiềm ẩn nguy cơ bị đánh cắp dữ liệu cá nhân và thông tin nhạy cảm.

Nghiện ứng dụng di động: Việc sử dụng ứng dụng di động quá mức có thể ảnh hưởng đến sức khỏe và tinh thần của người dùng.

Bất bình đẳng kỹ thuật số: Không phải ai cũng có thể tiếp cận và sử dụng ứng dụng di động do hạn chế về cơ sở hạ tầng và khả năng tài chính.

## Thời kỳ cross platform applications:

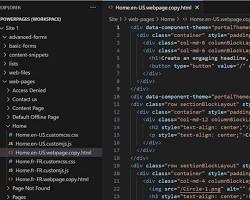
* Ứng dụng Cross-Platform (Ứng dụng đa nền tảng) đang ngày càng trở nên phổ biến trong thời đại ngày nay bởi những ưu điểm vượt trội so với các ứng dụng Native truyền thống. Dưới đây là một số lý do chính cho sự phát triển mạnh mẽ của xu hướng này:
  + Tiết kiệm chi phí và thời gian phát triển:
    - Việc phát triển một ứng dụng Native cho mỗi hệ điều hành riêng biệt đòi hỏi nhiều nguồn lực về nhân lực, tài chính và thời gian. Ứng dụng Cross-Platform giúp giải quyết vấn đề này bằng cách sử dụng một bộ mã duy nhất để xây dựng ứng dụng cho nhiều nền tảng khác nhau, từ đó tiết kiệm chi phí và thời gian phát triển đáng kể.
  + Mở rộng thị trường tiềm năng:
    - Thị trường di động ngày nay vô cùng đa dạng với nhiều hệ điều hành phổ biến như Android, iOS, Windows Phone, v.v. Việc phát triển ứng dụng Cross-Platform giúp doanh nghiệp tiếp cận được với lượng lớn người dùng trên nhiều nền tảng khác nhau, mở rộng thị trường tiềm năng và gia tăng cơ hội kinh doanh.
  + Dễ dàng bảo trì và cập nhật:
    - Việc quản lý và cập nhật một bộ mã duy nhất cho nhiều nền tảng sẽ dễ dàng và hiệu quả hơn so với việc quản lý nhiều phiên bản Native riêng biệt. Điều này giúp doanh nghiệp tiết kiệm thời gian và nguồn lực cho công tác bảo trì và cập nhật ứng dụng.
  + Nâng cao trải nghiệm người dùng:
    - Ứng dụng Cross-Platform được thiết kế để mang lại trải nghiệm người dùng nhất quán trên mọi nền tảng, đảm bảo sự mượt mà và tiện lợi cho người dùng bất kể họ sử dụng thiết bị nào.
  + Các công nghệ Cross-Platform tiên tiến:
    - Sự phát triển mạnh mẽ của các công nghệ Cross-Platform như React Native, Flutter, Xamarin, v.v. đã giúp việc xây dựng ứng dụng Cross-Platform trở nên dễ dàng và hiệu quả hơn bao giờ hết. Các công nghệ này cung cấp nhiều tính năng và khả năng mạnh mẽ, giúp các nhà phát triển tạo ra những ứng dụng đa nền tảng chất lượng cao.
  1. **Thời kì progressive web application**
* Progressive Web Applications (PWA) là một công nghệ web mới nổi đang nhanh chóng thu hút sự chú ý của giới lập trình viên và doanh nghiệp. PWA được xem là sự kết hợp hoàn hảo giữa website và ứng dụng di động, mang đến cho người dùng trải nghiệm mượt mà và tiện lợi như ứng dụng Native mà không cần tải xuống.
* Lý do cho sự bùng nổ của PWA:
  + Nâng cao trải nghiệm người dùng: PWA mang đến tốc độ tải trang nhanh chóng, khả năng hoạt động ngoại tuyến, khả năng đẩy thông báo và giao diện người dùng thân thiện, từ đó mang lại trải nghiệm mượt mà và tiện lợi cho người dùng.
  + Tiết kiệm chi phí phát triển: Do sử dụng công nghệ web nên PWA có thể được phát triển với chi phí thấp hơn so với ứng dụng Native.
  + Dễ dàng bảo trì và cập nhật: Việc quản lý và cập nhật một bộ mã duy nhất cho PWA sẽ dễ dàng và hiệu quả hơn so với việc quản lý nhiều phiên bản Native riêng biệt.
  + Tương thích với nhiều nền tảng: PWA có thể hoạt động trên mọi trình duyệt web và thiết bị di động, giúp doanh nghiệp tiếp cận được với lượng lớn người dùng tiềm năng.
  + SEO tốt hơn: PWA được tối ưu hóa cho công cụ tìm kiếm, giúp doanh nghiệp dễ dàng tiếp cận khách hàng tiềm năng hơn.
* Xu hướng tương lai:
  + Với những ưu điểm vượt trội, PWA được dự đoán sẽ trở thành xu hướng chủ đạo trong phát triển web trong tương lai. Các công nghệ liên quan đến PWA sẽ ngày càng được cải tiến và hoàn thiện, giúp việc xây dựng PWA trở nên dễ dàng, hiệu quả và tiết kiệm chi phí hơn. Doanh nghiệp sẽ ngày càng ưa chuộng sử dụng PWA để mang lại trải nghiệm tốt nhất cho người dùng và tăng cường khả năng cạnh tranh trên thị trường.
  1. **Thời kỳ của responsive design**
* Thiết kế Phản hồi (Responsive Design) đã và đang định hình thế giới web trong hơn một thập kỷ qua, trở thành tiêu chuẩn thiết yếu cho các trang web hiện đại. Xu hướng này mang đến những lợi ích to lớn cho cả người dùng và doanh nghiệp, dẫn đến sự áp dụng rộng rãi và ngày càng phổ biến.
* Lý do bùng nổ của Thiết kế Phản hồi:
  + Sự gia tăng truy cập web từ thiết bị di động: Nhu cầu truy cập web từ điện thoại thông minh và máy tính bảng không ngừng gia tăng, đòi hỏi các trang web phải thích ứng linh hoạt với nhiều kích thước màn hình khác nhau. Thiết kế Phản hồi đáp ứng hoàn hảo nhu cầu này, đảm bảo trải nghiệm người dùng nhất quán trên mọi thiết bị.
  + Tối ưu hóa trải nghiệm người dùng: Thiết kế Phản hồi chú trọng vào việc tạo ra giao diện dễ sử dụng, điều hướng mượt mà và hiển thị nội dung phù hợp trên mọi kích thước màn hình. Nhờ vậy, người dùng có thể truy cập và tương tác với trang web một cách dễ dàng và hiệu quả, bất kể họ sử dụng thiết bị nào.
  + Tiết kiệm chi phí và thời gian phát triển: Thay vì phải tạo ra các trang web riêng biệt cho từng thiết bị, Thiết kế Phản hồi cho phép xây dựng một trang web duy nhất có thể thích ứng với mọi kích thước màn hình. Điều này giúp tiết kiệm đáng kể chi phí và thời gian phát triển, quản lý và bảo trì trang web.
  + Tăng khả năng SEO: Google và các công cụ tìm kiếm khác ưu tiên các trang web thân thiện với thiết bị di động. Thiết kế Phản hồi giúp tối ưu hóa trang web cho SEO, nâng cao thứ hạng tìm kiếm và thu hút nhiều truy cập hơn.
  + Nâng cao hình ảnh thương hiệu: Thiết kế Phản hồi thể hiện sự chuyên nghiệp và uy tín của doanh nghiệp, mang đến hình ảnh thương hiệu nhất quán và ấn tượng trên mọi thiết bị.
* Xu hướng tương lai:

Thiết kế Phản hồi sẽ tiếp tục phát triển và hoàn thiện trong tương lai, song hành cùng sự phát triển của công nghệ web và nhu cầu ngày càng cao của người dùng. Một số xu hướng nổi bật bao gồm:

* + Thiết kế đáp ứng linh hoạt: Các trang web sẽ có khả năng thích ứng với nhiều kích thước màn hình hơn, từ màn hình lớn của máy tính để bàn đến màn hình gập và thiết bị đeo.
  + Trải nghiệm người dùng được cá nhân hóa: Thiết kế Phản hồi sẽ kết hợp với dữ liệu người dùng để cá nhân hóa trải nghiệm, cung cấp nội dung và chức năng phù hợp với từng cá nhân.
  + Sử dụng trí tuệ nhân tạo (AI): AI sẽ được ứng dụng để tự động tối ưu hóa giao diện và trải nghiệm người dùng cho từng thiết bị và ngữ cảnh truy cập.
  + Sự phát triển của các công nghệ web mới: Thiết kế Phản hồi sẽ tích hợp các công nghệ web mới như WebAssembly và Progressive Web Apps (PWA) để nâng cao hiệu suất và khả năng của trang web.

1. **Công nghệ frontend**
   1. **Các công cụ**

* Trình chỉnh sửa mã: Trình chỉnh sửa mã là cần thiết để viết mã HTML, CSS và JavaScript. Một số trình chỉnh sửa mã phổ biến bao gồm Visual Studio Code, Sublime Text và Atom.



* Công cụ gỡ lỗi: Công cụ gỡ lỗi cho phép bạn gỡ lỗi mã của mình và xác định các vấn đề. Một số công cụ gỡ lỗi phổ biến bao gồm Chrome DevTools, Firefox Developer Tools và Safari Web Inspector.

A screenshot of a computer

Description automatically generated

* Công cụ kiểm tra: Công cụ kiểm tra cho phép bạn kiểm tra trang web của mình trên các trình duyệt và thiết bị khác nhau. Một số công cụ kiểm tra phổ biến bao gồm BrowserStack, Sauce Labs và Selenium.
* Công cụ xử lý trước: Công cụ xử lý trước cho phép bạn xử lý mã của mình trước khi nó được trình duyệt tải. Điều này có thể hữu ích cho các tác vụ như tổng hợp LESS và SCSS thành CSS và nén tệp JavaScript. Một số công cụ xử lý trước phổ biến bao gồm Gulp, Grunt và Webpack.
* Công cụ quản lý gói: Công cụ quản lý gói cho phép bạn quản lý các phụ thuộc của dự án của mình. Một số công cụ quản lý gói phổ biến bao gồm npm và yarn.



* Công cụ kiểm soát phiên bản: Công cụ kiểm soát phiên bản cho phép bạn theo dõi thay đổi đối với mã của mình. Một số công cụ kiểm soát phiên bản phổ biến bao gồm Git và Mercurial.



* 1. **Các ngôn ngữ lập trình**
* HTML (HyperText Markup Language): Là ngôn ngữ nền tảng để tạo cấu trúc và nội dung cho trang web. HTML sử dụng các thẻ để định nghĩa các phần khác nhau của trang như tiêu đề, đoạn văn bản, hình ảnh, liên kết, v.v.



* CSS (Cascading Style Sheets): Được sử dụng để định dạng giao diện của trang web, bao gồm màu sắc, bố cục, kiểu chữ, v.v. CSS giúp bạn tạo ra các trang web đẹp mắt và thu hút người dùng.



* JavaScript: Mang đến khả năng tương tác cho trang web, cho phép bạn tạo các hiệu ứng động, xử lý sự kiện người dùng, thao tác với dữ liệu và giao tiếp với máy chủ. JavaScript đóng vai trò quan trọng trong việc tạo ra các trang web và ứng dụng web hiện đại, mượt mà và năng động.

A yellow sign with a black text

Description automatically generated

1. **Các frontend framework**
   1. **Định nghĩa**

* Frontend frameword là một bộ công cụ, thư viện và quy ước được thiết kế để hỗ trợ việc xây dựng các giao diện người dùng (UI) và ứng dụng web hiện đại. Framework front-end cung cấp một cấu trúc cơ bản và các tính năng sẵn có, giúp các nhà phát triển tiết kiệm thời gian và công sức, đồng thời tạo ra các ứng dụng web chất lượng cao, hiệu quả và dễ dàng bảo trì.
  1. **Các framework frontend phổ biến**
* React:
  + Được Facebook phát triển và ưa chuộng bởi sự linh hoạt, hiệu quả và dễ học.
  + Sử dụng mô hình component-based (dựa trên thành phần), giúp tái sử dụng code dễ dàng và bảo trì ứng dụng hiệu quả.
  + Cộng đồng người dùng lớn và nhiều tài liệu hướng dẫn, hỗ trợ đắc lực cho người mới bắt đầu.
  + Thư viện mở rộng phong phú, đáp ứng đa dạng nhu cầu phát triển.
* Vue.js:
  + Framework JavaScript nhẹ và dễ học, nổi tiếng với cú pháp đơn giản và trực quan.
  + Cung cấp nhiều tính năng mạnh mẽ để xây dựng các ứng dụng web hiện đại.
  + Tính linh hoạt cao, dễ dàng tích hợp với các thư viện và công cụ khác.
  + Cộng đồng người dùng đang phát triển nhanh chóng và có nhiều tài liệu hướng dẫn hữu ích.
* Angular:
  + Framework do Google phát triển, cung cấp bộ công cụ toàn diện để xây dựng ứng dụng web một trang (SPA).
  + Cấu trúc rõ ràng, ổn định và có khả năng mở rộng cao, phù hợp cho các dự án lớn và phức tạp.
  + Sử dụng TypeScript, giúp viết code an toàn và dễ bảo trì hơn.
  + Cộng đồng người dùng lớn và nhiều tài liệu hướng dẫn chi tiết.
* Bootstrap:
  + Framework CSS phổ biến để xây dựng giao diện người dùng responsive (phản hồi).
  + Cung cấp nhiều thành phần UI đã được tạo sẵn như nút, biểu mẫu, bảng, v.v., giúp dễ dàng tạo ra các trang web đẹp mắt và dễ sử dụng trên mọi thiết bị.
  + Dễ học và sử dụng, phù hợp cho cả người mới bắt đầu và lập trình viên front-end.
  + Cộng đồng người dùng lớn và nhiều tài liệu hướng dẫn chi tiết.
* Ngoài ra, còn có nhiều framework front-end khác cũng được sử dụng rộng rãi như:
  + jQuery: Thư viện JavaScript phổ biến để thao tác với DOM (Document Object Model), xử lý sự kiện và thực hiện các tác vụ AJAX.
  + Ember.js: Framework JavaScript mạnh mẽ để xây dựng các ứng dụng web một trang phức tạp.
  + Backbone.js: Framework JavaScript nhẹ và linh hoạt để xây dựng các ứng dụng web single-page (SPA).
  + Semantic UI: Framework CSS dựa trên Bootstrap, cung cấp nhiều thành phần UI đẹp mắt và dễ sử dụng.
  + Foundation: Framework CSS responsive (phản hồi) khác, cung cấp nhiều tính năng mạnh mẽ để xây dựng các giao diện người dùng hiện đại.