**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA CÔNG NGHỆ PHẦN MỀM**



Logo

Description automatically generated

**BÁO CÁO ĐỒ ÁN**

**XÂY DỰNG WEBSITE VIDEO CALL, CHAT REAL-TIME**

**Giảng viên hướng dẫn : Nguyễn Tấn Toàn**

**Sinh viên thực hiện 1 : Lương Quang Huy - 20521398**

**Sinh viên thực hiện 2 : Nguyễn Khoa Hiếu - 20521324**

**Lớp : SE400.M21**

**Thành phố Hồ Chí Minh, tháng 12 năm 2023**

# NHẬN XÉT CỦA GIẢNG VIÊN

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………...…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

………………………………………………………………………………………

Ngày tháng 07 năm 2023

**Giảng viên**

(Ký tên và ghi rõ họ tên )

# DANH SÁCH BẢNG BIỂU, HÌNH ẢNH

Biểu đồ 5.1 Tần suất sử dụng website quản lý trường học của học sinh và sinh viên 56

Biểu đồ 5.2 Khảo sát nhìn chung về SEQ 58

Hình 1.1 Hình minh họa SIP (Session Initiation Protocol) 18

Hình 1.2 Hình logo WebSockets 19

Hình 2.1 Hình logo ReactJS 21

Hình 2.2 Hình logo NextJS 25

Hình 2.3 Hình logo MySQL 28

Hình 2.4 Hình kiến trúc SignalR 31

Hình 2.5 Hình logo Docker 32

Hình 3.1 Lịch sử hình thành WebRTC 34

Hình 3.2 Kiến trúc WebRTC 36

Hình 3.3 Kiến trúc WebRTC 37

Hình 3.4 Cấu trúc WebRTC 38

Hình 3.5 Các tầng giao thức trong WebRTC 39

Hình 3.6 Giới hạn đa nhiệm trình duyệt của WebRTC 43

Hình 3.7 Giao thức Peer Connection 44

Hình 4.1 Sơ đồ use case 47

Hình 4.2 Khóa chính – Khóa ngoại trong database 50

**MỤC LỤC**

[NHẬN XÉT CỦA GIẢNG VIÊN 2](#_Toc139108718)

[DANH SÁCH BẢNG BIỂU, HÌNH ẢNH 3](#_Toc139108719)

[MỤC LỤC 4](#_Toc139108720)

[ĐỀ CƯƠNG CHI TIẾT 6](#_Toc139108721)

[Chương 1. Tổng quan 14](#_Toc139108722)

[1.1. Lý do chọn đề tài 14](#_Toc139108723)

[1.2. Mục tiêu 14](#_Toc139108724)

[1.3. Phạm vi 15](#_Toc139108725)

[1.4. Người dùng 15](#_Toc139108726)

[1.5. Phương pháp nghiên cứu 15](#_Toc139108727)

[1.6. Khảo sát, đánh giá các công nghệ cạnh tranh 15](#_Toc139108728)

[1.6.1. WebRTC 16](#_Toc139108729)

[1.6.2. Công nghệ SIP (Session Initiation Protocol) 17](#_Toc139108730)

[1.6.3. WebSockets 18](#_Toc139108731)

[Chương 2. Cơ sở lý thuyết và Công nghệ 20](#_Toc139108732)

[2.1. Giới thiệu về ReactJS 20](#_Toc139108733)

[2.1.1. Khái niệm: 20](#_Toc139108734)

[2.1.2. Ưu - nhược điểm: 23](#_Toc139108735)

[2.2. Giới thiệu về NextJS 24](#_Toc139108736)

[2.2.1. Các đặc điểm cơ bản của NextJs 25](#_Toc139108737)

[2.2.2. Ưu nhược điểm của NextJS: 25](#_Toc139108738)

[2.3. Giới thiệu về MySQL 27](#_Toc139108739)

[2.3.1. Các đặc điểm cơ bản của MySQL 28](#_Toc139108740)

[2.3.2. Ưu nhược điểm của MySQL: 28](#_Toc139108741)

[2.4. Giới thiệu về ASP.NET SignalR 30](#_Toc139108742)

[2.3. Giới thiệu về Docker 31](#_Toc139108743)

[Chương 3. Phân tích kiến trúc hệ thống 33](#_Toc139108744)

[3.1. Tổng quan WebRTC 33](#_Toc139108745)

[3.1.1. Quá trình phát triển 33](#_Toc139108746)

[3.1.2. Sự hỗ trợ từ trình duyệt 34](#_Toc139108747)

[3.1.3. Kiến trúc WebRTC trong trình duyệt 35](#_Toc139108748)

[3.1.4. Các APIs trong Web 37](#_Toc139108749)

[3.1.5. Các tầng giao thức trong WebRTC 38](#_Toc139108750)

[3.1.6. Ưu và nhược điểm của WebRTC 41](#_Toc139108751)

[3.2. Báo hiệu trong WebRTC 42](#_Toc139108752)

[3.2.1. Vai trò của báo hiệu 42](#_Toc139108753)

[3.2.2. Giao thức vận chuyển báo hiệu 43](#_Toc139108754)

[3.2.3. Giao thức báo hiệu 44](#_Toc139108755)

[3.2.4. Các quá trình trong báo hiệu 45](#_Toc139108756)

[Chương 4. Thiết kế hệ thống 46](#_Toc139108757)

[4.1. Thiết kế use case 46](#_Toc139108758)

[4.1.2. Danh sách các actor 46](#_Toc139108759)

[Tên Actor 46](#_Toc139108760)

[Mô tả 46](#_Toc139108761)

[Người dùng 46](#_Toc139108762)

[Người dùng đăng nhập vào hệ thống vào các phòng của hệ thống để sử dụng các tính năng như gọi video âm thanh,chia sẽ màn hình và truyền dữ liệu trong thời gian thực 46](#_Toc139108763)

[4.1.3. Danh sách các use case 46](#_Toc139108764)

[4.1.4. Đặc tả use case 46](#_Toc139108765)

[4.2. Thiết kế cơ sở dữ liệu 49](#_Toc139108766)

[4.2.1. Sơ đồ Database 49](#_Toc139108767)

[4.2.2. Mô tả các bảng 49](#_Toc139108768)

[Chương 5. Preference Test Report 51](#_Toc139108769)

[5.1. Study Design 51](#_Toc139108770)

[5.2. Methods 52](#_Toc139108771)

[5.3. Đối tượng 52](#_Toc139108772)

[5.4. Kết quả 53](#_Toc139108773)

[5.5. Sự hiểu 55](#_Toc139108774)

[Chương 6. Kết luận và hướng phát triển 58](#_Toc139108775)

[6.1. Kết quả đạt được 58](#_Toc139108776)

[6.2. Một số hướng phát triển 58](#_Toc139108777)

[TÀI LIỆU THAM KHẢO 60](#_Toc139108778)

|  |  |
| --- | --- |
| **TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**  **KHOA CÔNG NGHỆ PHẦN MỀM** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh phúc** |

# ĐỀ CƯƠNG CHI TIẾT

|  |
| --- |
| **TÊN ĐỀ TÀI:**  **Tìm hiểu Video Streaming và xây dựng website video call, chat real-time bằng Video Streaming** |
| **Tên đề tài tiếng Anh:**  LEARN Video Streaming AND BUILD REAL-TIME VIDEO CALL AND CHAT WEBSITE |
| **Cán bộ hướng dẫn:** ThS. Nguyễn Tấn Toàn |
| **Thời gian thực hiện:** Từ ngày 21/09/2023 đến ngày 06/1/2024 |
| **Sinh viên thực hiện:**  Nguyễn Khoa Hiếu-20521324  Lương Quang Huy-20521398 |
| **Nội dung đề tài:**  **1. Giới thiệu:**   * Đề tài "Xây dựng website video call, chat real-time" là một đề tài hấp dẫn và đang rất được quan tâm trong thời đại số hóa hiện nay.Web là một công nghệ cung cấp các giải pháp liên quan đến truyền thông trực tuyến, cho phép người dùng có thể truyền tải âm thanh, hình ảnh và dữ liệu thông qua internet một cách nhanh chóng và dễ dàng. * Đề tài này nhằm mục đích giúp sinh viên tìm hiểu về công nghệ Video Streaming, từ đó áp dụng kiến thức để xây dựng một website video call và chat real-time sử dụng công nghệ này. Để thực hiện được đề tài này, sinh viên cần phải hiểu về các nguyên tắc hoạt động của Video Streaming, cách xây dựng ứng dụng truyền thông trực tuyến và các vấn đề liên quan đến bảo mật và quản lý kết nối. * Sinh viên sẽ được yêu cầu thực hiện việc nghiên cứu các công nghệ liên quan đến Video Streaming, tìm hiểu về cách xây dựng một ứng dụng truyền thông trực tuyến, phát triển các tính năng video call và chat real-time trên trình duyệt web sử dụng Video Streaming. Sau đó, sinh viên sẽ phải thực hiện việc kiểm thử và đánh giá hiệu suất của ứng dụng này. * Với đề tài này, sinh viên sẽ có cơ hội phát triển các kỹ năng lập trình, tìm hiểu và nghiên cứu công nghệ mới, thực hiện kiểm thử và đánh giá hiệu suất của ứng dụng, và trình bày kết quả nghiên cứu của mình một cách chuyên nghiệp. Ngoài ra, đề tài này cũng mang tính ứng dụng cao, đáp ứng nhu cầu thực tế của người dùng trong việc kết nối và giao tiếp trực tuyến. * Website video call và chat real-time bằng Video Streaming có nhiều ưu điểm, bao gồm: * Khả năng truyền tải âm thanh và hình ảnh trực tuyến nhanh chóng và chất lượng cao: Với Video Streaming, người dùng có thể thực hiện video call và chat real-time một cách nhanh chóng và chất lượng hình ảnh và âm thanh được truyền tải mượt mà, giúp cho việc giao tiếp trở nên dễ dàng hơn. * Tính năng kết nối trực tiếp giữa các thiết bị: Video Streaming sử dụng kết nối trực tiếp (peer-to-peer) giữa các thiết bị, giúp cho việc truyền tải dữ liệu trực tuyến nhanh hơn và giảm thiểu độ trễ (latency). * Độ bảo mật cao: Video Streaming sử dụng các giao thức mã hóa thông tin đảm bảo tính an toàn và bảo mật của dữ liệu được truyền tải. * Hỗ trợ đa nền tảng: Video Streaming được hỗ trợ trên nhiều nền tảng khác nhau bao gồm trên trình duyệt web, điện thoại di động và máy tính bảng. * Tính sẵn có và dễ dàng triển khai: Với sự phát triển nhanh chóng của công nghệ, Video Streaming hiện nay đã trở thành một công nghệ phổ biến và được hỗ trợ rộng rãi trên các trình duyệt web phổ biến như Google Chrome, Firefox, Safari và Opera. * Hiện nay có nhiều website phổ biến tại Việt Nam đã sử dụng công nghệ Video Streaming để cung cấp các tính năng video call và chat real-time: Zalo, VNG, Garena, ViettelPay, FPT Play, … * Website sử dụng công nghệ Video Streaming hiện nay đang có tiềm năng phát triển và phổ biến hơn trong tương lai. Có nhiều lý do để tin rằng công nghệ này sẽ tiếp tục phát triển và được ứng dụng rộng rãi, nhóm em mong muốn tự tạo nên một website video call, chat real-time bằng Video Streaming của riêng mình, đó là lý do chúng em chọn đề tài này.  1. **Mục tiêu:**     * Xây dựng được một website website video call, chat real-time bằng Video Streaming đáp ứng các tiêu chí:  * Tính năng: Trang web cần phải đáp ứng được các tính năng cơ bản như tạo cuộc gọi video, chat real-time, chia sẻ tệp tin, gửi tin nhắn, điều khiển âm lượng và camera, và các tính năng khác tùy thuộc vào yêu cầu của người dùng. * Giao diện người dùng thân thiện: Giao diện trang web cần đơn giản, dễ sử dụng và thân thiện với người dùng. Thông tin phải được sắp xếp hợp lý và các tính năng phải được đặt ở vị trí dễ tiếp cận. * Tốc độ và độ trễ: Trang web cần phải được tối ưu hóa để đảm bảo tốc độ nhanh và độ trễ thấp cho các cuộc gọi video và chat real-time.  1. **Phạm vi:**     * **Phạm vi môi trường:** Triển khai sản phẩm đề tài trên môi trường web.    * **Phạm vi chức năng:**  * Truyền tải audio và video: Video Streaming cho phép truyền tải audio và video giữa các thiết bị mà không cần phải cài đặt phần mềm bổ sung hay plug-in. * Giao tiếp thời gian thực: Video Streaming cho phép truyền tải thông tin giữa các thiết bị một cách thời gian thực, giúp tạo ra trải nghiệm truyền thông tốt hơn. * Truyền tải dữ liệu: Video Streaming cũng cho phép truyền tải dữ liệu khác nhau như tin nhắn * Điều chỉnh chất lượng và độ phân giải: Video Streaming cho phép điều chỉnh chất lượng và độ phân giải của audio và video để phù hợp với các điều kiện mạng và thiết bị sử dụng. * Chia sẻ màn hình: Video Streaming cung cấp tính năng chia sẻ màn hình giữa các thiết bị, giúp cho việc trình chiếu bài giảng, thuyết trình, hoặc chỉ đường trở nên dễ dàng hơn.  1. **Đối tượng:**  * Được áp dụng rộng rãi trong môi trường giáo dục hiện nay. * Các doanh nghiêp: Video Streaming cho phép các doanh nghiệp triển khai các giải pháp truyền thông thời gian thực cho các cuộc họp trực tuyến, hội nghị và hợp tác từ xa.   Trong đời sống hằng ngày để gặp mặt nhau qua các phương tiện mà không cần phải di chuyển xa.   1. **Phương pháp thực hiện:**     * Trải nghiệm các Video Streaming hiện có, từ đó phân tích, xác định các tính năng cần triển khai.    * Lập kế hoạch, phân công công việc cho các thành viên.    * Tìm hiểu công nghệ sẽ sử dụng.    * Triển khai xây dựng website theo đúng kế hoạch đã đề ra.    * Kiểm thử, phát hiện và sửa lỗi (nếu có).    * Báo cáo đồ án. 2. **Công nghệ:**     * Front-end: HTML5, CSS3, JQuery , ReactJS    * Back-end: Supabase    * Database: Supabase 3. **Kết quả mong đợi**     * Xây dựng được một Video Streaming với đầy đủ chức năng cơ bản.    * Giao diện thân thiện, hiện đại, dễ sử dụng, có trải nghiệm người dùng tốt.    * Nắm bắt được các kiến thức cần có trong quy trình xây dựng một website.    * Có tính tiến hoá: có thể phát triển, mở rộng thêm các tính năng mới trong tương lai.   Quá trình thực hiện đạt đúng tiến độ của môn học. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Kế hoạch thực hiện:**     |  |  |  | | --- | --- | --- | | **Thời gian** | **Công việc thực hiện** | **Thành viên thực hiện** | | Từ 21/09/2023 đến  13/09/2023 | Tìm hiểu đề tài, phân tích yêu cầu, xác định các tính năng cần có trong sản phẩm. | Cả 2 thành viên | | Từ 14/09/2023 đến  27/09/2023 | Tìm hiểu về công nghệ sử dụng: HTML,  CSS, JQUERY, JAVASCRIPT, REACTJS,NODEJS | Cả 2 thành viên | | Từ 27/9/2023 đến  10/10/2023 | Thiết kế giao diện | Cả 2 thành viên | | Từ 10/10/2023 đến  10/11/2023 | Thiết kế Backend | Cả 2 thành viên | | 10/11/2023 đến 6/1/2024 | Phát triển các tính năng app | Cả 2 thành viên | |  |  |  | | |
| **Xác nhận của CBHD**  (Ký tên và ghi rõ họ tên)          **ThS. Ngyuyễn Tấn Toàn** | **TP. HCM, ngày 06 tháng 1 năm 2024**  **Sinh viên**  (Ký tên và ghi rõ họ tên)           |  |  | | --- | --- | | **Nguyễn Khoa Hiếu** | **Lương Quang Huy** | |

# Chương 1. Tổng quan

## Lý do chọn đề tài

Ngày nay, ứng dụng công nghệ thông tin và việc tin học hóa được xem là một trong những yếu tố mang tính quyết định trong hoạt động của các chính phủ, tổ chức, cũng như các công ty, cửa hàng, nó đóng vai trò hết sức quan trọng có thể tạo ra những bước đột phá mạnh mẽ.

Nhu cầu sử dụng Video Streaming đang tăng: Với sự phát triển của công nghệ truyền thông thời gian thực, nhu cầu sử dụng Video Streaming để xây dựng các ứng dụng truyền thông tăng lên đáng kể. Do đó, việc thực hiện đồ án về Video Streaming sẽ giúp bạn hiểu rõ hơn về công nghệ này và cũng cung cấp cho bạn cơ hội để phát triển các kỹ năng kỹ thuật cần thiết để xây dựng các ứng dụng truyền thông thời gian thực.

Hiện nay nhu cầu mọi người học tập và làm việc ở nhà đang trở nên ngày càng nhiều hơn nhất là sau đại dịch Covid-19 thì nhu cầu làm việc và học tập online ngày càng nhiều vì vậy chúng em muốn tạo ra đồ án này để giải quyết các vấn đề trên.

* ***Tính ứng dụng thực tiễn cao:*** Việc xây dựng một ứng dụng video call và chat real-time sử dụng Video Streaming có tính ứng dụng thực tiễn cao. Bạn có thể sử dụng ứng dụng của mình để kết nối với người thân, bạn bè, đồng nghiệp, v.v. Mặt khác, ứng dụng của bạn cũng có thể được ứng dụng trong các công việc trực tuyến như học tập trực tuyến, làm việc từ xa, v.v.

## Mục tiêu

* + Hiểu rõ hơn về công nghệ Video Streaming: Nắm vững các khái niệm cơ bản và kiến thức liên quan đến Video Streaming như đặc điểm, ứng dụng, lịch sử phát triển, cơ chế hoạt động, v.v.
  + Xây dựng ứng dụng video call và chat real-time: Sử dụng các kỹ thuật lập trình web và Video Streaming để xây dựng một ứng dụng truyền thông thời gian thực cho phép người dùng kết nối với nhau thông qua video call và chat.
  + Đảm bảo tính ổn định và bảo mật của ứng dụng: Đảm bảo tính ổn định và bảo mật của ứng dụng truyền thông thời gian thực, bao gồm các chức năng liên quan đến mã hóa dữ liệu, xác thực người dùng, chống tấn công, v.v.
  + Kiểm tra và đánh giá hiệu suất của ứng dụng: Kiểm tra và đánh giá hiệu suất của ứng dụng truyền thông thời gian thực, bao gồm độ trễ, tốc độ truyền tải, độ phân giải video, v.v.

## Phạm vi

Phạm vi của đồ án là tập trung vào xây dựng một ứng dụng video call và chat real-time sử dụng công nghệ Video Streaming. Đồ án sẽ tập trung vào các yêu cầu chức năng cơ bản của ứng dụng như kết nối video call và chat, tạo, gia nhập và quản lý các phòng trò chuyện, mã hóa và giải mã dữ liệu, v.v.

## Người dùng

Đối tượng sử dụng của ứng dụng video call và chat real-time sử dụng Video Streaming có thể là bất kỳ ai có nhu cầu giao tiếp trực tuyến bằng video hoặc chat, như những người dùng cá nhân, các doanh nghiệp, tổ chức, trường học hoặc bất kỳ tổ chức nào cần liên lạc trực tuyến.

Tuy nhiên, đối tượng sử dụng cụ thể của đồ án có thể tập trung vào các công ty, tổ chức hoặc cá nhân có nhu cầu triển khai một ứng dụng video call và chat real-time trên nền tảng web của họ. Đồ án có thể cung cấp cho họ một nền tảng tùy chỉnh để triển khai một ứng dụng tương tự hoặc sử dụng đồ án như một tham khảo để phát triển các ứng dụng tương tự trong tương lai.

## Phương pháp nghiên cứu

* + Phương pháp hệ thống, phương pháp tư duy.
  + Phương pháp phân tích, tổng hợp.
  + Phương pháp tin học hóa bằng công cụ lập trình để giải quyết vấn đề.

## Khảo sát, đánh giá các công nghệ cạnh tranh

* + Phương pháp: Dựa trên các công nghệ cạnh tranh trực tiếp trên thị trường.
  + Trên toàn cầu, Video Streaming được sử dụng rộng rãi trong các ứng dụng gọi video và chat real-time trên web, bao gồm cả các ứng dụng như Google Meet, Skype và Facebook Messenger.
  + Các đối thủ cạnh tranh của Video Streaming trên toàn cầu bao gồm các công nghệ như SIP (Session Initiation Protocol), WebSockets, WebSIP và các nền tảng cloud-based như TokBox và Twilio.

**1.6.1. WebRTC**

Thị phần của công nghệ Video Streaming đang tăng trưởng nhanh chóng trong những năm gần đây. Theo thống kê của Bitmovin vào năm 2021, hơn 1/3 (36.3%) số trang web trên toàn thế giới đã sử dụng công nghệ WebRTC để cung cấp các tính năng thời gian thực như video call và chat.

* Về lượng người sử dụng, WebRTC được tích hợp trong nhiều ứng dụng gọi video và chat real-time phổ biến như Google Meet, Zoom, Facebook Messenger, và Discord. Điều này cho thấy sự phổ biến của công nghệ này và nhu cầu sử dụng của người dùng.
* Về các trang web ứng dụng công nghệ WebRTC thành công, có thể kể đến như: Appear.in - một ứng dụng video conference được sử dụng rộng rãi trên thế giới; Houseparty - một ứng dụng gọi video và chat real-time được sử dụng phổ biến trong giới trẻ; và Talky - một ứng dụng gọi video và chat real-time đơn giản và dễ sử dụng.

Tổng quan, WebRTC là một công nghệ được ưa chuộng và phát triển mạnh mẽ trong việc xây dựng các ứng dụng gọi video và chat real-time trên web, và có sự cạnh tranh từ nhiều công nghệ tương tự. Nhiều ứng dụng thành công đã sử dụng công nghệ này để cung cấp các tính năng thời gian thực cho người dùng.

### **1.6.2. Công nghệ SIP (Session Initiation Protocol)**



***Hình 1.1 Hình minh họa SIP (Session Initiation Protocol)***

Công nghệ SIP (Session Initiation Protocol) được sử dụng để thiết lập và quản lý các cuộc gọi âm thanh, video và các phiên trò chuyện trực tuyến khác. Đây là một trong những công nghệ truyền thông thời gian thực phổ biến và được sử dụng rộng rãi trong các ứng dụng như VoIP (Voice over Internet Protocol) và video conferencing.

Tuy nhiên, theo một báo cáo của trang web Techjury, thị phần của công nghệ SIP đang giảm dần và chỉ chiếm khoảng 8,6% thị phần trên toàn cầu. Các đối thủ cạnh tranh của công nghệ này bao gồm các công nghệ khác như WebRTC, Microsoft Skype và Zoom.

* Một số website và ứng dụng thành công sử dụng công nghệ SIP bao gồm:
* ***Google Voice:*** Được ra mắt năm 2009, Google Voice cho phép người dùng gọi điện, nhận cuộc gọi và gửi tin nhắn văn bản miễn phí trong nước và quốc tế. Google Voice sử dụng công nghệ SIP để kết nối các cuộc gọi đến điện thoại của người dùng.
* ***Zoom:*** Là một trong những ứng dụng video conference phổ biến nhất trên thế giới, Zoom sử dụng công nghệ SIP để kết nối các cuộc gọi video và âm thanh.
* ***Twilio:*** Là một nền tảng đám mây cung cấp các dịch vụ truyền thông, Twilio cho phép các nhà phát triển tích hợp các tính năng truyền thông vào ứng dụng của họ bằng cách sử dụng công nghệ SIP.

Tuy nhiên, với sự phát triển của các công nghệ mới như WebRTC và các ứng dụng video conference như Zoom, thị phần của công nghệ SIP đang dần giảm đi.

### **1.6.3. WebSockets**



***Hình 1.2 Hình logo WebSockets***

Công nghệ WebSockets là một công nghệ được sử dụng để tạo ra kết nối giữa trình duyệt và máy chủ, cho phép truyền tải thông tin real-time giữa hai đầu.

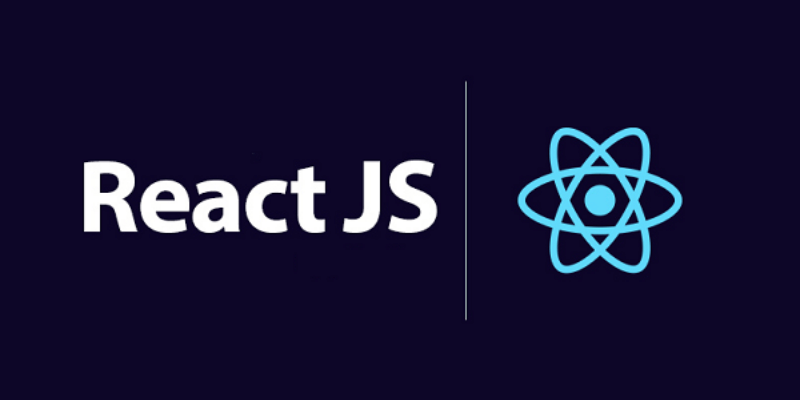
Thị phần: Hiện nay, WebSockets không phải là công nghệ phổ biến nhất trên thị trường, nhưng nó vẫn được sử dụng khá rộng rãi trong các ứng dụng web real-time như game online, chat, video call, ... Theo một báo cáo của W3Techs, chỉ khoảng 1,1% trong số 10 triệu trang web phổ biến nhất trên thế giới sử dụng công nghệ WebSockets.

* Độ phổ biến: Công nghệ WebSockets được giới chuyên môn đánh giá là một công nghệ khá tiên tiến và phát triển, với nhiều tính năng và lợi ích cho các ứng dụng web real-time. Tuy nhiên, việc sử dụng WebSockets cần có kiến thức về lập trình front-end và back-end để phát triển, cũng như đảm bảo được tính bảo mật của ứng dụng.
* Lượng người sử dụng: Do công nghệ WebSockets được sử dụng chủ yếu trong các ứng dụng web real-time, nên lượng người sử dụng phụ thuộc vào loại ứng dụng đó. Ví dụ như các trang web game online, video call hay chat sử dụng công nghệ WebSockets sẽ có lượng người sử dụng khá đông đảo.
* Các website ứng dụng công nghệ WebSockets thành công: Có nhiều trang web ứng dụng công nghệ WebSockets thành công trong việc cung cấp các tính năng real-time cho người dùng, trong đó có thể kể đến:
* ***Trello:*** Trello là một ứng dụng quản lý dự án trực tuyến nổi tiếng, cho phép người dùng tạo các bảng và thẻ để quản lý công việc. Trello sử dụng WebSockets để cập nhật thông tin ngay lập tức và đồng bộ hóa dữ liệu trên nhiều thiết bị.
* ***LiveChat:*** LiveChat là một công cụ hỗ trợ trực tuyến cho các doanh nghiệp, cho phép họ trò chuyện với khách hàng trực tiếp trên trang web. LiveChat sử dụng WebSockets để cập nhật các tin nhắn mới và trao đổi dữ liệu ngay lập tức giữa khách hàng và đội ngũ hỗ trợ.
* ***Uber:*** Uber là một ứng dụng gọi xe trực tuyến phổ biến, sử dụng WebSockets để cập nhật vị trí xe và tài xế ngay lập tức, giúp khách hàng có thể theo dõi tình trạng chuyến đi của mình.
* ***Discord:*** Discord là một ứng dụng chat và cộng tác trực tuyến cho các nhóm game thủ. Discord sử dụng WebSockets để cập nhật tin nhắn và trò chuyện ngay lập tức giữa các thành viên trong nhóm.

# Chương 2. Cơ sở lý thuyết và Công nghệ

## 2.1. Giới thiệu về ReactJS

### **2.1.1. Khái niệm:**



***Hình 2.1 Hình logo ReactJS***

ReactJS là một opensource được phát triển bởi Facebook, ra mắt vào năm 2013, bản thân nó là một thư viện Javascript được dùng để để xây dựng các tương tác với các thành phần trên website. Một trong những điểm nổi bật nhất của ReactJS đó là việc render dữ liệu không chỉ thực hiện được trên tầng Server mà còn ở dưới Client. Một trong những điểm hấp dẫn của React là thư viện này không chỉ hoạt động trên phía client, mà còn được render trên server và có thể kết nối với nhau. React so sánh sự thay đổi giữa các giá trị của lần render này với lần render trước và cập nhật ít thay đổi nhất trên DOM. Trước khi đến cài đặt và cấu hình, chúng ta sẽ đi đến một số khái niệm cơ bản:

* + ***Virtual DOM:***

Công nghệ DOM ảo giúp tăng hiệu năng cho ứng dụng. Việc chỉ node gốc mới có trạng thái và khi nó thay đổi sẽ tái cấu trúc lại toàn bộ, đồng nghĩa với việc DOM tree cũng sẽ phải thay đổi một phần, điều này sẽ ảnh hưởng đến tốc độ xử lý. React JS sử dụng Virtual DOM (DOM ảo) để cải thiện vấn đề này. Virtual DOM là một object Javascript, mỗi object chứa đầy đủ thông tin cần thiết để tạo ra một DOM, khi dữ liệu thay đổi nó sẽ tính toán sự thay đổi giữa object và tree thật, điều này sẽ giúp tối ưu hoá việc re-render DOM tree thật. React sử dụng cơ chế one-way data binding – luồng dữ liệu 1 chiều. Dữ liệu được truyền từ parent đến child thông qua props. Luồng dữ liệu đơn giản giúp chúng ta dễ dàng kiểm soát cũng như sửa lỗi. Với các đặc điểm ở trên, React dùng để xây dựng các ứng  dụng lớn mà dữ liệu của chúng thay đổi liên tục theo thời gian. Dữ liệu thay đổi thì hầu hết kèm theo sự thay đổi về giao diện.

* *Ví dụ như Facebook*: trên Newsfeed của bạn cùng lúc sẽ có các status khác nhau và mỗi status lại có số like, share, comment liên tục thay đổi. Khi đó React sẽ rất hữu ích để sử dụng.
  + ***Giới thiệu về JSX:***

JSX là một dạng ngôn ngữ cho phép viết các mã HTML trong Javascript. Đặc điểm: Faster: Nhanh hơn. JSX thực hiện tối ưu hóa trong khi biên dịch sang mã Javascript. Các mã này cho thời gian thực hiện nhanh hơn nhiều so với một mã tương đương viết trực tiếp bằng Javascript. Safer: an toàn hơn. Ngược với Javascript, JSX là kiểu statically-typed, nghĩa là nó được biên dịch trước khi chạy, giống như Java, C++. Vì thế các lỗi sẽ được phát hiện ngay trong quá trình biên dịch. Ngoài ra, nó cũng cung cấp tính năng gỡ lỗi khi biên dịch rất tốt. Easier: Dễ dàng hơn. JSX kế thừa dựa trên Javascript, vì vậy rất dễ dàng để cho các lập trình viên Javascript có thể sử dụng (*tham khảo tại* [*https://jsx.github.io/*](https://jsx.github.io/))

* + ***Giới thiệu về Components:***

React được xây dựng xung quanh các component, chứ không dùng template như các framework khác. Trong React, chúng ta xây dựng trang web sử dụng những thành phần (component) nhỏ. Chúng ta có thể tái sử dụng một component ở nhiều nơi, với các trạng thái hoặc các thuộc tính khác nhau, trong một component lại có thể chứa thành phần khác. Mỗi component trong React có một trạng thái riêng, có thể thay đổi, và React sẽ thực hiện cập nhật component dựa trên những thay đổi của trạng thái. Mọi thứ React đều là component. Chúng giúp bảo trì mã code khi làm việc với các dự án lớn. Một react component đơn giản chỉ cần một method render. Có rất nhiều methods khả dụng khác, nhưng render là method chủ đạo.

* + ***Props và State:***
* *Props:* giúp các component tương tác với nhau, component nhận input gọi là props, và trả thuộc tính mô tả những gì component con sẽ render. Prop là bất biến.
* *State:* thể hiện trạng thái của ứng dụng, khi state thay đồi thì component đồng thời render lại để cập nhật UI.

React Redux Redux là một predictable state management tool cho các ứng dụng Javascript. Nó giúp bạn viết các ứng dụng hoạt động một cách nhất quán, chạy trong các môi trường khác nhau (client, server, and native) và dễ dàng để test. Redux ra đời lấy cảm hứng từ tư tưởng của ngôn ngữ Elm và kiến trúc Flux của Facebook. Do vậy Redux thường dùng kết hợp với React.

* + - Các thành phần trong Redux:
* Actions đơn giản là các events. Chúng là cách mà chúng ta send data từ app đến Redux store. Những data này có thể là từ sự tương tác của user vs app, API calls hoặc cũng có thể là từ form submission.
* Reducers là các function nguyên thủy chúng lấy state hiện tại của app, thực hiện một action và trả về một state mới. Những states này được lưu như những objects và chúng định rõ cách state của một ứng dụng thay đổi trong việc phản hồi một action được gửi đến store. Store lưu trạng thái ứng dụng và nó là duy nhất trong bất kỳ một ứng dụng Redux nào. Bạn có thể access các state được lưu, update state, và đăng ký or hủy đăng ký các listeners thông qua helper methods.

### **2.1.2. Ưu - nhược điểm:**

#### Ưu điểm:

* Reactjs cực kì hiệu quả: Reactjs tạo ra cho chính nó DOM ảo – nơi mà các component thực sự tồn tại trên đó. Điều này sẽ giúp cải thiện hiệu suất rất nhiều. Reactjs cũng tính toán những thay đổi nào cần cập nhật lên DOM và chỉ thực hiện chúng. Điều này giúp Reactjs tránh những thao tác cần trên DOM mà nhiều chi phí.
* Reactjs giúp việc viết các đoạn code JS dễ dàng hơn: Nó dùng cú pháp đặc biệt là JSX (Javascript mở rộng) cho phép ta trộn giữa code HTML và Javascript. Ta có thể thêm vào các đoạn HTML vào trong hàm render mà không cần phải nối chuỗi. Đây là đặc tính thú vị của Reactjs. Nó sẽ chuyển đổi các đoạn HTML thành các hàm khởi tạo đối tượng HTML bằng bộ biến đổi JSX.
* Nhiều công cụ phát triển: Khi bạn bắt đầu Reactjs, đừng quên cài đặt ứng dụng mở rộng của Chrome dành cho Reactjs. Nó giúp bạn debug code dễ dàng hơn. Sau khi bạn cài đặt ứng dụng này, bạn sẽ có cái nhìn trực tiếp vào virtual DOM như thể bạn đang xem cây DOM thông thường.
* Render tầng server: Một trong những vấn đề với các ứng dụng đơn trang là tối ưu SEO và thời gian tải trang. Nếu tất cả việc xây dựng và hiển thị trang đều thực hiện ở client, thì người dùng sẽ phải chờ cho trang được khởi tạo và hiển thị lên. Điều này thực tế là chậm. Hoặc nếu giả sử người dùng vô hiệu hóa Javascript thì sao? Reactjs là một thư viện component, nó có thể vừa render ở ngoài trình duyệt sử dụng DOM và cũng có thể render bằng các chuỗi HTML mà server trả về.
* Làm việc với vấn đề test giao diện: Nó cực kì dễ để viết các test case giao diện vì virtual DOM được cài đặt hoàn toàn bằng JS.
* Hiệu năng cao đối với các ứng dụng có dữ liệu thay đổi liên tục, dễ dàng cho bảo trì và sửa lỗi.

#### Nhược điểm:

* Reactjs chỉ phục vụ cho tầng View. React chỉ là View Library nó không phải là một MVC framework như những framework khác. Đây chỉ là thư viện của Facebook giúp render ra phần view. Vì thế React sẽ không có phần Model và Controller, mà phải kết hợp với các thư viện khác. React cũng sẽ không có 2-way binding hay là Ajax.
* Tích hợp Reactjs vào các framework MVC truyền thống yêu cầu cần phải cấu hình lại.
* React khá nặng nếu so với các framework khác React có kích thước tương tương với Angular (Khoảng 35kb so với 39kb của Angular). Trong khi đó Angular là một framework hoàn chỉnh.
* Khó tiếp cận cho người mới học Web.

## 2.2. Giới thiệu về NextJS

A picture containing graphics, screenshot, font, graphic design

Description automatically generated

***Hình 2.2 Hình logo NextJS***

### **2.2.1. Các đặc điểm cơ bản của NextJs**

NextJs là một framework front-end React được phát triển dưới dạng open-source bổ sung các khả năng tối ưu hóa như render phía máy chủ (SSR) và tạo trang web static. Nextjs xây dựng dựa trên thư viện React, có nghĩa là các ứng dụng Nextjs sử dụng core của React và chỉ thêm các tính năng bổ sung. Việc triển khai ứng dụng SSR cho phép máy chủ truy cập tất cả dữ liệu được yêu cầu và xử lý JavaScript cùng nhau để hiển thị trang. Sau đó, trang được gửi lại toàn bộ cho trình duyệt và ngay lập tức được hiển thị. SSR cho phép các trang web load trong thời gian nhỏ nhất và tăng trải nghiệm người dùng với khả năng phản hồi nhanh hơn.

Ngoài ra, sử dụng SSR cũng mang lại cho bạn lợi thế về SEO, giúp trang web của bạn hiển thị cao hơn trên các trang kết quả của công cụ tìm kiếm. SSR làm cho các trang web xếp hạng tốt hơn cho SEO vì chúng tải nhanh hơn và nhiều nội dung trang web có thể được quét bởi các SEO trackers. Thẻ <head> trong Nextjs cũng cho phép bạn chỉnh sửa thẻ <head> của một trang web, điều mà bạn không thể thực hiện trong React. Thẻ <head> là một phần cốt lõi trong metadata của trang web và góp phần vào xếp hạng SEO của trang web.

### **2.2.2. Ưu nhược điểm của NextJS:**

#### Ưu điểm:

* + Mạng lại khả năng SEO tốt: Next.js cung cấp khả năng tối ưu hóa SEO tốt hơn cho ứng dụng web của bạn. Với việc sử dụng cơ chế Server Side Rendering (SSR), các trang web được tạo bởi Next.js có thể được render ngay từ phía máy chủ trước khi được gửi đến trình duyệt của người dùng. Điều này giúp các công cụ tìm kiếm như Google hiểu và đánh giá nội dung của trang web một cách tốt hơn, cải thiện khả năng xuất hiện trong kết quả tìm kiếm.
  + Trải nghiệm người dùng tốt hơn: Next.js hỗ trợ cả Server Side Rendering và Client Side Rendering (CSR), cho phép tùy chỉnh cách ứng dụng của bạn được hiển thị cho người dùng. Với việc sử dụng SSR, trang web sẽ được tải nhanh hơn và người dùng có thể xem nội dung cơ bản ngay lập tức. Sau đó, khi người dùng tương tác với trang web, Next.js sẽ chuyển sang CSR để cung cấp trải nghiệm tương tác mượt mà và nhanh chóng.
  + Hỗ trợ nền React cực kỳ tốt: Next.js được xây dựng trên nền tảng React, và do đó có sự tích hợp tốt với các thành phần React và cộng đồng React rộng lớn. Bạn có thể tận dụng tất cả các tính năng và thư viện của React khi phát triển ứng dụng Next.js của mình, bao gồm cả React Hooks, React Context và React Router.
  + Khởi tạo nhanh chóng: Next.js đi kèm với một bộ công cụ mạnh mẽ giúp bạn khởi tạo dự án một cách nhanh chóng. Bạn có thể sử dụng lệnh tạo dự án như "create-next-app" để tạo một dự án Next.js mới chỉ trong vài phút, với cấu hình mặc định sẵn sàng để bắt đầu phát triển.
  + Hỗ trợ phát triển tính năng nhanh chóng: Next.js cung cấp sự hỗ trợ tốt cho việc cấu hình các công cụ phát triển như Webpack, Babel và nhiều công cụ khác. Điều này cho phép bạn dễ dàng tùy chỉnh và mở rộng dự án của mình để đáp ứng các yêu cầu cụ thể.
  + Bảo mật về dữ liệu: Next.js có tích hợp sẵn các tính năng bảo mật cho việc xử lý dữ liệu nhạy cảm. Bạn có thể tạo các trang yêu cầu xác thực và phân quyền truy cập dựa trên vai trò người dùng, đảm bảo rằng chỉ những người có quyền truy cập mới có thể truy cập vào các dữ liệu quan trọng.
  + Khả năng thích ứng và đáp ứng thay đổi: Next.js hỗ trợ việc phát triển ứng dụng có khả năng thích ứng với các thay đổi trong yêu cầu kinh doanh và nhu cầu người dùng. Với khả năng sử dụng cả SSR và CSR, bạn có thể dễ dàng thích ứng với các yêu cầu mới và cải thiện trải nghiệm người dùng một cách linh hoạt.

#### Nhược điểm:

* + Ít plugin thích ứng: Mặc dù Next.js có một số plugin hữu ích, nhưng cộng đồng plugin cho Next.js vẫn chưa phát triển rộng rãi như các framework khác như React hoặc Vue. Điều này có nghĩa là bạn có thể gặp khó khăn trong việc tìm các plugin phù hợp để giải quyết các vấn đề cụ thể trong dự án của mình.
  + Giới hạn trong việc xử lý tuyến đường: Next.js có giới hạn về việc chỉ sử dụng bộ định tuyến mặc định của nó. Bạn không thể tùy chỉnh hoặc thay đổi cách Next.js xử lý các tuyến đường một cách linh hoạt. Để sử dụng tuyến đường động, bạn sẽ cần làm việc với Node.js server để xử lý các tuyến đường tùy chỉnh.
  + Đòi hỏi xây dựng toàn bộ front-end từ đầu: Next.js không cung cấp nhiều trang mẫu tích hợp sẵn, và do đó bạn cần phải xây dựng toàn bộ giao diện người dùng từ đầu. Điều này có thể là một thách thức đối với những người mới bắt đầu hoặc những người không có kỹ năng thiết kế giao diện tốt.

## 2.3. Giới thiệu về Supabase



***Hình 2.2 Hình logo Supabase***

### **2.2.1. Các đặc điểm cơ bản của Supabase**

Supabase là một nền tảng cung cấp các dịch vụ cơ sở dữ liệu trong thời gian thực và các công cụ để phát triển ứng dụng web và di động. Dưới đây là một số đặc điểm cơ bản của Supabase:

-PostgreSQL Database:Sử dụng cơ sở dữ liệu PostgreSQL mạnh mẽ để lưu trữ dữ liệu.

-RESTful API: Cung cấp RESTful API cho việc quản lý và truy vấn dữ liệu từ cơ sở dữ liệu.

-Real-time Data: Hỗ trợ dữ liệu thời gian thực thông qua WebSocket, giúp cập nhật dữ liệu ngay khi có sự thay đổi trong cơ sở dữ liệu.

-Authentication: Cung cấp các phương tiện để quản lý xác thực người dùng và phân quyền, bao gồm cả xác thực email, xác thực số điện thoại và xác thực bằng các nhà cung cấp bên thứ ba như Google, GitHub.

-Storage:Cung cấp dịch vụ lưu trữ tệp đơn giản, giúp bạn lưu trữ và quản lý các tệp đa phương tiện.Serverless Functions:Hỗ trợ việc triển khai các hàm serverless để xử lý logic ứng dụng mà không cần quản lý cơ sở hạ tầng.

-Dashboard và Công cụ Phát triển:Cung cấp bảng điều khiển dễ sử dụng và các công cụ phát triển để giúp quản lý và theo dõi dự án của bạn.Open Source:Mã nguồn mở, giúp cộng đồng có thể đóng góp vào phát triển và kiểm soát dự án.

-Client Libraries:Hỗ trợ các thư viện khách cho nhiều ngôn ngữ lập trình, giúp việc tích hợp và sử dụng Supabase trở nên dễ dàng.

-Data Validation and Constraints:Hỗ trợ xác nhận và ràng buộc dữ liệu để đảm bảo tính nhất quán và an toàn của cơ sở dữ liệu.

-Supabase được thiết kế để cung cấp một nền tảng toàn diện cho việc phát triển ứng dụng web và di động, giúp giảm bớt công việc quản lý cơ sở dữ liệu và hạ tầng, để nhà phát triển có thể tập trung vào việc xây dựng các tính năng của ứng dụng

.**2.2.2. Ưu nhược điểm của Supabase:**

#### Ưu điểm:

#### Supabase là một nền tảng cơ sở dữ liệu thời gian thực dựa trên PostgreSQL, cung cấp một loạt các tính năng giúp nhà phát triển xây dựng ứng dụng web và di động hiệu quả. Các ưu điểm của Supabase bao gồm sử dụng cơ sở dữ liệu mạnh mẽ, hỗ trợ dữ liệu thời gian thực, RESTful API, xác thực và phân quyền, dịch vụ lưu trữ tích hợp, serverless functions, bảng điều khiển và công cụ phát triển, mã nguồn mở, thư viện khách đa ngôn ngữ, và giá cả hợp lý. Những đặc điểm này giúp giảm bớt công việc quản lý hạ tầng và tăng cường hiệu suất phát triển ứng dụng.

#### Nhược điểm:

## Supabase, mặc dù có nhiều ưu điểm như PostgreSQL mạnh mẽ, dữ liệu thời gian thực và giá cả hợp lý, nhưng cũng có nhược điểm như cộng đồng nhỏ, thiếu một số tính năng so với đối thủ lớn, khả năng mở rộng có hạn, và phù hợp chủ yếu cho dự án nhỏ đến trung bình. Cần xem xét kỹ lưỡng trước khi chọn sử dụng, tùy thuộc vào yêu cầu cụ thể của dự án.

# Chương 3. Phân tích kiến trúc hệ thống

## 3.1. Tổng quan WebRTC

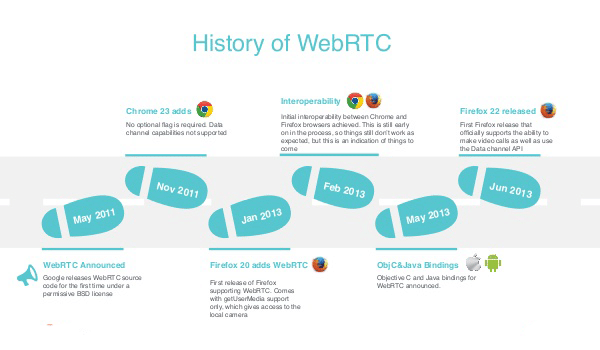
**3.1.1. Quá trình phát triển**

#### a) WebRTC là gì?

WebRTC là các API viết bằng Javascript giúp giao tiếp theo thời gian thực mà không cần cài plugin hay phần mềm hỗ trợ. WebRTC có khả năng hỗ trợ trình duyệt giao tiếp thời gian thực thông qua Video Call, Voice Call hay transfer data P2P(peer-to-peer), không cần đến plugin, phần mềm khác.

WebRTC là viết tắt của cụm từ Web Real-Time Communication rất được các lập trình viên ưa chuộng. WebRTC là một sản phẩm của World Wide Web Consortium (W3C).WebRTC cho phép các trình duyệt giao tiếp với nhau theo thời gian thực .Ví dụ như: gọi điện, video, chơi game,…

#### b) Lịch sử của WebRTC



***Hình 3.1 Lịch sử hình thành WebRTC***

WebRTC đã được hình thành ý tưởng từ những năm 2009 bởi nhóm kỹ sư Google Hangouts. Thay vì sử dụng Flash để truyền video, hình ảnh trên web, học quyết định tự tạo ra một sản phẩm của riêng mình.

Năm 2010, khi hai công ty On2 và Global IP Solutions(GIPS) chính thức bị Google thâu tóm, công nghệ truyền dữ liệu thời gian thực cũng được lấy làm nền tảng cho WebRTC. Đến tháng 5 năm 2011, dự án nguồn mở hỗ trợ giao tiếp thời gian thực giữa trình duyệt mang tên WebRTC mới chính thức được ra mắt.

W3C và Hiệp Hội Kỹ Sư Quốc Tế(IETF) cũng phát triển các giao thức kết nối thời gian thực. Vì thế, họ đã quyết định bắt tay để cùng nhau xây dựng sản phẩm này. Tháng 10 năm 2011, phiên bản đầu tiên của WebRTC chính thức ra mắt, đến tháng 11, Chrome 23 ra đời và trở thành trình duyệt đầu tiên tích hợp sẵn WebRTC.

### **3.1.2. Sự hỗ trợ từ trình duyệt**

Chrome và Firefox là hai trình duyệt hỗ trợ mạnh nhất cho WebRTC, không có gì lạ khi mà WebRTC được ủng hộ rất mạnh mẽ bởi Google và Mozilla. Opera cũng cho phép chạy hầu hết các tính năng quan trọng nhưng bạn sẽ không thể chia sẻ màn hình cho người khác được. Đứng cuối bảng có lẽ là IE và Safari với việc hỗ trợ cho WebRTC rất kém. Nhưng nói tóm lại thì bộ hàm này vẫn chưa được hỗ trợ một cách đầy đủ từ các hãng trình duyệt, bởi ngay cả Chrome từ Google cũng không thể tương thích 100% các hàm API có trong WebRTC.

Nhưng hãng không hỗ trợ không có nghĩa là chúng ta hoàn toàn không thể xài được WebRTC. Vẫn có những plugin cài thêm từ bên ngoài vào để giúp trình duyệt tương thích tốt hơn, nhưng lúc đó thì sự thuận lợi sẽ không còn nữa. Điểm tuyệt vời của WebRTC là phải được hỗ trợ sẵn từ trong trình duyệt để người ta không cần cài thêm plugin gì cơ mà.

Tóm lại, những trình duyệt lớn sau là có hỗ trợ WebRTC. Phiên bản ghi trong bài là phiên bản đầu tiên được triển khai WebRTC, còn hiện tại thì chúng đã được update lên mới hơn.

* PC
  + - * Google Chrome 23
      * Mozilla Firefox 22
      * Opera 18
* Android
* Google Chrome 28 (Enabled by default since 29)
* Mozilla Firefox 24
* Opera Mobile 12
* Chrome OS
* Firefox OS
* iOS
* Bowser

### **3.1.3. Kiến trúc WebRTC trong trình duyệt**

A picture containing text, screenshot, line, diagram

Description automatically generated

***Hình 3.2 Kiến trúc WebRTC***

Ứng dụng web với WebRTC (thường viết bằng HTML5 và JavaScript) tương tác với trình duyệt qua những WebRTC APIs đang được chuẩn hóa, cho phép nó khai thác hợp lý và điều khiển chức năng thời gian thực của trình duyệt.

Hình 3.2 cho thấy mô hình trình duyệt và vai trò của các chức năng truyền thông thời gian thực. Khối màu sáng là chức năng truyền thông thời gian thực (Real Time Communication – RTC) của trình duyệt. Do tính chất riêng và yêu cầu của truyền thông thời gian thực nên việc chuẩn hóa khối này là không đơn giản, hiện tại vẫn đang trong quá trình bàn thảo. Các chức năng RTC tương tác với các ứng dụng web sử dụng các APIs chuẩn. Nó giao tiếp với các hệ điều hành bằng cách sử dụng trình duyệt.



***Hình 3.3 Kiến trúc WebRTC***

* Trong kiến trúc WebRTC có 3 lớp API:
* APIs cho nhà lập trình web: lớp này chứa tất cả các APIs mà nhà lập trình web cần, bao gồm các đối tượng chính là RTCPeerConnection, RTCDataChannel, MediaStream.
* APIs cho nhà phát triển trình duyệt sử dụng.
* Overridable API: nhà phát triển trình duyệt có thể thay đổi, phát triển APIs của riêng mình.

### **3.1.4. Các APIs trong Web**

WebRTC bao gồm các APIs, các giao thức liên quan và làm việc với nhau để hỗ trợ việc trao đổi dữ liệu đa phương tiện giữa các trình duyệt. WebRTC đang trong quá trình chuẩn hóa và sử dụng các APIs quanh ba khái niệm chính:

A picture containing text, screenshot, diagram, design

Description automatically generated

***Hình 3.4 Cấu trúc WebRTC***

* ***MediaStream:***
* MediaStream là một stream dữ liệu âm thanh và hình ảnh, bằng cách gọi hàm getUserMedia để khởi tạo khi làm việc cục bộ. MediaStream sẽ cho phép truy cập vào stream của một máy tính sau khi một kết nối WebRTC được thiết lập với một máy tính khác.
* Một MediaStream sẽ có input và output với input dùng để lấy dữ liệu hình ảnh và âm thanh của local trong khi output dùng để hiển thị các dữ liệu này lên view hoặc được RTCPeerConnection sử dụng.
* ***RTCDataChannel:***

RTCDataChannel là một kênh hai chiều chịu trách nhiệm trao đổi dữ liệu thời gian thực. RTCDataChannel sẽ trao đổi các dữ liệu dạng text, chia sẻ tệp P2P và các loại khác mà không chứa các dữ liệu nghe nhìn(âm thanh, hình ảnh). Khả năng trao đổi thông tin nhanh chóng, an toàn, đáng tin khiến RTCDataChannel được khai thác để xây dựng các giải pháp mới và hiệu quả về chi phí.

* ***RTCPeerConnection:***

RTCPeerConnection là phần quan trọng giúp kết nối MediaStream và RTCDataChannel trở thành WebRTC. RTCPeerConnection là API giúp kết nối giữa hai trình duyệt, cung cấp các phương thức để kết nối, duy trì kết nối và đóng kết nối khi không còn nhu cầu sử dụng.

### **3.1.5. Các tầng giao thức trong WebRTC**

A picture containing text, screenshot, font, number

Description automatically generatedWebRTC bao gồm các APIs, các giao thức liên quan và làm việc với nhau để hỗ trợ việc trao đổi dữ liệu đa phương tiện giữa các trình duyệt. WebRTC đang trong quá trình chuẩn hóa và sử dụng các APIs quanh ba khái niệm chính:

***Hình 3.5 Các tầng giao thức trong WebRTC***

* ***SRTP***

Giao thức quan trọng nhất mà WebRTC sử dụng là Secure Real-time Transport Protocol, hay SRTP. SRTP được sử dụng để mã hóa và chuyển các gói tin media giữa các WebRTC client. Sau khi thiết lập thành công PeerConnection, kết nối SRTP sẽ được thiết lập giữa các trình duyệt hoặc trình duyệt và máy chủ. Với dữ liệu nonaudio hay video, SRTP không được sử dụng, thay vào đó là SCTP.

* ***SCTP***

WebRTC sử dụng SCTP - Stream Control Transmission Protocol để truyền các dữ liệu non-media giữa các Peer. Giao thức SCTP là giao thức vận chuyển, tương tự như TCP và UDP, có thể chạy trực tiếp trên giao thức IP. Tuy nhiên trong WebRTC, SCTP chạy trên DTLS tên UDP. SCTP được lựa chọn do có những tính năng tốt nhất của TCP và UDP như: message-oriented transmission, khả năng cấu hình tùy biến tính tin cậy và thứ tự gói tin, có cơ chế quản lý lưu lượng và chống nghẽn.

* ***SDP***

WebRTC sử dụng Session Description Protocol - SDP, được encode trong đối tượng RTCSessionDescription, để mô tả đặc tính media của hai đầu trong kết nối P2P như loại media đề truyền/nhận (audio, video, application data), network transports, loại codecs sử dụng và cấu hình, thông tin băng thông, và các thông tin metadata khác. Thông điệp SDP được trao đổi qua máy chủ báo hiệu hay còn gọi là được trao đổi qua kênh báo hiệu. Máy chủ báo hiệu có trách nhiệm gửi và nhận tất cả thông điệp đến tất cả các peers mà mong muốn kết nối đến peer khác. Mặc dù SDP là định dạng dữ liệu dùng để trao đổi, thống nhất thông số giữa kết nối Peer-to-Peer, nhưng do WebRTC không ràng buộc cho các SDP “offer” và “answer” giao tiếp như nào, nên nó không được thể hiện ở Hình x.x ở trên. Tuy nhiên, mô hình offer/answer được thiết kế tuân thủ theo RFC3264.

* ***DTLS***

Datagram Transport Layer Security- DTLS dựa trên giao thức TLS, cung cấp tính bảo mật và toàn vẹn dữ liệu truyền giữa các ứng dụng tương tự TLS. Tuy nhiên, WebRTC sử dụng DTLS do nó chạy trên giao thức UDP thích hợp với việc vượt NAT cho các ứng dụng P2P. Tất cả các dữ liệu truyền P2P đều được bảo mật sử dụng DTLS. Cụ thể, DTLS được sử dụng trong việc thống nhất khóa bảo mật cho việc mã hóa dữ liệu media và cho việc bảo mật sự vận chuyển dữ liệu ứng dụng. Mặc dù cung cấp tính mã hóa, tính toàn vẹn, nhưng phần xác thực trong WebRTC được gán cho ứng dụng.

* ***STUN***

Session Traversal Utilities for NAT - STUN, là giao thức giúp cho việc vượt NAT trong quá trình thiết lập kết nối. Trong WebRTC, một STUN client sẽ được xây dựng trong User Agent của trình duyệt để kết nối đến STUN server ngoài Internet. STUN server thực thi nhiệm vụ khá đơn giản, kiểm tra thông tin địa chỉ IP, port của request đến từ ứng dụng sau NAT, sau đó trả thông tin đó về dưới dạng response, nói cách khác là STUN giúp ứng dụng biết địa chỉ IP, cổng của nó sử dụng khi đi ra Internet. STUN có thể được vận chuyển trên UDP, TCP hoặc TLS Trong đa số các trường hợp thì chỉ cần sử dụng STUN trong việc thiết lập kết nối P2P, trừ trường hợp một peer đứng sau symmetric NAT, một peer đứng sau Symmetric NAT hoặc port-restricted NAT. Trường hợp này quá trình hole punching sẽ không thành công, cần phải sử dụng đến TURN - Traversal Using Relays around NAT.

* ***TURN***

Traversal Using Relays around NAT - TURN, là một mở rộng (extension) của giao thức STUN, cung cấp media relay cho tình huống thực hiện hole punching không thành công. Trong WebRTC, User Agent của trình duyệt sẽ bao gồm một TURN client. TURN server được cung cấp trên Internet qua các nhà cung cấp dịch vụ, hoặc có thể cài đặt trong mạng doanh nghiệp. Giao thức UDP được sử dụng để giao tiếp giữa TURN client và TURN server qua NAT. Cổng UDP mặc định cho TURN là 3478. Trên thực tế TURN server thường là STUN server có bổ sung tính năng relay. TURN server thì có chức năng STUN, nhưng không phải mọi STUN server đều có chức năng TURN.

* ***ICE***

Interactive Communication Establishment - ICE là giao thức quan trọng trong WebRTC, sử dụng kỹ thuật hole punching [RFC5128] để vượt NAT.

### **3.1.6. Ưu và nhược điểm của WebRTC**

#### Ưu điểm:

* **Mã nguồn mở miễn phí:** WebRTC là một dự án mã nguồn mở miễn phí. Google cho biết đây là một công cụ truyền thông thời gian thực hoàn toàn miễn phí và có sẵn trên mọi trình duyệt.
* **Hỗ trợ đa nền tảng:** Mặc dù WebRTC vẫn trong giai đoạn phát triển nhưng nó đã hoạt động tốt trên hầu hết mọi trình duyệt của các hệ điều hành bất kì. Cho phép lập trình viên viết các đoạn mã HTML làm việc với máy tính hoặc thiết bị di động.
* **Bảo mật voice và video:** Giao thức SRTP (Secure Real-Time Transport Protocol) được dùng để mã hóa và xác thực dữ liệu media. Chống lại các khả năng bị nghe trộm trong quá trình thực hiện tác vụ video hay voice.
* **Không cần plugin hay phần mềm hỗ trợ:** Yếu tố quan trọng giúp WebRTC được đánh giá rất cao chính là khả năng hoạt động không cần đến plugin bên thứ ba mang đến sự tiện lợi, tối ưu tốc độ, tiết kiệm chi phí,…
* **Tương đối dễ sử dụng:** WebRTC có thể được tích hợp trong các dịch vụ web bằng cách dùng JavaScript APIs, các Framework có sẵn.
* **Sử dụng bằng thông hiệu quả:** Hỗ trợ nhiều kiểu media và các thiết bị đầu cuối khác nhau, WebRTC sử dụng băng thông hiệu quả hơn, hoạt động tốt trong mọi điều kiện đường truyền mạng.

#### Nhược điểm:

***Hình 3.6 Giới hạn đa nhiệm trình duyệt của WebRTC***

* WebRTC bị cản bởi NAT và tường lửa khi cố gắng thực hiện kết nối P2P.
* Không có cơ chế báo hiệu cài sẵn khi WebRTC tạo kết nối P2P giữa các trình duyệt.
* WebRTC vẫn chưa chính thức hoàn thiện, một số trình duyệt như IE, Safari chưa thực sự được hỗ trợ tốt nhất.
* Các hãng trình duyệt chưa thống nhất được chuẩn video sử dụng cho WebRTC.
* Số lượng hàm API WebRTC hỗ trợ cho mỗi trình duyệt là khác nhau, tăng rủi ro phát sinh lỗi khi sử dụng trên các trình duyệt khác nhau.

## 3.2. Báo hiệu trong WebRTC

### **3.2.1. Vai trò của báo hiệu**

Quá trình chuyển các thông điệp từ trình duyệt này qua máy chủ trung gian đến trình duyệt khác được gọi là quá trình báo hiệu, hay gọi tắt là báo hiệu. Máy chủ trung gian là máy chủ báo hiệu. Báo hiệu không được chuẩn hóa trong WebRTC, cho phép nhà phát triển ứng dụng tự do lựa chọn phương án phù hợp. Trong truyền thông thời gian thực, báo hiệu có bốn vai trò chính:

* Xác định địa chỉ vận chuyển (IP và port) của peer: trao đổi địa chỉ ứng viên trong quá trình ICE hole punching.
* Thương lượng/thống nhất khả năng media và các thiết lập cấu hình: đây là chức năng quan trọng nhất của báo hiệu, giúp trao đổi thông tin thường được chứa trong đối tượng SDP giữa các trình duyệt tham gia vào Peer Connection. SDP chứa tất cả các thông tin cần thiết cho RTP media stack trên trình duyệt để cấu hình media session, bao gồm loại media (voice, video, data), codecs sử dụng (Opus, G.711,..) hay bất kỳ tham số hay thiết lập nào cho codecs, thông tin về băng thông.
* Định danh và xác thực các bên tham gia trong session.
* Điều khiển media session: khởi tạo, đóng, thay đổi session.

Lý do báo hiệu không được chuẩn hóa trong WebRTC đơn giản vì nó không cần được chuẩn hóa để giúp trình duyệt có khả năng tương tác với nhau. Hiện nay, quá trình báo hiệu đang được xây dựng dựa trên Javascript Session Establishment Protocol (JSEP), là cơ chế cho phép ứng dụng JavaScript kiểm soát hoàn toàn phần báo hiệu của phiên đa phương tiện qua API RTCPeerConnection.

### **3.2.2. Giao thức vận chuyển báo hiệu**

Các giao thức được sử dụng phổ biến cho vận chuyển báo hiệu WebRTC là HTTP, WebSocket và đặc biệt kênh Data Channel. Kênh dữ liệu, sau khi được thiết lập P2P giữa trình duyệt, cung cấp kết nối trực tiếp, độ trễ thấp nên cũng phù hợp với việc vận chuyển báo hiệu. Vì sự khởi tạo thiết lập kênh dữ liệu (Data Channel) đòi hỏi cơ chế báo hiệu riêng, kênh dữ liệu không thể sử cho tất cả báo hiệu WebRTC. Tuy nhiên, nó có thể sử dụng để handle các báo hiệu khác sau khi thiết lập thành công kênh trực tiếp, bao gồm báo hiệu cho audio, video media qua Peer Connection.

A diagram of a computer network

Description automatically generated with low confidence

***Hình 3.7 Giao thức Peer Connection***

### **3.2.3. Giao thức báo hiệu**

Một phần quan trọng trong xây dựng ứng dụng WebRTC là lựa chọn giao thức báo hiệu, nó không cần thiết gắn với việc lựa chọn giao thức vận chuyển báo hiệu. Ta có thể chọn giao thức báo hiệu chuẩn như SIP, Jingle hoặc một cách báo hiệu riêng tự định nghĩa.

* ***Sử dụng giao thức báo hiệu SIP***

SIP là giao thức báo hiệu thường sử dụng trong VoIP và hệ thống hội nghị truyền hình. SIP có thể sử dụng UDP, TCP, SCTP hay TLS cho việc vận chuyển, tuy nhiên thường thì sử dụng Websocket.

* ***Sử dụng giao thức báo hiệu Jingle over WebSockets***

Jingle là một mở rộng của XMPP (extensible Messaging and Presence Protocol), được biết đến là Jabber [RFC6120]), thêm khả năng báo hiệu media cho XMPP. Jingle cung cấp cách chuyển mô tả phiên SDP sang định dạng XML, sau đó có thể được vận chuyển qua TCP hoặc TLS đến máy chủ XMPP. Để triển khai báo hiệu sử dụng Jingle, client phải load đoạn mã JavaScrip từ máy chủ để thiết lập kết nối XMPP qua WebSockets với máy chủ XMPP. Client sau đó sẽ map thông tin SDP offer và SDP answer được tạo ra bởi trình duyệt thành thông điệp Jingle call setup và chuyển tiếp nó cho trình duyệt kia.

* ***Sử dụng JSON over WebSockets***

Hướng tiếp cận này được Google sử dụng và tương đối phổ biến hiện nay. JSON có thể coi là tập con cú pháp JavaScript nên có ưu điểm lớn khi thông dịch bởi trình duyệt web. Tất cả cấu trúc dữ liệu và thông tin trạng thái trong ứng dụng web được lưu trữ trong đối tượng đều được map rõ ràng, trực tiếp vào JSON nên không đòi hỏi nhiều nỗ lực cho việc mã hóa (encoding), phân tích (parsing) và xử lý (processing. Sử dụng JSON cùng với thư viện đảm nhận việc thiết lập và duy trì kênh hai chiều tin cậy với máy máy chủ báo hiệu là khá đơn giản và hiệu quả, dù phát sinh thêm chi phí dựng cổng ứng dụng tùy biến (custom gateway) nếu muốn kết nối ứng dụng web với dịch vụ thông tin liên lạc bên ngoài.

### **3.2.4. Các quá trình trong báo hiệu**

Có ba quá trình “bán” bất đồng bộ chính trong thiết lập session WebRTC bao gồm:

* *WebRTC Javascript callback logic:* các logic này handle tất cả những xử lý mức trình duyệt của WebRTC.
* *STUN/TURN Sever Session Description Protocol (SDP) messaging*: Đây là logic báo hiệu diễn ra ngoài kết nối WebRTC để cài đặt kết nối P2P giữa 2 trình duyệt theo yêu cầu.
* *ICE (Interactive Connectivity Establishment) messaging*: quá trình hỗ trợ vượt NAT, chuyển tiếp (relay) media/data trong trường hợp cần thiết.

Các quá trình tạm gọi là bán đồng bộ vì trong chuỗi kết nối, luồng logic call back và luồng logic báo hiệu được kích hoạt (gọi) lẫn nhau. Trong đó quá trình SDP, ICE nêu trên đều là một phần của báo hiệu, đều cần có máy chủ báo hiệu riêng để chuyển tiếp thông điệp SDP, hoặc để chuyển tiếp thông điệp ICE.

## 3.3. Tổng quan về HLS

### **3.3.1. Giới thiệu về HLS** HLS là một giao thức truyền video. Nhưng chính xác thì nó có nghĩa là gì? Bạn có thể đã nghe thuật ngữ codec, nhưng điều quan trọng cần lưu ý là giao thức phát trực tuyến không phải là codec. Giao thức là một phạm trù rộng hơn.

### Giao thức phát trực tuyến là một phương pháp được tiêu chuẩn hóa để truyền nội dung video giữa các thiết bị. Một giao thức có thể giả sử việc sử dụng một codec (hoặc codec) cụ thể, sẽ được sử dụng để nén và giải nén nội dung video và âm thanh.

**3.3.2. Nguyên tắc hoạt động**

HTTP Live Streaming (HLS) là giao thức truyền tải dữ liệu đa phương tiện qua internet, thường được sử dụng cho phát sóng video trực tuyến. Nguyên tắc hoạt động bao gồm chia nhỏ nội dung thành các đoạn và tạo tệp playlist M3U8, phân phối qua HTTP, hỗ trợ Adaptive Bitrate Streaming (ABR) để tự động chuyển đổi chất lượng, và thích ứng với điều kiện mạng. HLS sử dụng buffering và phát lại, tương thích rộng rãi trên nhiều nền tảng, bảo mật và kiểm soát tốt quá trình phát sóng.

**3.3.3. Adaptive Bitrate Streaming (ABR):** Adaptive Bitrate Streaming (ABR) là tính năng quan trọng của HTTP Live Streaming (HLS), cho phép tự động điều chỉnh chất lượng video dựa trên điều kiện mạng và hiệu suất thiết bị người xem. ABR giúp cải thiện trải nghiệm xem bằng cách tự động chuyển đổi giữa các phiên bản chất lượng video khác nhau để đảm bảo phát sóng liên tục và không gián đoạn, ngay cả trong điều kiện mạng thay đổi.

**3.3.4. Phân phối qua HTTP** Phân phối qua HTTP là quá trình cung cấp nội dung đa phương tiện, như video, thông qua giao thức HTTP. Với phương pháp này, tệp playlist M3U8 và các đoạn video được truyền tải qua HTTP, giúp dễ dàng vượt qua các ràng buộc tường lửa và proxy. Điều này tạo ra một cách tiếp cận linh hoạt cho việc phân phối nội dung trực tuyến, với khả năng tương thích rộng rãi trên nhiều nền tảng và thiết bị.

# Chương 4. Thiết kế hệ thống

## 4.1. Thiết kế use case

**4.1.1. Sơ đồ use case** A picture containing diagram, text, line, circle

Description automatically generated

***Hình 4.1 Sơ đồ use case***

**4.1.2. Danh sách các actor**

|  |  |
| --- | --- |
| **Tên Actor** | **Mô tả** |
| Người dùng | Người dùng đăng nhập vào hệ thống vào các phòng của hệ thống để sử dụng các tính năng như gọi video âm thanh,chia sẽ màn hình và truyền dữ liệu trong thời gian thực |

**4.1.3. Danh sách các use case**

* Dưới đây là danh sách các use case của hệ thống. Tất cả đều có tương tác với actor duy nhất của ứng dụng là Người dùng.
* Đăng ký
* Đăng nhập
* -Sau khi join vào ground thì sử dụng các tính năng như chat,video, share màn hình

**4.1.4. Đặc tả use case**

#### 4.1.4.1 Đăng ký

|  |  |
| --- | --- |
| **Mã Use Cake** | #UC\_DK |
| **Tên Use Cake** | Đăng Ký |
| **Mô Tả** | Để sử dụng ứng dụng, người dùng cần có tài khoản.Người dùng đăng ký bằng tài khoản gmail |
| **Người thực hiện** | Người dùng |
| **Kết quả sau khi sử lý** | Người dùng đăng ký thành công và chuyển đến màn hình cập nhật  thông tin cho người đăng ký mới. |
| **Ngoại lệ** | Trường hợp tài khoản đã được đăng ký, hệ thống hiển thị lỗi cho người dùng |
| **Dòng sự kiện chính** | 1. Người dùng đăng ký nhập tên gmail và mật khẩu 2. Người dùng đồng ý các điều khoản đã nhập và chọn kết nối ứng dụng 3. Hệ thống dẫn người dùng đến trang đăng nhập. |
| **Dòng sự kiện khác** | Không |

#### 4.1.4.1 Đăng Nhập

|  |  |
| --- | --- |
| **Mã Use Cake** | #UC\_DN |
| **Tên Use Cake** | Đăng Nhập |
| **Mô Tả** | Người dùng đăng nhập vào hệ thống bằng tài khoản đã đăng ký |
| **Người thực hiện** | Người dùng |
| **Kết quả sau khi sử lý** | Người dùng đăng nhập thành công và được chuyển đến màn hình chính của ứng dụng |
| **Ngoại lệ** | Nếu tài khoản chưa được đăng ký, hệ thống sẽ đăng ký và yêu cầu cập nhật thông tin như ở use case đăng ký |
| **Dòng sự kiện chính** | 1. Người dùng đăng ký nhập bằng tài khoản đã đăng ký 2. Hệ thống dẫn người dùng đến trang chủ |
| **Dòng sự kiện khác** | Không |

#### 4.1.4.1 Đăng Nhập

|  |  |
| --- | --- |
| **Mã Use Cake** | #UC\_RTC |
| **Tên Use Cake** | RealTimeCommunication |
| **Mô Tả** | Người dùng sau khi đăng nhập vào trang chủ thì chọn user muốn nhắn tin trao đổi và gọi video call |
| **Người thực hiện** | Người dùng |
| **Kết quả sau khi sử lý** | Người dùng đăng nhập thành công và sử dụng các tính năng của một wedRTC như chat RealTime và gọi video call với các user khác |
| **Ngoại lệ** | Sẽ không gọi được các user khác nếu các user khác không online |
| **Dòng sự kiện chính** | 1. Người dùng đăng ký nhập thành công vào hệ thống 2. Người dùng chọn đến phòng có user muốn giao tiếp và gọi điện 3. Người dùng nhắn tin realtime chat và có thể gọi video call với người dùng đang online |
| **Dòng sự kiện khác** | Không |

## 4.2. Thiết kế cơ sở dữ liệu

### **4.2.1. Sơ đồ Database**

A screenshot of a computer

Description automatically generated

***Hình 4.2 Khóa chính – Khóa ngoại trong database***

### **4.2.2. Mô tả các bảng**

#### 4.2.2.1.Bảng user

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên ID | Kiểu dữ liệu | Khóa | Ràng buộc | Note |
| MessegeID | String | Khóa chính | Not null |  |
| User Email | String |  |  |  |
| User Password | String |  |  |  |
| User Fullname | String |  |  |  |

#### 4.2.2.2.Bảng Conversation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên ID | Kiểu dữ liệu | Khóa | Ràng buộc | Note |
| Conversation Id | String | Khóa chính | Not null |  |
| User Id 1 | String |  |  |  |
| User Id 2 | String |  |  |  |

#### 4.2.2.3.Bảng MessageId

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên ID | Kiểu dữ liệu | Khóa | Ràng buộc | Note |
| Conversation Id | String | Khóa chính | Not null |  |
| Conversation Id | String |  |  |  |
| SenderID | String |  |  |  |
| ReceiverID | String |  |  |  |
| Content | String |  |  |  |
| Timestamp | DateTime |  |  |  |

# Chương 5. Preference Test Report

## 5.1. Study Design

Chúng em đang cân nhắc cải thiện hệ thống WebRTC của mình bằng cách bổ sung các tính năng mới. Tuy nhiên, trước khi quyết định triển khai, chúng em muốn đánh giá sự ưu tiên của mọi người đối với các tính năng mới này.

Để giải quyết vấn đề này, chúng em quyết định sử dụng phương pháp Preference Test để thu thập và phân tích ý kiến của các đối tượng liên quan. Các tính năng được đề xuất bao gồm:

* ***Tính năng A:*** Hỗ trợ gọi video trực tuyến.
* ***Tính năng B:*** Hỗ trợ chia sẻ màn hình.
* ***Tính năng C:*** Hỗ trợ ghi âm cuộc gọi.
* ***Tính năng D:*** Hỗ trợ truyền file đa phương tiện.
* ***Tính năng E:*** Hỗ trợ mã hóa dữ liệu.

Sau đó, chúng em tiến hành các bước sau:

1. Xây dựng bảng câu hỏi khảo sát để thu thập ý kiến của mọi người đối với các tính năng được đề xuất.

2. Phân phối bảng câu hỏi cho các đối tượng tương ứng để thu thập ý kiến của họ.

3. Tiến hành phân tích dữ liệu và tính toán điểm ưu tiên của từng tính năng dựa trên phản hồi của đối tượng.

4. Trình bày báo cáo với kết quả phân tích và đề xuất các tính năng nên được triển khai trước.

Kết quả của nghiên cứu sẽ giúp chúng em đưa ra quyết định tốt hơn về việc triển khai các tính năng mới và cải thiện trải nghiệm người dùng của hệ thống quản lý trường học của mình.

## 5.2. Methods

Khảo sát trực tuyến: Phương pháp này đã được sử dụng trong nhiều nghiên cứu về đánh giá sự ưu tiên của người dùng đối với các tính năng của WebRTC. 27/10/2011: Bản dự thảo WebRTC đầu tiên được W3C công bố.

Tiến hành khảo sát trực tuyến một nhóm khách hàng quy mô nhỏ về vấn đề nghiêp vụ cho WebRTC đồng thời tiến hành phân tích dưa ra quyết định tốt hơn về việc triển khai các tính năng mới trong phiên bản mới của hệ thống WebRTC, từ đó nâng cao trải nghiệm người dùng và cải thiện hiệu quả hoạt động.

## 5.3. Đối tượng

Đối tượng của WebRTC và Video Streaming là tất cả người dùng của hệ thống, bao gồm mọi cá nhân hoặc tổ chức có nhu cầu sử dụng truyền thông trực tiếp qua video, âm thanh hoặc dữ liệu trên web. Điều này có thể bao gồm:

1. Người dùng cá nhân: Bất kỳ ai muốn sử dụng các tính năng truyền thông trực tiếp của WebRTC, Video Streaming để thực hiện cuộc gọi video, gọi thoại hoặc chia sẻ dữ liệu trực tuyến với người khác trên web.

2. Các doanh nghiệp và tổ chức: Những tổ chức như các doanh nghiệp, tổ chức phi lợi nhuận, tổ chức giáo dục hoặc chính phủ có thể sử dụng WebRTC để thiết lập các hệ thống giao tiếp trực tuyến trong nội bộ hoặc với khách hàng, đối tác hoặc người dùng cuối khác.

3. Nhà phát triển ứng dụng: Các nhà phát triển ứng dụng có thể sử dụng WebRTC để tích hợp tính năng truyền thông trực tiếp vào ứng dụng của họ, bao gồm ứng dụng di động, trang web, ứng dụng hội nghị trực tuyến và nhiều loại ứng dụng khác.

4. Nhà cung cấp dịch vụ: Các nhà cung cấp dịch vụ trực tuyến như các nền tảng họp trực tuyến, ứng dụng gọi điện trực tuyến, dịch vụ trò chuyện trực tuyến có thể sử dụng WebRTC để cung cấp truyền thông trực tiếp cho người dùng của họ.

Để đánh giá ý kiến và ưu tiên của các đối tượng này đối với tính năng và trải nghiệm của WebRTC, bạn có thể triển khai phương pháp khảo sát trực tuyến như đã đề xuất trước đó, thu thập phản hồi và phân tích dữ liệu để đưa ra quyết định về việc cải thiện và phát triển tính năng của WebRTC.

## 5.4. Kết quả

Phát hiện chính của nghiên cứu là phiên bản kiểm soát và thử nghiệm thực hiện tương tự nhau.

1. Đăng nhập (all): Tính năng cho phép người dùng đăng nhập vào hệ thống bằng tài khoản và mật khẩu đã đăng ký trước đó. Người dùng sẽ được chuyển hướng đến trang chính của hệ thống sau khi đăng nhập thành công.

2. Đăng xuất (all): Tính năng cho phép người dùng đăng xuất khỏi hệ thống sau khi hoàn tất các hoạt động và không muốn tiếp tục sử dụng. Người dùng sẽ được chuyển hướng đến trang đăng nhập sau khi đăng xuất thành công.

3. Đăng kí (user): Tính năng cho phép người dùng tạo tài khoản mới trong hệ thống bằng cách cung cấp các thông tin cần thiết như tên, địa chỉ email và mật khẩu. Sau khi đăng ký thành công, người dùng sẽ có thể sử dụng tài khoản của mình để đăng nhập vào hệ thống.

4. Gọi thoại và video: WebRTC cho phép thiết lập cuộc gọi thoại và video trực tiếp trên trình duyệt mà không cần cài đặt phần mềm bổ sung. Người dùng có thể thiết lập cuộc gọi 1-1 hoặc tham gia các cuộc họp trực tuyến với nhiều người tham gia.

5. Truyền dữ liệu trực tiếp (Data Channel): WebRTC cho phép truyền dữ liệu trực tiếp giữa các trình duyệt, không chỉ giới hạn ở gọi thoại và video. Điều này cho phép chia sẻ tệp tin, tin nhắn và dữ liệu khác giữa các người dùng.

6. Giao diện đa phương tiện (Media Devices): WebRTC cung cấp khả năng truy cập vào các thiết bị âm thanh và video trên máy tính hoặc thiết bị di động. Người dùng có thể chọn thiết bị đầu vào và đầu ra âm thanh/video mà họ muốn sử dụng trong cuộc gọi.

7. Xử lý mã hóa và giải mã: WebRTC hỗ trợ mã hóa và giải mã âm thanh/video để đảm bảo tính riêng tư và bảo mật trong quá trình truyền dữ liệu. Các thuật toán mã hóa hiện đại được sử dụng để bảo vệ thông tin cá nhân và ngăn chặn việc đánh cắp dữ liệu.

8. Điều chỉnh băng thông (Adaptive Bitrate): HLS có khả năng điều chỉnh tỷ lệ bit truyền thông theo điều kiện mạng hiện tại để đảm bảo chất lượng cuộc gọi tốt nhất. Điều này giúp giảm thiểu sự gián đoạn và đảm bảo cuộc gọi vẫn liên tục và ổn định trong các môi trường mạng khác nhau.

Nhận thức về tần suất sử dụng website quản lý trường học của học sinh và sinh viên.

|  |  |
| --- | --- |
| **Bạn có thường xuyên sử dụng WebRTC để tiến hành các cuộc họp nhóm ,học tập ,trao đổi (Đối với học sinh sinh viên)** | **Phần trăm** |
| Ít hơn 25% thời gian | 25% |
| 25-49% thời gian | 45% |
| 50-75% thời gian | 10% |
| Hơn 75% thời gian | 5% |
| Tôi không chắc | 15% |

***Biểu đồ 5.1 Tần suất sử dụng website quản lý trường học của học sinh và sinh viên***

## 5.5. Sự hiểu

Các kết quả thử nghiệm và kiểm soát thực hiện rất giống nhau về sự hiểu biết của người dùng. Mỗi câu hỏi được chấm từ bốn điểm, trong đó một kết quả phù hợp hoặc câu trả lời đúng bằng một điểm. Không có hình phạt cho các câu trả lời không chính xác.

Các kết quả thử nghiệm và kiểm soát về WebRTC cũng cho thấy mức độ hiểu biết tương đối đồng nhất của người dùng. Trong thử nghiệm, người dùng được đánh điểm từ 1 đến 4 cho mỗi câu hỏi, với một điểm cho mỗi câu trả lời phù hợp hoặc đúng. Không có hình phạt nào được áp dụng cho các câu trả lời không chính xác.

Trong các cuộc thử nghiệm này, một câu hỏi liên quan đến WebRTC đã được đặt. WebRTC (Web Real-Time Communication) là một công nghệ tiên tiến cho phép truyền tải dữ liệu âm thanh, video và tin nhắn trong thời gian thực thông qua trình duyệt web. Nó được phát triển bởi World Wide Web Consortium (W3C) và Internet Engineering Task Force (IETF).

Kết quả thử nghiệm cho thấy rằng hầu hết người tham gia đã có hiểu biết đúng về WebRTC, với hơn 80% người tham gia đưa ra câu trả lời chính xác. Điều đáng chú ý là mức độ hiểu lầm thấp nhất trong các danh mục đã được đánh giá là Chức năng, với chỉ có 52% người tham gia trả lời đúng. Tuy nhiên, tổng thể, cả hai nhóm đối chứng và nhóm thử nghiệm đạt được điểm trung bình tương tự, ở mức 3,00/4.

Như vậy, kết quả thử nghiệm cho thấy sự đồng nhất trong sự hiểu biết của người dùng về WebRTC, và phần lớn người tham gia đã có hiểu biết đúng về công nghệ này. Điều này chỉ ra sự thành công trong việc cung cấp thông tin và giải thích về WebRTC trong quá trình thử nghiệm và kiểm soát.

|  |  |
| --- | --- |
| **Nhóm** | **Điểm (trên 4)** |
| Khảo sát 1 (Kiểm soát) | 3.00 |
| Khảo sát 2 (Thử nghiệm) | 3.00 |

SEQ có kết quả thuyết phục hơn một chút, kết quả chính là nhóm thử nghiệm có xếp hạng 22% khó khăn cho nhiệm vụ tổng thể, so với 8,3% đối với nhóm kiểm soát. Không ai trong nhóm thử nghiệm gọi nhiệm vụ này là rất dễ dàng.

|  |  |  |
| --- | --- | --- |
| **Nhìn chung về nhiệm vụ** | **Khảo sát 1(Kiểm soát)** | **Khảo sát 2(Thử nghiệm)** |
| Rất dễ dàng | 8.3% | 0% |
| Dễ dàng | 66.7% | 77.8% |
| Bình thường | 16.7% | 0% |
| Khó khăn | 8.3% | 22.2% |
| Rất khó | 0% | 0% |

**Biểu đồ 5.2 Khảo sát nhìn chung về SEQ**

**Phụ lục**

1. Phương pháp điều trị thử nghiệm và kiểm soát

2. Khảo sát đầy đủ

3. Dữ liệu từ khảo sát

# Chương 6. Kết luận và hướng phát triển

## 6.1. Kết quả đạt được

Với yêu cầu của đề tài xây dựng website video call, chat real-time đã thu được một số kết quả sau:

* + Tìm hiểu được những nội dung cơ bản của WebbRTC và Video Streaming như: các chuẩn giao thức, APIs trong WebRTC, cách thức vượt NAT trong WebRTC,HLS.
  + Nghiên cứu sâu về báo hiệu, vai trò của báo hiệu và các quá trình trong báo hiệu Video Streaming.
  + Khảo sát và đánh giá các thư viện/công nghệ sử dụng trong WebRTC, các hướng tiếp cận sử dụng thư viện WebRTC cũng như Video Streaming.
  + Nghiên cứu và sử dụng thư viện EasyRTC trong việc xây dựng ứng dụng Peer-to-Peer tương tác thời gian thực.
  + Phân tích yêu cầu, thiết kế ứng dụng, cài đặt thử nghiệm ứng dụng sử dụng các công nghệ WebRTC và Video Streaming.

## 6.2. Một số hướng phát triển

Qua nghiên cứu Video Streaming và thử nghiệm ứng dụng, tôi nhận thấy Video Streaming là công nghệ rất tiềm năng, đặc biệt hiệu quả khi triển khai nhanh những ứng dụng đòi hỏi tương tác thời gian thực giữa các tình duyệt với tính đơn giản khi cài đặt, dễ sử dụng với người dùng. Các hạn chế của Video Streaming như chưa được hỗ trợ bởi IE của Microsoft, Safari của Apple hiện tại đã có giải pháp cài đặt thêm các Video Streaming plugin, dù như vậy nó không đúng theo tiêu chí là không plugin mà Video Streaming hướng đến.

Vì vậy, Video Streaming vẫn đang tiếp tục được nghiên cứu chuẩn hóa (bản update mới nhất là vào tháng 9/2016), tiếp tục phát triển, tương lai ứng dụng Video Streaming sẽ có tác động không nhỏ đến ngành công nghiệp web, thậm chí thay thế các ứng dụng cộng tác hiện tại.

Với phân tích đánh giá kết quả thử nghiệm ứng dụng, hướng phát triển tiếp theo của đề tài có thể nghiên cứu, phát triển tiếp những công việc sau:

* + Hoàn thiện các tính năng tương tự như các OTT như hỗ trợ lưu thông tin chat office, cảnh báo notification.
  + Bổ sung khả năng resumable cho việc gửi/nhận file.
  + Nghiên cứu khả năng phát triển tính năng chia sẻ màn hình – screen sharing. Đến thời điểm hiện tại mới chỉ có trình duyệt Chrome hỗ trợ để ứng dụng có thể phát triển tính năng này, về bản chất là chụp ảnh màn hình liên tục và gửi cho trình duyệt đầu xa.
  + Nghiên cứu thêm về cách cài đặt tối ưu hiệu năng từng chức năng như máy chủ EasyRTC, máy chủ báo hiệu, lựa chọn phương án tối ưu trong trường hợp số lượng người dùng lên đến hơn 5000 cán bộ công nhân viên.

# TÀI LIỆU THAM KHẢO

1. Alan B.Johnson, Daniel C.Burnett (2014), APIs and RTCWEB Protocols of the HTML5 Real-Time Web, Digital Codex LLC

2. Salvatore Loreto, Simon Pietro Romano (2014), Real-time Communication with Video Streaming, O’Reilly, USA

3. Andrii Sergiienko (2014), Video Streaming Blueprints, Packt Publishing Ltd, UK

4. Ilya Grigorik (2015), High Performance Browser Networking, O’Reilly Media.

5. Altanai (2014), Video Streaming Intergrator’s Guide, Packt Publishing Ltd, UK

6. Video Streaming for Enterprises

7. Tsahi Levent-Levi (2013), Video Streaming for Business People: Unraveling the challenges and opportunities of the Video Streaming ecosystem

8. Dan Ristic (2015), Learning Video Streaming, Packt Publishing Ltd, UK

9. Rob Manson (2013), Getting Started with Video Streaming, Packt Publishing Ltd, UK

10. Video Streaming Architecture, [https://Video Streaming.org/architecture](https://webrtc.org/architecture). Thời gian truy cập: 11-09-2016

11. RFC 1631 - The IP Network Address Translator (NAT), 1994, <https://tools.ietf.org/html/rfc1631>.

12. RFC 6716 - Definition of the Opus Audio Codec, 2012 <https://tools.ietf.org/html/rfc6716>.

13. RFC 5245, Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols, 2012, <https://tools.ietf.org/html/rfc5245>.

14. RFC 5389, Session Traversal Utilities for NAT (STUN), 2008, <https://tools.ietf.org/html/rfc5389>.

15. RFC 4960, Stream Control Tranmission Protocol (SCTP), 2007, <https://tools.ietf.org/html/rfc4960>.

16. RFC 4347, Datagram Transport Layer Security (DTLS), 2006 <https://tools.ietf.org/html/rfc4347>.

17. RFC 3711, The Secure Real-time Transport Protocol (SRTP), 2004, <https://www.ietf.org/rfc/rfc3711.txt>.

18. RFC 4566, SDP: Session Description Protocol, 2006, <https://tools.ietf.org/html/rfc4566>.

19. RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2, 2008, <https://tools.ietf.org/html/rfc5246>.

20. RFC 5128, State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs), 2008, <https://tools.ietf.org/html/rfc5128>.

21. RFC 5766, Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN), 2010, https://tools.ietf.org/html/rfc5766

22. EasyRTC website, <https://easyrtc.com/docs/browser/easyrtc.php> , Thời gian truy cập: 11-09-2016.

23. <https://www.pkcsecurity.com/blog> Thời gian truy cập 11-10- 2016

24. RFC 3264, An Offer/Answer Model with the Session Description Protocol (SDP), 2002, <https://tools.ietf.org/html/rfc3264> .

25. Javascript Session Establishment Protocol draft-ietf-rtcweb- jsep version 16, 20-09-2016, [https://tools.ietf.org/html/draft- ietf-rtcweb-jsep-16](https://tools.ietf.org/html/draft-%20ietf-rtcweb-jsep-16).