TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

──────── \* ───────



**BÁO CÁO TIỂU LUẬN**

***ĐỀ TÀI:* TÌM HIỂU WEBRTC THÔNG QUA NODEJS VÀ SOCKET.IO**

Môn Học: CÔNG NGHỆ WEB

Giáo viên hướng dẫn : Đỗ Bá Lâm

Sinh viên thực hiện : Nguyễn Văn Thái-20156445

***Hà Nội, tháng 5 năm 2018***

Menu

[I. Giới thiệu về mô hình webRTC 3](#_Toc514662408)

[1. **Sơ lược lịch sử của WebRTC** 3](#_Toc514662409)

[2. **WebRTC là gì?** 3](#_Toc514662410)

[**3.** **Lợi ích của WebRTC mang lại** 3](#_Toc514662411)

[**4.** **Tính phổ biến và tiềm năng phát triển** 4](#_Toc514662412)

[**5.** **Các ứng dụng** 5](#_Toc514662413)

[II. Chạy webRTC trên server Nodejs 7](#_Toc514662414)

[**1.** **Giới thiệu về Nodejs** 7](#_Toc514662415)

[**2.** **Các đặc tính của NodeJS** 7](#_Toc514662416)

[**3.** **Tại sao lại chọn NodeJs làm server cho webRTC ?** 8](#_Toc514662417)

[III. Cài đặt ứng dụng Real-time với socket và socket.io 9](#_Toc514662418)

[1. **Sơ lược về Socket và SocketIO** 9](#_Toc514662419)

[2. **Các cấu hình cơ bản:** 10](#_Toc514662420)

[**a.** **Tích hợp Socket.io vào ứng dụng NodeJs.** 10](#_Toc514662421)

[**b.** **Một số thông số và lệnh cơ bản trong socket.io** 10](#_Toc514662422)

[IV. Cài đặt demo thao tác cơ bản với socket.io và nodejs. 11](#_Toc514662423)

[1. **Các bước chuẩn bị.** 11](#_Toc514662424)

[2. **Thực hiện.** 11](#_Toc514662425)

[V. Tạo ứng dụng live stream. 16](#_Toc514662426)

[1. **Giới thiệu ứng dụng** 16](#_Toc514662427)

[2. **Giao diện cơ bản của trang web.** 16](#_Toc514662428)

[3. **Source code của trang web** 18](#_Toc514662429)

[4. **Deloy on heroku server** 19](#_Toc514662430)

[VI. Kết luận 20](#_Toc514662431)

1. **Giới thiệu về mô hình webRTC**

[WebRTC](https://tinhte.vn/tags/webrtc/) ([Web Real-Time Communications](https://tinhte.vn/tags/web-real-time-communications/)) là một tập hợp các hàm lập trình dùng cho việc liên lạc [thời gian thực](https://tinhte.vn/tags/thoi-gian-thuc/) bằng video, âm thanh cũng như các loại dữ liệu khác. WebRTC có thể giúp chúng ta gọi điện video ngay trong [trình duyệt](https://tinhte.vn/tags/trinh-duyet/) mà không cần đăng kí tài khoản, cũng không cần cài thêm plugin gì phức tạp, ngoài ra chúng còn được xài để phát triển game chơi trực tiếp trong trình duyệt và rất nhiều loại ứng dụng khác. Trong bài bên dưới mời các bạn tìm hiểu kĩ hơn WebRTC là gì, người ta đang dùng nó ra sao và những trở ngại nào đang hiện hữu với chuẩn vẫn còn rất mới này.

1. **Sơ lược lịch sử của WebRTC**

Ý tưởng phát triển WebRTC được nhóm kỹ sư chịu trách nhiệm cho [Google](https://tinhte.vn/tags/google/) Hangouts đưa ra từ tận năm 2009. Vào thời gian đó, để truyền tải video, hình ảnh trên web thì người ta thường phải xài đến Flash. Nhóm kỹ sơ Hangouts lại không muốn sử dụng công nghệ này, và họ bắt đầu tự làm một chuẩn riêng cho mình. Đến năm 2010, Google thâu tóm hai công ty On2 và Global IP Solutions (GIPS) để lấy công nghệ truyền dữ liệu thời gian thực làm nền tảng cho WebRTC về sau.  
 Vào tháng 5/2011, Google ra mắt một dự án nguồn mở dành cho việc giao tiếp thời gian thực giữa trình duyệt với nhau, và từ lúc này dự án mang tên WebRTC. Song song đó, Hiệp hội World Wide Web (W3C) và Hiệp hội Kĩ sư quốc tế (IETF) cũng đang phát triển một số giao thức để dùng cho việc việc kết nối thời gian thực, thế nên họ bắt tay nhau tiếp tục hoàn thiện để rồi quyết định kết hợp chung vào WebRTC.  
 Đến 27/10/2011, W3C ra mắt bản nháp đầu tiên của WebRTC. Tháng 11/2011, [Chrome](https://tinhte.vn/tags/chrome/) 23 ra mắt, trở thành trình duyệt đầu tiên có tích hợp WebRTC ngay từ bên trong. Và tính đến thời mà mình viết bài này thì WebRTC vẫn còn đang tiếp tục được phát triển chứ chưa hoàn thiện một cách chính thức.

1. **WebRTC là gì?**

**WebRTC** (**Web Real-Time Communication**) là một web API được phát triển bởi World Wide Web Consortium (W3C), khả năng hỗ trợ trình duyệt (browser) giao tiếp với nhau thông qua VideoCall, VoiceCall hay transfer data "Peer-to-Peer" (P2P) mà không cần browser phải cài thêm plugins hay phần mềm hỗ trợ nào từ bên ngoài.

1. **Lợi ích của WebRTC mang lại**

Ở góc độ nhà phát triển, nếu không có WebRTC, việc tạo ra ứng dụng RTC đòi hỏi phải mất nhiều công sức từ việc lấy dữ liệu từ thiết bị camera, microphone đến việc thiết lập phiên, xử lý tín hiệu, truyền tín hiệu, … Nhưng với WebRTC, tất cả công việc để tạo ra một cuộc hội thoại chỉ nằm trong vài chục dòng lệnh. Việc phát triển ứng dụng với chức năng gọi điện, video chat và chia sẻ file,.. là rất đơn giản khi dùng WebRTC kết hợp giữa JavaScript và HTML5.

Ở góc độ người sử dụng, sử dụng WebRTC chỉ cần thông qua trình duyệt Web. Tính sẵn sàng cao cho phép thực hiện cuộc gọi mà không cần đăng ký tài khoản hay cài đặt thêm thành phần nào ngoài một trình duyệt có hỗ trợ WebRTC. Ví dụ, hai người dùng chỉ cần truy cập vào cùng một đường dẫn web để gọi video với nhau sử dụng trình duyệt Google Chrome hay Mozilla Firefox.

1. **Tính phổ biến và tiềm năng phát triển**

Điểm đột phá của WebRTC là ta có thể tham gia cuộc hội thoại ngay trên trình duyệt mà không cần cài thêm bất cứ một phần mềm hay plugin nào khác. Nó đang được chuẩn hóa ở cấp độ API của W3C và cấp độ giao thức của IETF, được hỗ trợ bởi các trình duyệt Google Chrome, Mozilla Firefox và Opera trên PC và Android. Ngoài ra WebRTC còn được hỗ trợ trên Chrome OS.

Tính đến thời điểm hiện tại, đã có trên 1 tỷ thiết bị đầu cuối hỗ trợ WebRTC, dự báo tăng lên 4 tỷ vào năm 2016, trong đó có khoảng 1,5 tỷ người dùng thường xuyên. WebRTC có thể hoạt động trên bất cứ thiết bị nào có cài một trong các trình duyệt hỗ trợ WebRTC.



1. **Các ứng dụng**

Với các ưu điểm kể trên, ta có thể sử dụng WebRTC trong rất nhiều ứng dụng khác nhau để mang tới một cách kết nối đơn giản tiện lợi. WebRTC có thể tạo ra các webphone, ứng dụng hội nghị trực tuyến, chăm sóc khách hàng, hướng dẫn trực tuyến, giáo dục, y tế, chăm sóc sức khỏe, tư vấn bảo hiểm, game, …

Ngày nay, hầu hết các công ty đều có Website cung cấp các thông tin và thường là điểm đầu tiên giúp khách hàng liên hệ với doanh nghiệp. Nếu khách hàng không tìm thấy đủ thông tin trên Website hoặc cần phản ánh một vấn đề liên quan đến sản phẩm thì họ chỉ có cách gọi điện thoại. Tuy nhiên, cách này có thể làm mất các ngữ cảnh trên website hay không mô tả rõ ràng được vấn đề. WebRTC có thể ứng dụng để giúp thực hiện cuộc gọi ngay trên Website và cho phép chia sẻ màn hình của khách hàng, từ đó các nhân viên hỗ trợ có thể nhanh chóng nhận ra và giải quyết vấn đề của khách hàng.

Một ví dụ khác, WebRTC được dùng trong chăm sóc sức khỏe, trang [www.wello.com](http://www.wello.com/) cung cấp kênh thông tin cho việc giảng dạy cũng như tham gia vào các khóa học rèn luyện sức khỏe tại nhà thông qua hướng dẫn trực tiếp bằng video. Giảng viên hướng dẫn trực tiếp cho các học viên, có thể quan sát họ thực hiện các động tác và chỉnh sửa khi họ thực hiện sai. WebRTC giúp việc luyện tập trở nên thoải mái và tiện lợi hơn.

1. **Chạy webRTC trên server Nodejs**
2. **Giới thiệu về Nodejs**

* **NodeJS là gì?**

NodeJS là một mã nguồn được xây dựng dựa trên nền tảng Javascript V8 Engine, nó được sử dụng để xây dựng các ứng dụng web như các trang video clip, các forum và đặc biệt là trang mạng xã hội phạm vi hẹp. NodeJS là một mã nguồn mở được sử dụng rộng bởi hàng ngàn lập trình viên trên toàn thế giới. NodeJS có thể chạy trên nhiều nền tảng hệ điều hành khác nhau từ WIndow cho tới Linux, OS X nên đó cũng là một lợi thế. NodeJS cung cấp các thư viện phong phú ở dạng Javascript Module khác nhau giúp đơn giản hóa việc lập trình và giảm thời gian ở mức thấp nhất.

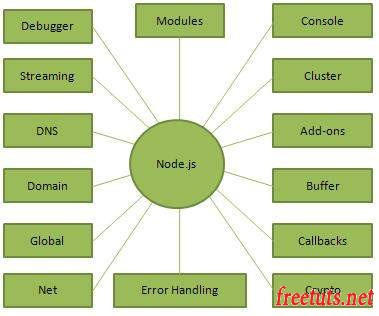
Khi nói đến NodeJS thì phải nghĩ tới vấn đề Realtime. Realtime ở đây chính là xử lý giao tiếp từ client tới máy chủ theo thời gian thực. Giống như khi bạn lướt Facebook thì mỗi khi bạn comment hay like một topic nào đó thì ngay lập tức chủ topic và những người đã comment trên đó sẽ nhận được thông báo là bạn đã comment. Nếu ban nghĩ Facebook đang sử dụng [Ajax](https://freetuts.net/hoc-php/hoc-ajax) thì bạn đã sai rồi nhé, nếu họ sử dụng Ajax thì Server họ sẽ chết ngay lập tức bởi các request gửi lên với số lượng tính theo tấn :3.

1. **Các đặc tính của NodeJS**

Qua phần tìm hiểu **NodeJS là gì** mình có giới thiệu một đặc tính rất quan trọng đó là Realtime, tuy nhiên vẫn còn khá nhiều đặc tính mà bạn cần phải biết trước khi học NodeJS.

* **Không đồng bộ**: Tất cả các API của NodeJS đều không đồng bộ (none-blocking), nó chủ yếu dựa trên nền của NodeJS Server và chờ đợi Server trả dữ liệu về. Việc di chuyển máy chủ đến các API tiếp theo sau khi gọi và cơ chế thông báo các sự kiện của Node.js giúp máy chủ để có được một phản ứng từ các cuộc gọi API trước (Realtime).
* **Chạy rất nhanh**: NodeJ được xây dựng dựa vào nền tảng V8 Javascript Engine nên việc thực thi chương trình rất nhanh.
* **Đơn luồng nhưng khả năng mở rộng cao**: Node.js sử dụng một mô hình luồng duy nhất với sự kiện lặp. cơ chế tổ chức sự kiện giúp các máy chủ để đáp ứng một cách không ngăn chặn và làm cho máy chủ cao khả năng mở rộng như trái ngược với các máy chủ truyền thống mà tạo đề hạn chế để xử lý yêu cầu. Node.js sử dụng một chương trình đơn luồng và các chương trình tương tự có thể cung cấp dịch vụ cho một số lượng lớn hơn nhiều so với yêu cầu máy chủ truyền thống như Apache HTTP Server.
* **Không đệm**: NodeJS không đệm bất kì một dữ liệu nào và các ứng dụng này chủ yếu là đầu ra dữ liệu.
* **Có giấy phép**: NodeJS đã được cấp giấy phép bởi [MIT License](https://raw.githubusercontent.com/joyent/node/v0.12.0/LICENSE).

Sau đây là sơ đồ về các thành phần quan trọng trong NodeJS mà trong series này chúng ta sẽ tìm hiểu dần (hình sưu tầm từ tutorialspoint).



1. **Tại sao lại chọn NodeJs làm server cho webRTC ?**

**Ưu điểm đễ thấy của Nodejs là:**

* Tối ưu hóa thời gian thực hiện tiến trình.
* Có khả năng mở rộng trong các ứng dụng web với nhiều hoạt động I/O liên tục
* Phù hợp để xây dựng các ứng dụng real time đảm bảo việc độ trễ thời gian xử lý hành động là nhỏ nhất.
* Dễ dàng để xây dựng các ứng dụng real-time.
* Hiệu năng cao.

Với những ưu điểm trên NodeJs rất phù hợp với các ứng dụng liên quan đến real-time trong đó có webRTC.

1. **Cài đặt ứng dụng Real-time với socket và socket.io**
   1. **Sơ lược về Socket và SocketIO**

Trước tiên, Socket là 1 công nghệ. Đừng nhầm lẫn giữa [Socket.IO](http://socket.io/) và Socket. [Socket.IO](http://socket.io/) không phải là mô hình Socket duy nhất hiện nay, và cũng không phải là mô hình web socket duy nhất hiện nay.Socket là cách bạn tổ chức mô hình client-server để một trong 2 bên luôn trong tình trạng sẵn sàng trả lời bên kia và ngược lại. Để đảm bảo việc này, kết nối giữa Client và Server phải ở trạng thái “keep-alive” và phải luôn xảy ra quá trình đồng bộ giữa Client-Server. Socket sẽ mang lại khả năng trả lời tức thì từ một trong 2 bên khi bên kia đưa ra một sự kiện, thay vì phải thực thi lại một loạt các thủ tục kết nối phức tạp như trước, và ứng dụng của bạn sẽ trở thành ứng dụng thời gian thực ví dụ: Yahoo Messenger, Skype v.v… đều là các ứng dụng được xây dựng theo mô hình Socket.

Trong lập trình web trước đây, việc xây dựng client-server theo mô hình socket phải thông qua các phần mềm thứ . Vì mô hình socket không phù hợp với các ngôn ngữ lập trình Server như: PHP, [ASP.NET](http://asp.net/), JSP v.v… Các ngôn ngữ này luôn làm việc theo cách: Die ngay connection khi Server trả lời Client xong. Tuy nhiên, mình nhấn mạnh: Chúng ta có thể làm được web-socket với bất kỳ ngôn ngữ lập trình nào. Chỉ có điều, với các ngôn ngữ cũ, việc làm này cần bạn phải am hiểu các giao thức http, tcp; hiểu thế nào là 1 request header, v.v…

Node.js là một ngôn ngữ mới, xây dựng thuần túy bằng javascript. Đây là một điểm lợi thế của Node.js để lập trình web-socket:

Thứ nhất: javascript là ngôn ngữ lập trình hướng sự kiện, mà trong lập trình thời gian thực, cách tiếp cận bằng lập trình sự kiện là cách tiếp cận khôn ngoan nhất.

Thứ hai: Node.js chạy non-blocking việc hệ thống không phải tạm ngừng để xử lý xong một request sẽ giúp cho server trả lời client gần như ngay tức thì.

Thứ ba: lập trình socket yêu cầu bạn phải xây dựng được mô hình lắng nghe - trả lời từ cả 2 bên. Nói khác đi, vai trò của client và server phải tương đương nhau, mà client thì chạy bằng javascript, nên nếu server cũng chạy bằng javascript nữa, thì việc lập trình sẽ dễ dàng và thân thiện hơn.

Chính vì những đặc điểm này, [socket.io](http://socket.io/) ra đời. Tuy nhiên, khi bạn thực sự am hiểu về Node.js, http request header, bạn hoàn toàn có thể viết một socket cho riêng mình.

* 1. **Các cấu hình cơ bản:**
     1. **Tích hợp Socket.io vào ứng dụng NodeJs.**

**Thư viện Socket.io được tích hợp sẵn vào trong gói npm của NodeJs vì thế để tích hợp socket.io vào ứng dung rất là đơn giản chỉ cần chạy lệnh :**



* + 1. **Một số thông số và lệnh cơ bản trong socket.io**
* socket.id: id của một socket client.
* Socket.rooms:

là tập hợp các client đang kết nối vào 1 room được định danh bằng name.ví dụ:

io.on('connection', (socket) =>

socket.join('room 237', () => {

let rooms = Object.keys(socket.rooms);

console.log(rooms); // [ <socket.id>, 'room 237' ]

});

});

* Socket.leave([room, callback])

Lệnh thoát khỏi room.

* socket.emit('nameOfSend', anyThing);

Là lệnh gửi hành động của socket.io( có thể gửi từ server tới client và ngược lại)

trong đó: nameOfSend là tên của đường truyền để ở phía nhận có thể nhận biêt.

anyThing là tham số truyền vào( có thể là số, string, obj, images,video...) để gửi đi.

* socket.on('nameOfSend', function(data){

//do somethings

});

Là lệnh nhận thao tác từ hàm sockit.emit().

* socket.to('others').emit('an event', { some: 'data' });

Lệnh gửi emit tới các client trong room(có thể gửi 1 lúc nhiều room)

* Io.sockets.emit('nameOfSend', anyThing);

Lệnh dành riêng cho phía server để gửi cho tất cả các client đang kết nối.

* socket.broadcast.emit('nameOfSend', anyThing);

Lệnh dành riêng cho phía server để gửi cho tất cả các client đang kết nối trừ client đang thao tác.

-----------------------------------------------------------

Xem thêm tại: <https://socket.io/docs/>

1. **Cài đặt demo thao tác cơ bản với socket.io và nodejs.**
   1. **Các bước chuẩn bị.**

* Cài đặt IDE viết code subline Text 3
* Cài đặt Nodejs và tạo ứng dụng nodejs
* thêm thư viện socket.io vào ứng dụng nodejs
* viết code.
  1. **Thực hiện.**
* **Cài đặt subline Text 3.**

**Subline Text 3** là công cụ viết code miến phí, đơn giản, nhẹ, dễ sử dụng và co giao diện đẹp mắt. Chúng ta có thể download trên: <https://www.sublimetext.com/>

* **Cài đặt và tạo ứng dụng nodejs**

- Vào trang chủ của nodejs để tải nodejs về: <https://nodejs.org/en/download/>

- tạo project nodejs

* **Thêm thư viện socket.io**: npm install socket.io –save
* **Viết Code.**

**B1:** Tạo server/js để chạy ứng dụng.



Bên trong server.js có hàm lắng nghe các client kết nối.

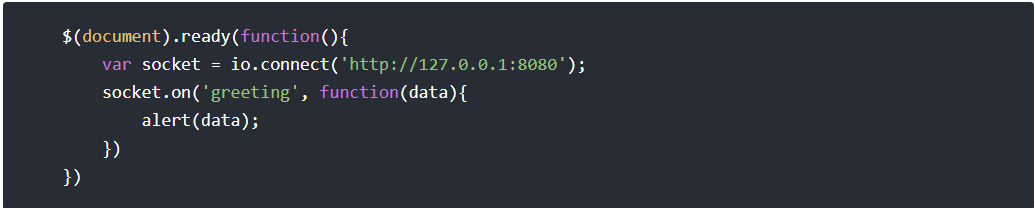


**B2**: Tạo client( là trang index.html)

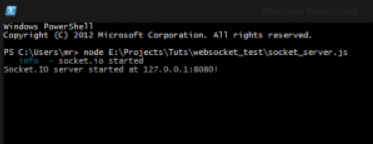
chèn đoạn mã cực-kỳ-cơ-bản này vào file index.html và lưu lại(\*):



Đây chính là cách bạn phát một sự kiện từ server đến client. Phương thức socket.emit(string event, object data) sẽ thực hiện việc phát 1 sự kiên về client và đồng thời gửi dữ liệu data về client. Bây giờ là phần client, bạn chèn đoạn mã sau vào file 'socket\_client.js'(\*):



Sau đó, chạy file socket\_server.js : node server.js



và mở file index.html bằng một browser bất kỳ (tuy nhiên không khuyến khích IE cho lắm và khuyến khích GoogleChrome). Bạn sẽ có kết quả:

Đó là cách server phát sự kiện, client nhận sự kiện. Bây giờ là chiều ngược lại, hoàn toàn tương tự:

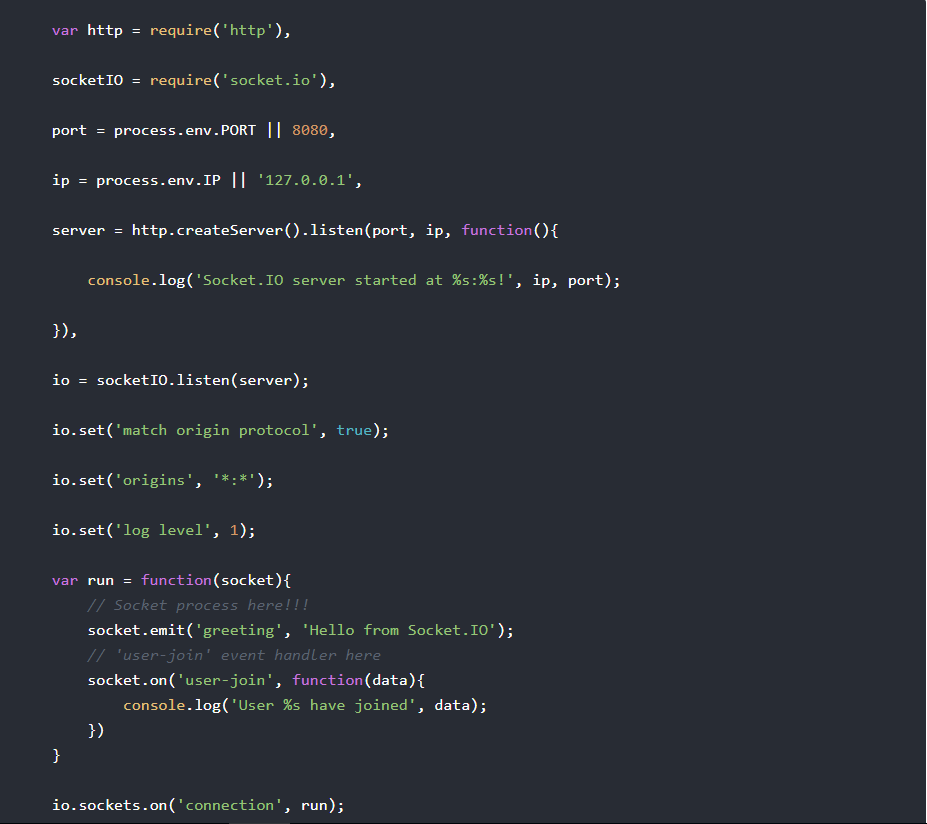
Trước tiên bạn sửa file index.html thành như sau:



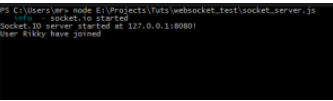
và file 'socket\_client.js' thành:



Vậy là client đã gửi event 'user-join' lên server, kèm theo đó là dữ liệu chứa tên của người join. Bây giờ, server sẽ lắng nghe event này:



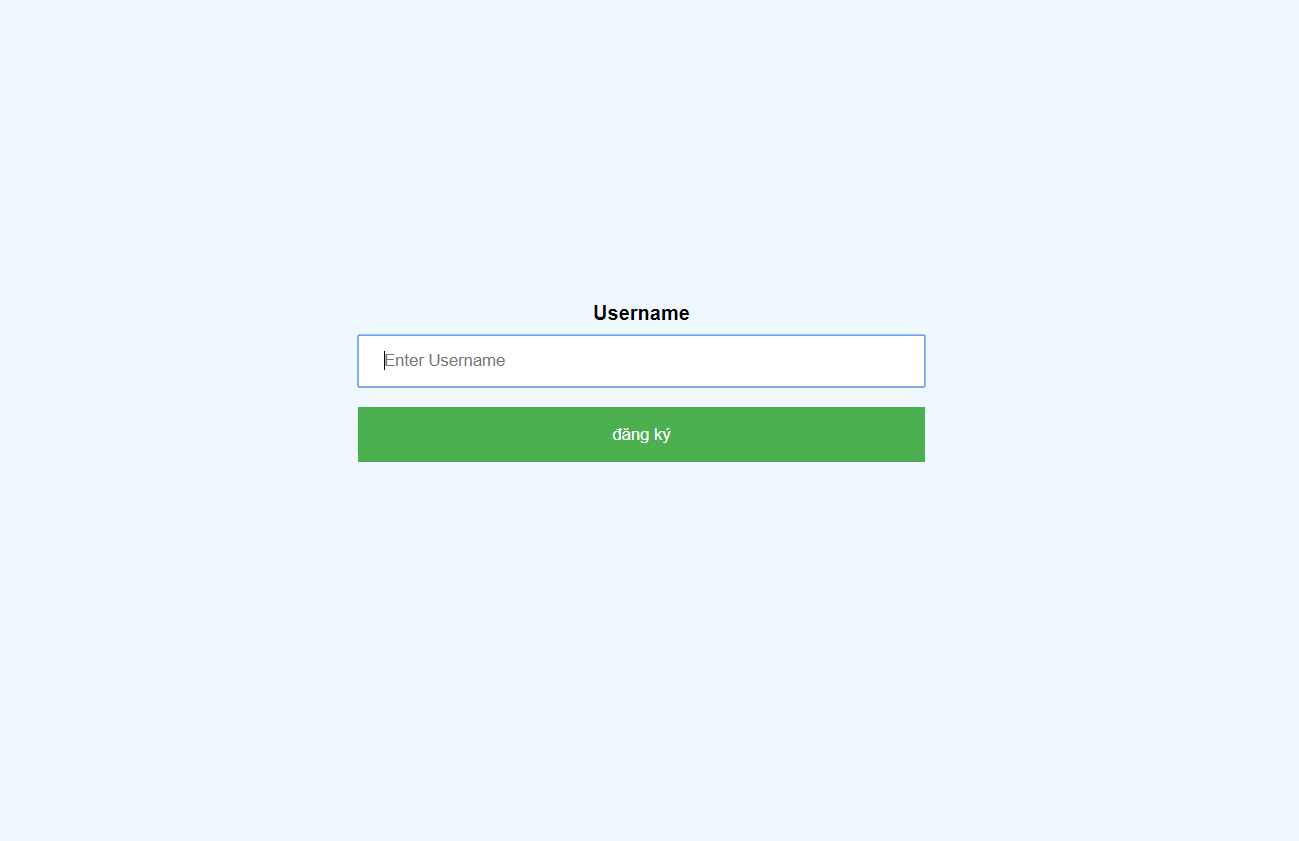
Sau đó, nhập dữ liệu và gửi tới server. Kết quả:



1. **Tạo ứng dụng live stream.**
   1. **Giới thiệu ứng dụng**

Là trang web sử dụng socket.io và nodejs để có thể phát trực tiếp video thông qua webcame cho người dùng khi vào trang web.

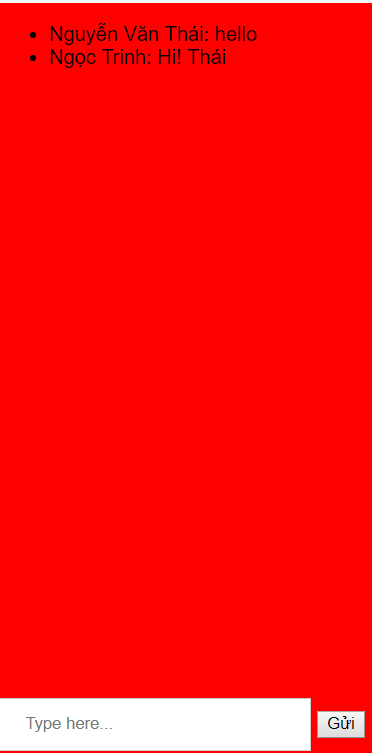
* 1. **Giao diện cơ bản của trang web.**
* Giao diện đăng ký:



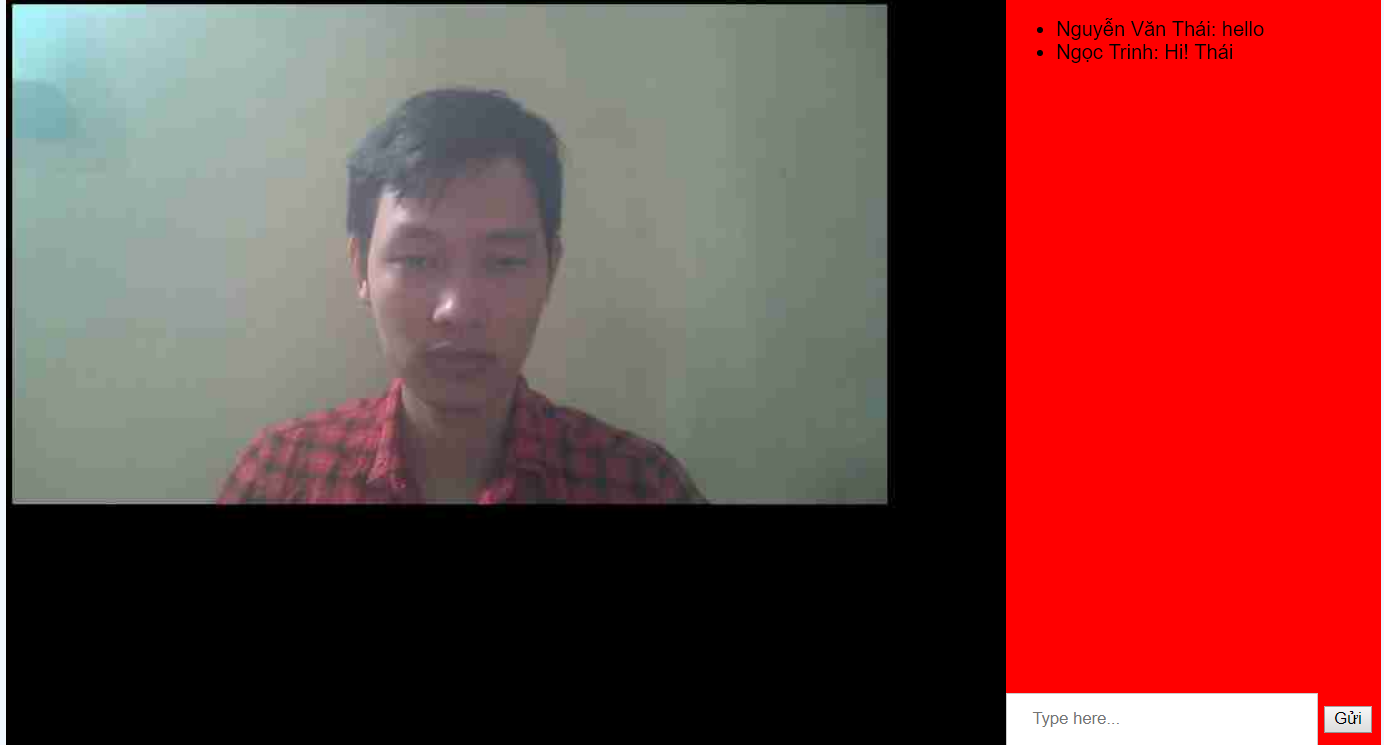
* Đăng Nhập



* Khung chat real-time:



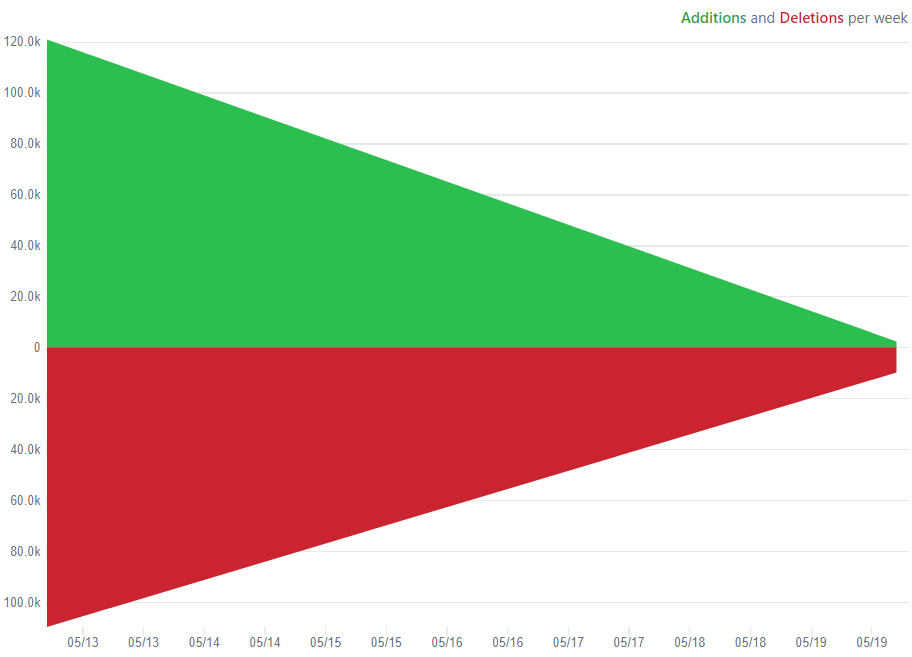
* Video live stream:



* 1. **Source code của trang web**

[**https://github.com/0henri0/web--video-call-by-nodejs-socket-io**](https://github.com/0henri0/web--video-call-by-nodejs-socket-io)

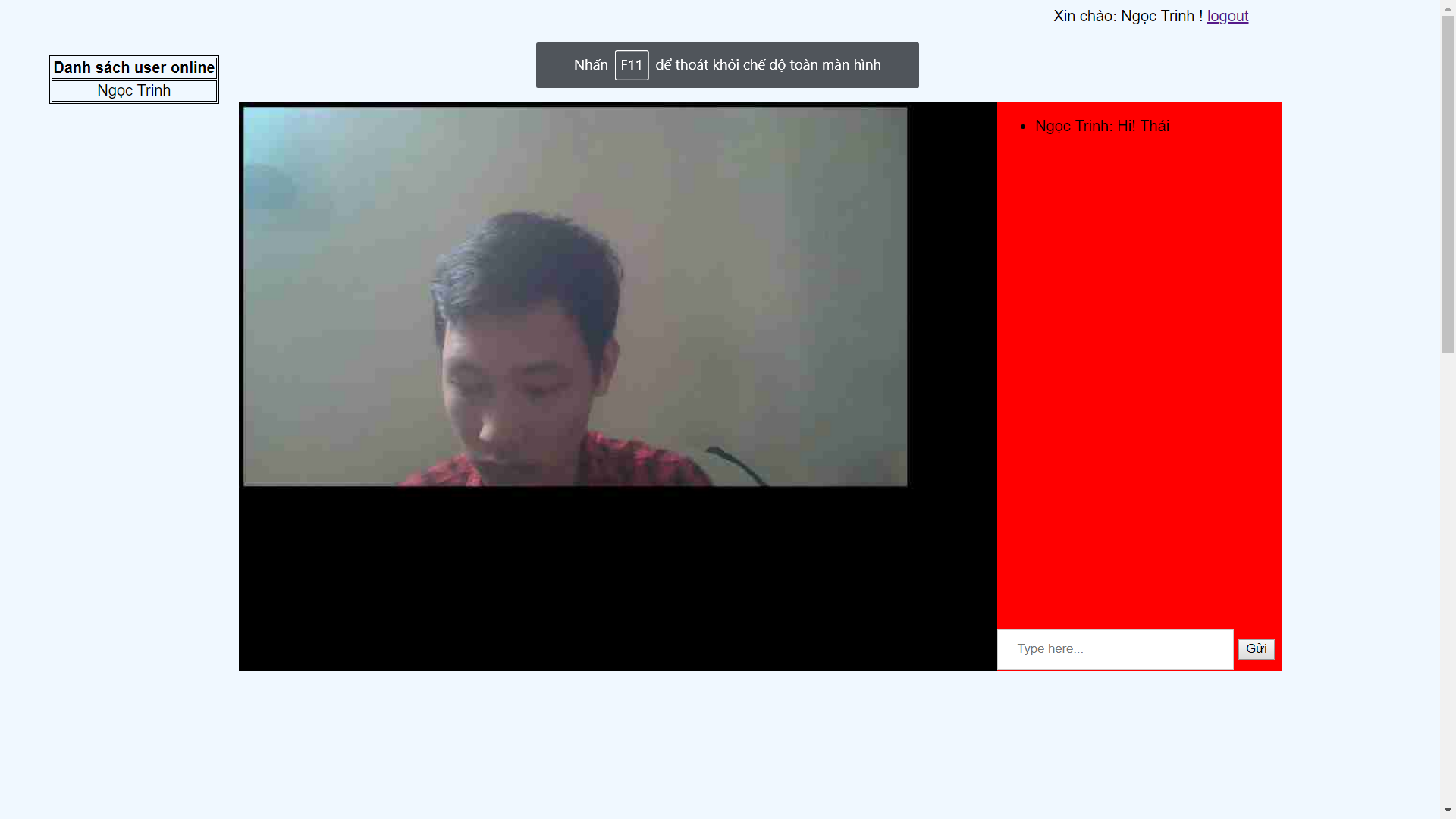
* **Code frequency**



* 1. **Deloy on heroku server**

[**https://webrtcbythaicute.herokuapp.com/#**](https://webrtcbythaicute.herokuapp.com/%23)

**Giao diện hoàn chỉnh của trang web.**



1. **Kết luận**

Hiện nay công nghệ real-time đang phát triển mạnh mẽ với những ứng dụng to lớn. Live stream video chỉ là một phần nhỏ của ứng dung real-time. Công nghệ Nodejs và socket.io làm việc tạo lên một ứng dụng real-time một cách dễ dàng và hiệu quả.