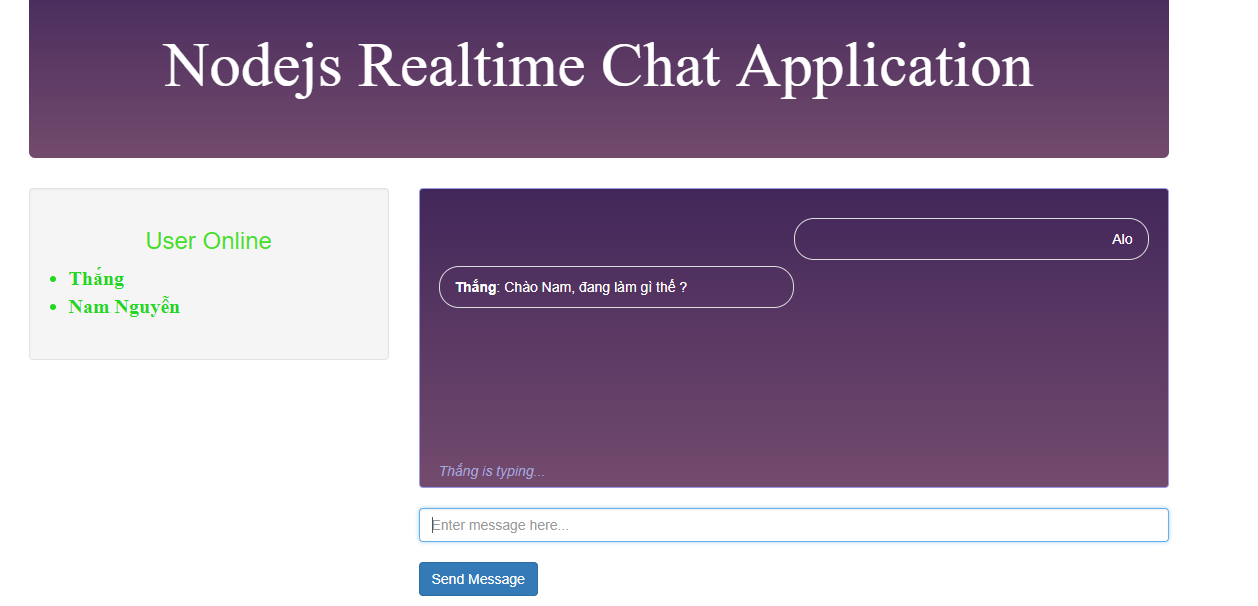
**Xây dựng ứng dụng chat realtime bằng nodejs**



**Demo ứng dụng tại đây:** [**https://realtime-chatting.herokuapp.com/**](https://realtime-chatting.herokuapp.com/)

**SourceCode:** [**https://github.com/Danptit/nodejs\_realtimeChat\_app**](https://github.com/Danptit/nodejs_realtimeChat_app)

**Cài đặt môi trường phát triển**

Khởi tạo ứng dụng bằng cách tạo 1 thư mục , cd vào thư mục đó và tạo file package.json bằng cú pháp: **npm init.**

Tiếp theo chúng ta cài đặt các modules cần thiết cho ứng dụng chat:

**npm install -- save express**

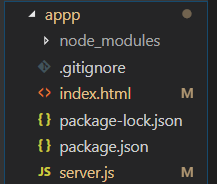
**npm install -- save socket.io**

**Tạo 2 file:**

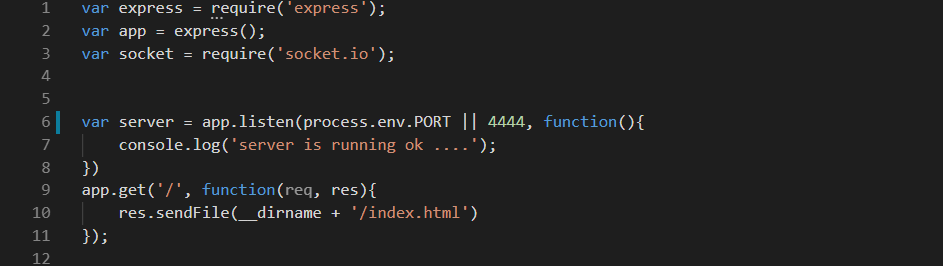
**server.js** để xử lí lắng nghe, emit các sự kiện trao đổi giữa server và client

**index.html**  giao diện trang web và xử lí các hàm lắng nghe, emit tới server trong thẻ <script>

**Cấu trúc thư mục của chúng ta sẽ có dạng như sau :**



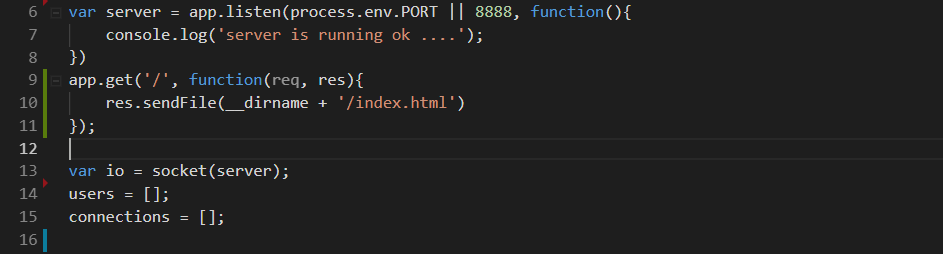
**Server.js:**



Đầu tiên, chúng ta require các module cần dùng: express, socket.io và tạo 1 biến app để sử dụng express

Dòng 6: Chạy 1 server lắng nghe tại địa chỉ : http://localhost:8888/

Dòng 9: Request gửi tới root url sẽ được response về 1 giao diện index.html



Dòng 13: Khai báo 1 biến io để có thể sử dụng các hàm của socket.io library

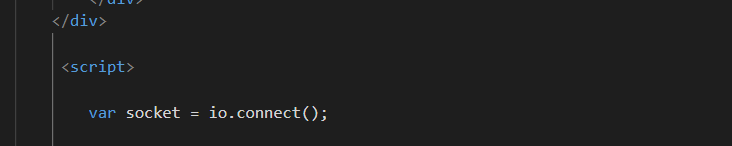
Dòng 14, 15 tạo 2 mảng để lưu trữ tên users chứa tên của user khi connect tới server và mảng connections lưu trữ các connect sau này.

**Index.html:**

Tại file index.html, ta cần phải cài đặt socket.io lên client cái mà sẽ connect tới server, để làm điều đó chúng ta thêm 1 library socket.io bằng 1 thẻ <script> viết trong thẻ <head>



Ta tạo thêm thẻ <script> ngay trước khi đóng thẻ </body> và viết viết các đoạn mã xử lí tại client trong đó:

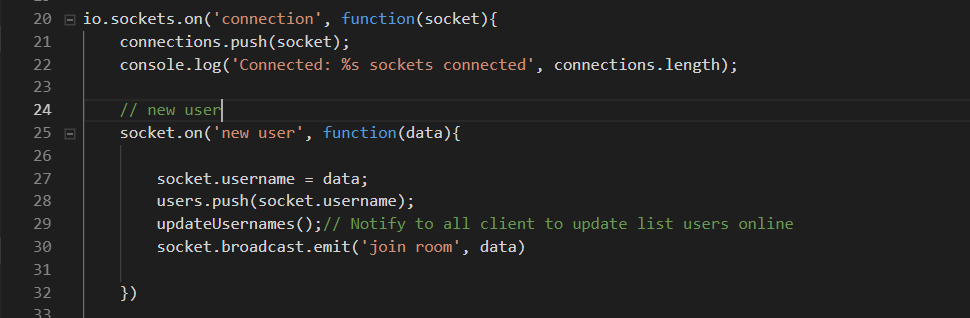


Như vậy tại client chúng ta đã tạo 1 socket và có thể kết nối tới server

**Gửi và nhận data**

Server và client trao đổi dữ liệu với nhau thông qua các events, chúng ta có thể gửi đi và lắng nghe các events đó bằng cú pháp **socket.emit** và **socket.on**

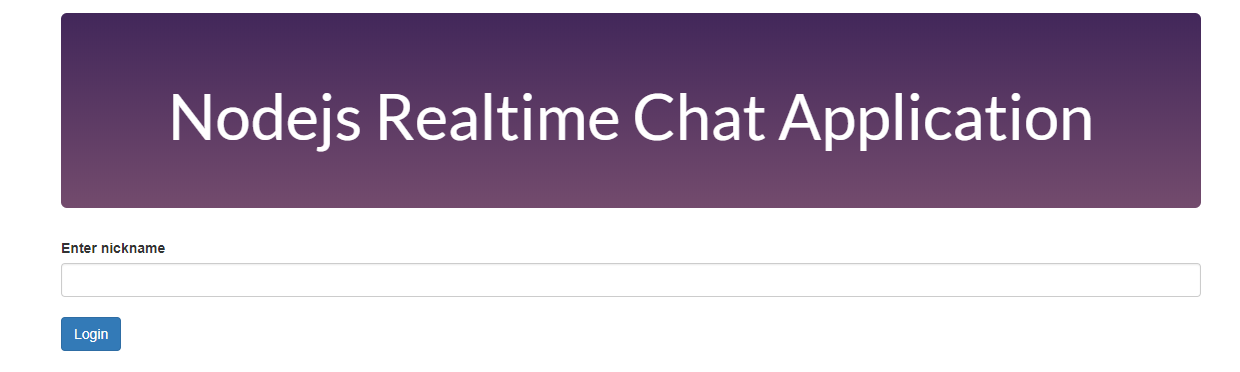
**Server.js:**

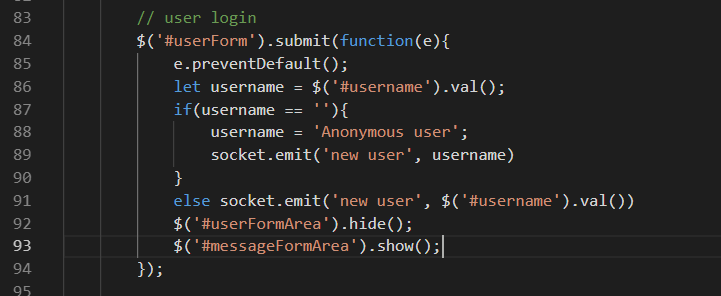


Dòng 20 sẽ lắng nghe các socket kết nối tới server.

Khi các socket kết nối tới server, chúng ta sẽ lưu nó vào mảng connections và dòng 22 sẽ log ra số socket đang kết nối

**Index.html:**

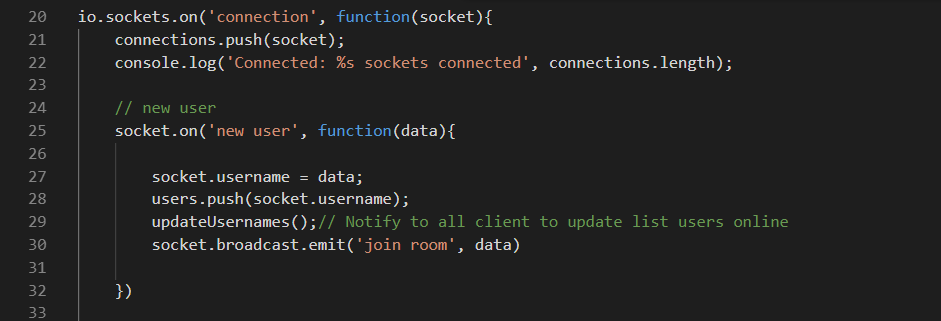




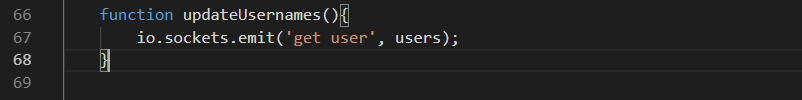
Khi 1 client truy cập vào địa chỉ : <http://localhost:8888/>, giao diện hiện ra yêu cầu user nhập nickname vào form và sau đó ấn enter.

Sau khi user nhập nickname, dòng 89 hoặc 91 sẽ gửi 1 emit : new-user lên server với data là tên user nhập vào

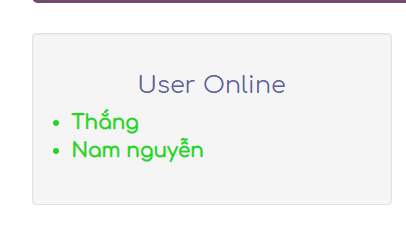
**Server.js**



Tại server, chúng ta sẽ xử lí emit : “new -user” đặt tên cho socket kết nối tới bằng data nhận được từ client, lưu nó vào mảng users, sau đó gọi làm updateUsernames()

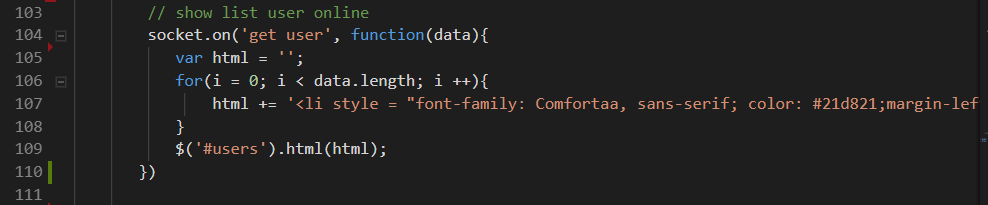


Hàm updateUsernames() sẽ gửi tới tất cả client 1 emit : get user, kèm data là mảng users, client sẽ nhận emit này và cập nhật lại list các user online.

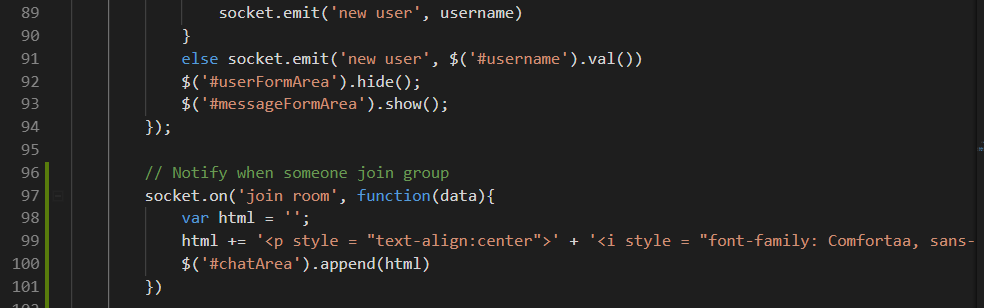


Dòng 30 thực hiện việc emit : “join –room ’ cho tất cả client ngoại trừ socket hiện thời, để thông báo cho các user khác là có người đã tham gia phòng chat

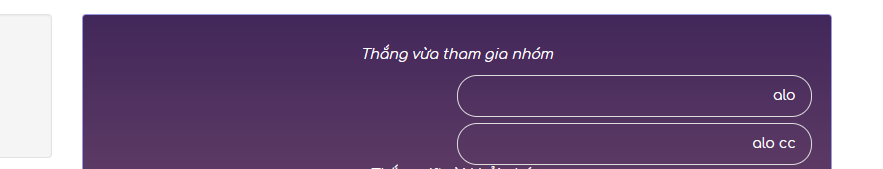
**Index.html**



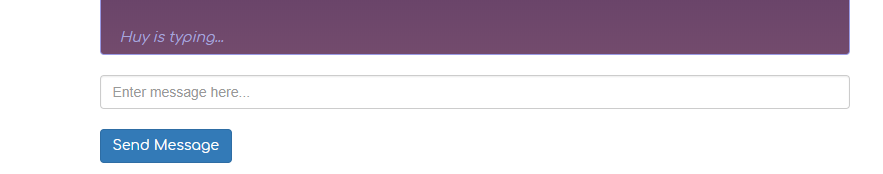
Client lắng nghe emit “get user” từ server với data nhận được là mảng users, ta sẽ tạo các thẻ <li> tương ứng độ dài mảng users

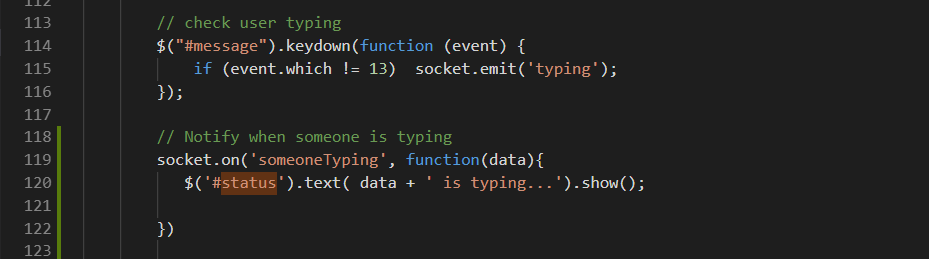


Phía khi nhận được emit joint-room từ server, chúng ta sẽ tạo 1 thẻ <p> + data để hiển thị thông báo có người tham gia cuộc trò chuyện



Tiếp theo chúng ta sẽ xử lí hiệu ứng khi ai đó đang nhập tin nhắn thì thông báo cho các client còn lại

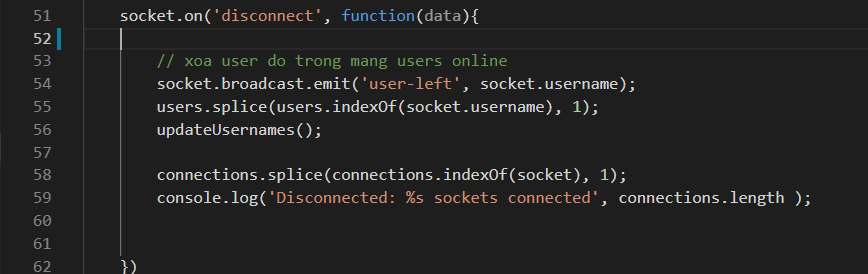




Dòng 113 chúng ra lắng nghe user gõ phím, ngoại trừ phím enter(keycode = 13), ta sẽ bắn 1 emit tới server, server lắng nghe emit này, và gửi về emit : someone typing kèm data là tên của socket đang nhập văn bản, phía client lắng nghe emit này, gán text vào thẻ <p> có id = status với data là tên username của socket đang nhập văn bản

**Xử lí khi socket ngắt kết nối tới server**

**Server.js**



Dòng 51 sẽ lắng nghe khi 1 user nào đó ngừng kết nối tới server, chúng ta thực hiện:

Dòng 54: Thông báo cho các user khác 1 user tên socket.username đã rời khỏi nhóm chat bằng 1 emit : user-left

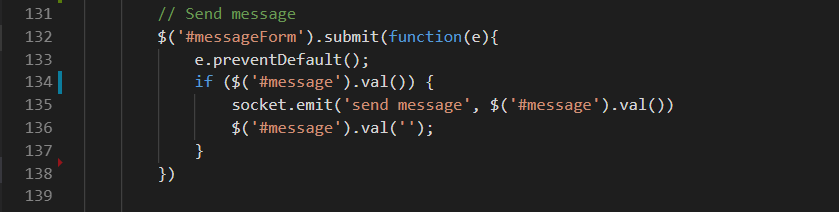
Dòng 55: Xóa user đó trong mảng user,

Dòng 56: Cập nhập lại list user online cho client,

Dòng 58: Xóa socket đó trong mảng connections

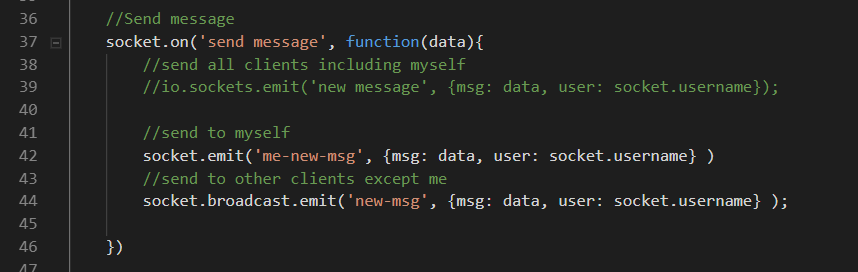
**Xử lí nhắn tin giữa các socket**

**Index.html:**



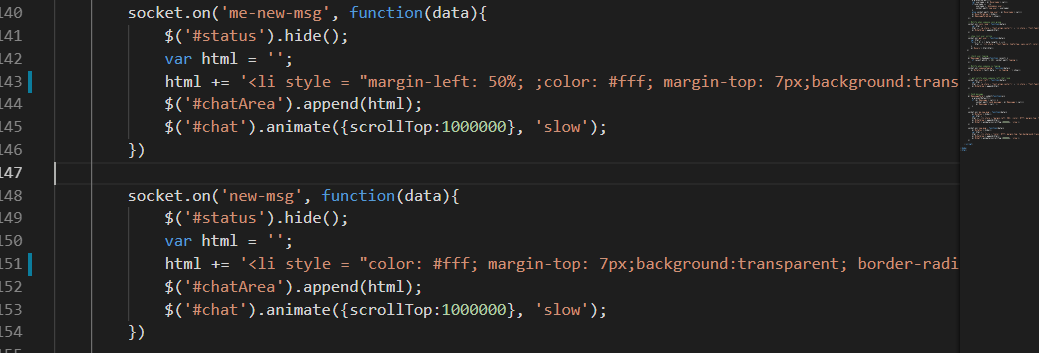
Khi user nhập nội dung tin nhắn, chúng ta sẽ bắn 1 emit với nội dung send-message kèm data là giá trị của ô input, sau đó set text cho ô input bằng đoạn string rỗng

**Server.js**



Tại server sẽ lắng nghe emit: send-message từ client gửi lên, sau đó emit lại cho client 2 emit để client có thể hiển thị giao diện tin nhắn đúng như mình muốn

**Index.html**



Tại client chúng ta nhận được 2 emit từ server, 2 emit này chỉ khác nhau phần giao diện hiển thị, tin nhắn của chính user đó thì chúng ta thêm style margin-left 50% để dịch sang bên trái

