NODE.JS

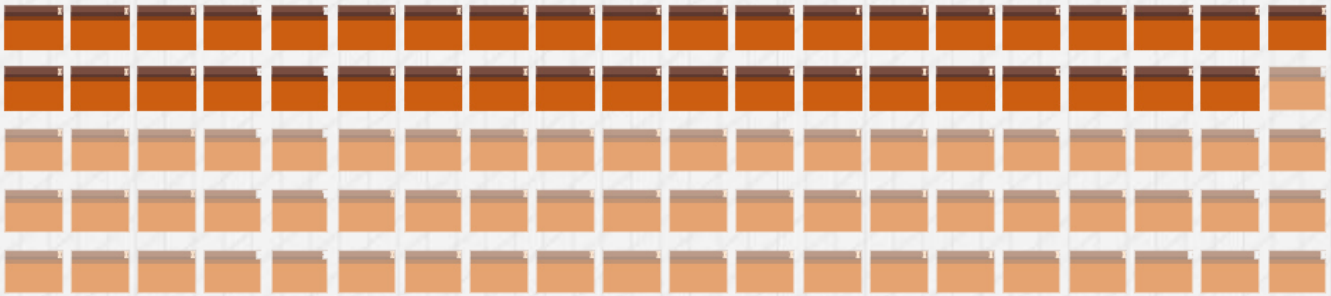
Biên soạn: ZendVN Group

Hướng dẫn: Lê Tấn Tài

Khóa Học: Lập trình Jquery

Sự bùng nổ web thời gian thực

SỐ NGƯỜI MỸ TRUY CẬP INTERNET
THÔNG QUA DI ĐỘNG



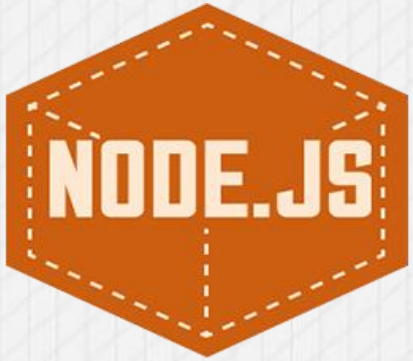
34/100 WEBSITE HÀNG ĐẦU THẾ GIỚI SỬ DỤNG HTML5

Sự phát triển của di động

Sự phát triển của HTML5

SỰ BÙNG NỔ WEB THỜI GIAN THỰC

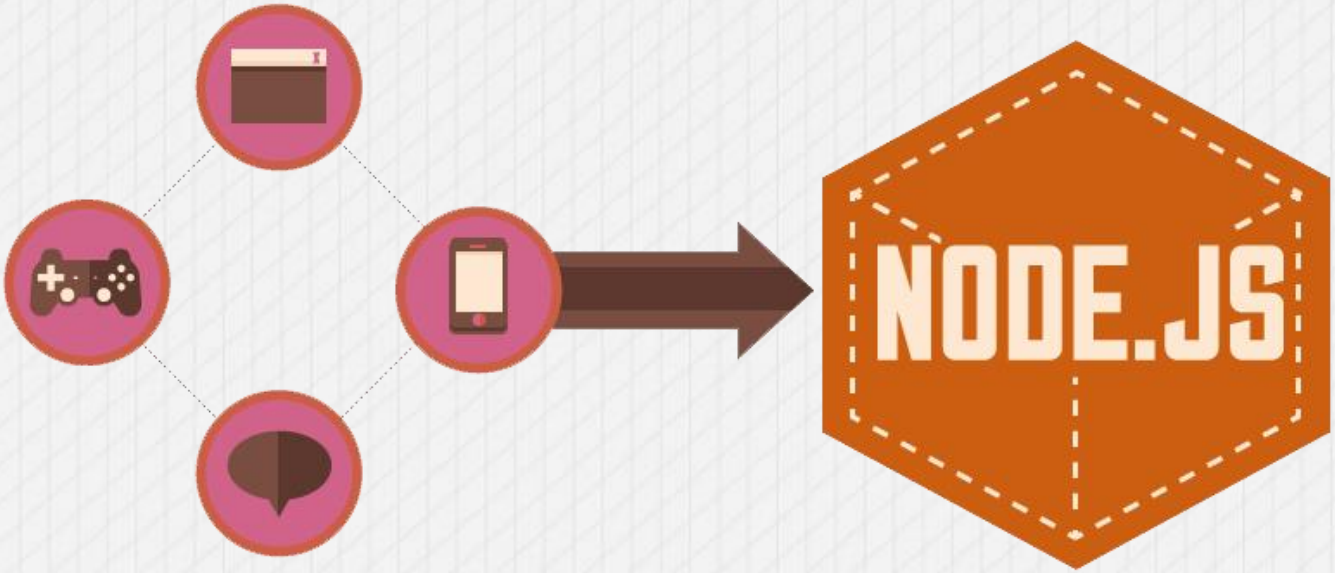
Hãy Chọn Node.js



1. Nó là JavaScript
2. Lập Trình ở bất kì cấp nào (Client - Server- database)
3. Cộng đồng lớn
4. Hiệu suất & khả năng mở rộng cao
5. Thân thiện với nhà phát triển

Tương lai Node.js

KHI CỘNG ĐỒNG PHÁT TRIỂN HỌ LUÔN ĐẶT RA YÊU CẦU MỚI



Với sự phát triển nhanh chóng về số lượng ứng dụng web,di động, mạng xã hội và game cho thấy 1 tương lai tươi sáng với node.js

Nội dung

I. Node.js
II. Node Package Modules (NPM)
III. Socket.io Package
IV. Mysql Package
V. Xây dựng ứng dụng Chat.

Giới thiệu Node.js

- Là một nền tảng phần mềm cho khả năng mở rộng máy chủ và các ứng dụng mạng.
- Được viết bằng JavaScript.
- Sử dụng kỹ thuật điều hướng theo sự kiện.
- Mô hình Non-Blocking I/O.
- Nhiều thư viện hỗ trợ thông qua **Node Packaged Modules (NPM)**.

Node.js dùng để làm gì ?

- Xây dựng websocket Server. chat server
- Ứng dụng upload file rất nhanh trên client.
- Xây dựng Ad Server.
- Hoặc bất kỳ ứng dụng dữ liệu thời gian thực nào.

Cài đặt & Tài liệu

- Cài đặt

http://nodejs.org ----> INSTALL

- Tài liệu

http://nodejs.org/api

Run "Hello World."

hello.js

```
var http = require('http');    require modules

http.createServer(function(request,reponse){
    response.writeHead(200);  status code in header
    response.write("Hello World.");   response body
    response.end();      close the connection
}).listen(8080);      listen for connection on this port

console.log('Listening on port 8080...');
```

```
node hello.js      Run the server
```

----> listening on port 8080

```
Go url    http://localhost:8080
```

----> Hello World.

Giới thiệu NPM

- Dùng để quản lý các gói của Node.js
- Đi kèm và được cài đặt tự động với môi trường máy tính khi cài Node.js
- Cho phép cài đặt các gói Node.js có sẵn trên Registry Npm hoặc gỡ bỏ các gói cài đặt.
- Npm được viết hoàn toàn bằng Javascript , và chạy trên nền tảng Node.js

Sử dụng https://www.npmjs.org

- **Cài đặt**

```
npm install packpage_name
```

----> *npm install socket.io*

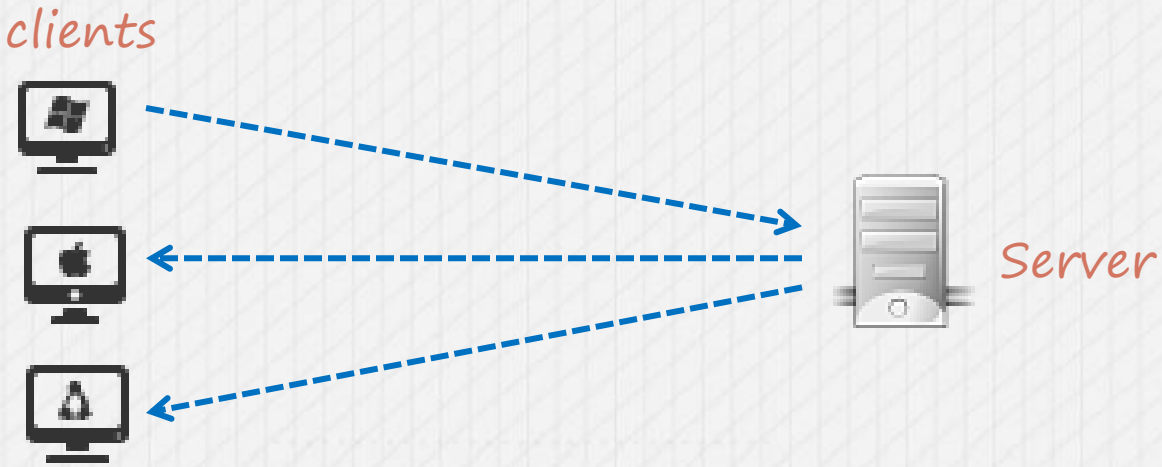
- **Gỡ bỏ**

```
npm uninstall package_name
```

----> *npm uninstall socket.io*

Giới thiệu Socket.io http://socket.io

- Là một mô hình websocket được đóng gói để chạy với Node.js
- Giữ cho Client và Server ở trạng thái luôn kết nối.
- Khả năng phản hồi nhanh với các yêu cầu.
- Hỗ trợ các ứng dụng chạy thời gian thực.



Cài đặt & Gỡ bỏ

- **Cài đặt**

```
npm install socket.io
```

- **Gỡ bỏ**

```
npm uninstall socket.io
```

Kết nối Server & Client

Kết nối Server & Client

server.js

```
var io = require('socket.io').listen(8080);
    io.set('transports',['xhr-polling']);

io.sockets.on('connection',function(socket){
    console.log('Client connected ...');
});
```

index.html

```
<script src="http://localhost:8080/socket.io/socket.io.js" ></script>
<script>
   var socket = io.connect('http://localhost:8080');
</script>
```

- Run: `node server.js`
- Cancel: `Ctrl+C`

Client gửi dữ liệu đến Server

- Gửi:

```
socket.emit('name',data)
```

- Nhận:

```
socket.on('name',function(data){

});                                    // server
```

Client gửi dữ liệu đến Server

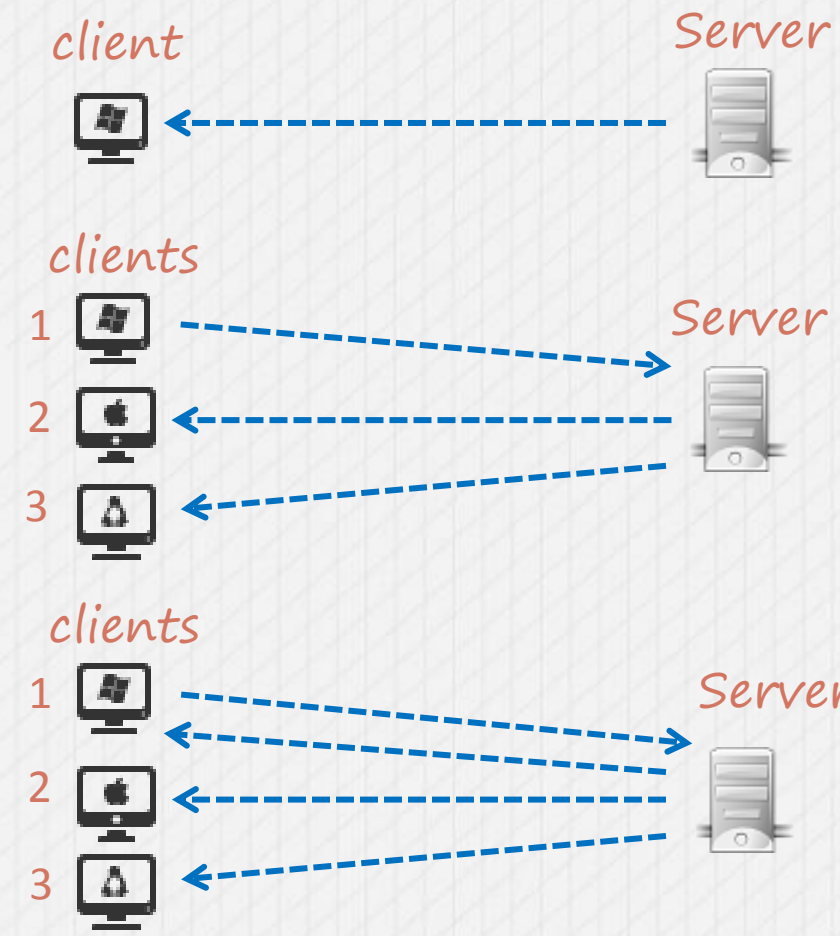
server.js

```
io.sockets.on('connection',function(socket){
    console.log('Client connected ...');

    socket.on('messages',function(data){
        console.log(data);
    });                                    listen for 'messages' events
});
```

index.html

```
<script src="http://localhost:8080/socket.io/socket.io.js" ></script>
<script>
    var socket = io.connect('http://localhost:8080');

        emit the 'messages' event on the server
        socket.emit('messages',{hello:'world'});
</script>
```

Server gửi dữ liệu đến clients



1. Server gửi đến client đang tạo ra kết nối.

```
socket. emit("name",data);
```

2. Server gửi đến tất cả các clients ngoại trừ client tạo ra kết nối để bắt đầu nó.

```
socket.broadcast.emit("name",data);
```

3. Server gửi đến tất cả các clients.

```
io.sockets. emit("name",data)
```

Server gửi dữ liệu đến Client

1. Server gửi đến client đang tạo ra kết nối.

server.js

```
io.sockets.on('connection',function(socket){
    console.log('Client connected ...');

    emit the 'messages' event on the client
    socket.emit('messages',{hello:'world'});
});
```

index.html

```
<script src="http://localhost:8080/socket.io/socket.io.js" ></script>
<script>
   var socket = io.connect('http://localhost:8080');
   socket.on('messages',function(data){
        console.log(data);
   });                                  listen for 'messages' events
</script>
```

Server phát dữ liệu đến clients

2. Server gửi đến tất cả các clients ngoại trừ client tạo ra kết nối để bắt đầu nó.

server.js

```
io.sockets.on('connection',function(socket){
    socket.broadcast.emit ('messages' ,{hello:'world'});
    bradcase message to all other clients connected
});
```

index.html

```
<script src="http://localhost:8080/socket.io/socket.io.js" ></script>
<script>
   var socket = io.connect('http://localhost:8080');
        socket.on('messages',function(data){
            console.log(data);
        });
</script>
```

Server phát dữ liệu đến clients

3. Server gửi đến tất cả các clients.

server.js

```
io.sockets.on('connection',function(socket){
    io.sockets.emit ('messages' ,{hello:'world'});
    bradcase message to all other clients connected
});
```

index.html

```
<script src="http://localhost:8080/socket.io/socket.io.js" ></script>
<script>
   var socket = io.connect('http://localhost:8080');
        socket.on('messages',function(data){
            console.log(data);
        });
</script>
```

Giới thiệu Mysql package

- Dùng để kết nối và xử lý mysql.
- Được viết bằng JavaScript
- Không yêu cầu biên dịch

Cài đặt https://www.npmjs.org/package/mysql

- **Cài đặt**

```
npm install mysql
```

- **Gỡ bỏ**

```
npm uninstall mysql
```

Kết nối với Mysql

server.js

```
var connection = require('mysql').createConnection({
      host          : 'localhost',
      user          : 'root',
      password      : '123',
      database      : 'database_name'
});
connection.connect();
```

Truy vấn database có 3 cách

Cách 1:

```
connection.query( sql, function(err,results){ });
```

- Sql : *Câu truy vấn sql*

```
connection.query('SELECT * FROM users ORDER BY id ASC', function(err, results) {
      if (err) throw err;
       else {
          console.log(results);
      }
});
```

Truy vấn database có 3 cách

Cách 2:

```
connection.query(sql, selectionArgs, function(err,results){ });
```

- Sql : *Câu truy vấn sql.*
- selectionArgs: *Mảng các tham số phụ cho câu điều kiện.*

```
connection.query('SELECT * FROM users WHERE id = ?', [1],function(err, results) {
      if (err) throw err;
       else {
          console.log(results);
      }
});
```

Truy vấn database có 3 cách

Cách 3:

```
connection.query(sql)
    .on('error', function(err) {
          handle error
     }) .on('result', function(row) {
          receive data
    }).on('end', function() {
        all rows have been received
    });
```

```
connection.query('SELECT * FROM users')
                .on('error', function(err) {
                    throw err;
                 })
                .on('result', function(row) {
                        console.log(row);
                 })
                 .on('end', function() {
                        console.log('end');
                 });
```

INSERT

```
connection.query(sql,data, function(err,results){ });
```

- Sql : *Câu truy vấn sql*
- data: *dữ liệu cần thêm*

```
var data = {name:'zendvn'};
connection.query('INSERT INTO users SET ?',data, function(err, result) {
        if (err) throw err;
        else {
            console.log(result);
        }
});
```

UPDATE

connection.query(sql, selectionArgs, function(err,results){ });

- Sql : *Câu truy vấn sql.*
- selectionArgs: *Mảng các tham số phụ cho câu điều kiện.*

```
var data = ['nodejs',2];
connection.query('UPDATE  users SET name = ? WHERE id = ?',data,
            function(err, result) {
                if (err) throw err;
            else {
                console.log(result);
            }
});
```

DELETE

connection.query(sql, selectionArgs, function(err,results){ });

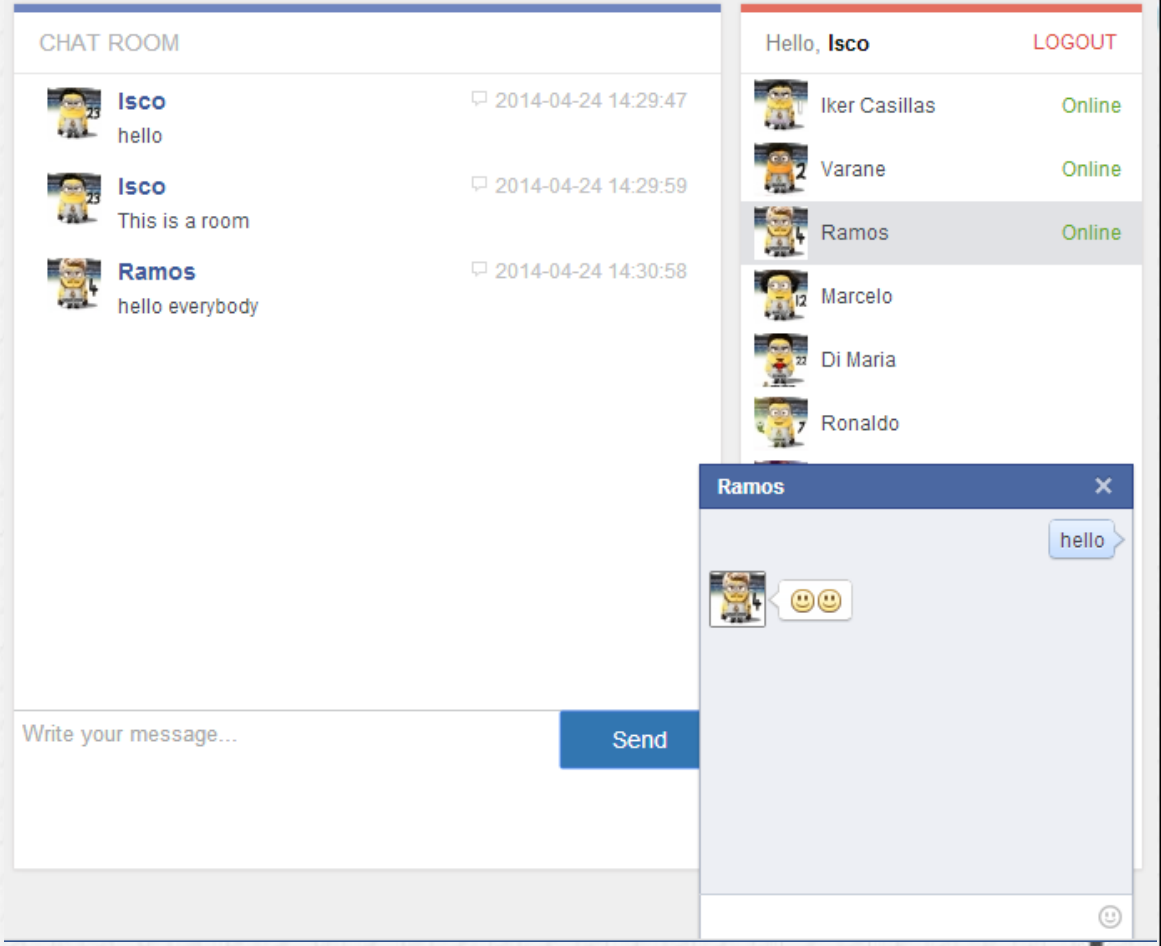
- Sql : *Câu truy vấn sql.*
- selectionArgs: *Mảng các tham số phụ cho câu điều kiện.*

```
connection.query('DELETE FROM users WHERE id = ?',[2], function(err, result) {
            if (err) throw err;
             else {
                 console.log(result);
             }
});
```

Xây dựng ứng dụng chat

Chức năng ?

- Nhiều người chat với nhau
- 2 người chat với nhau



Kiến thức

Cần gì ?

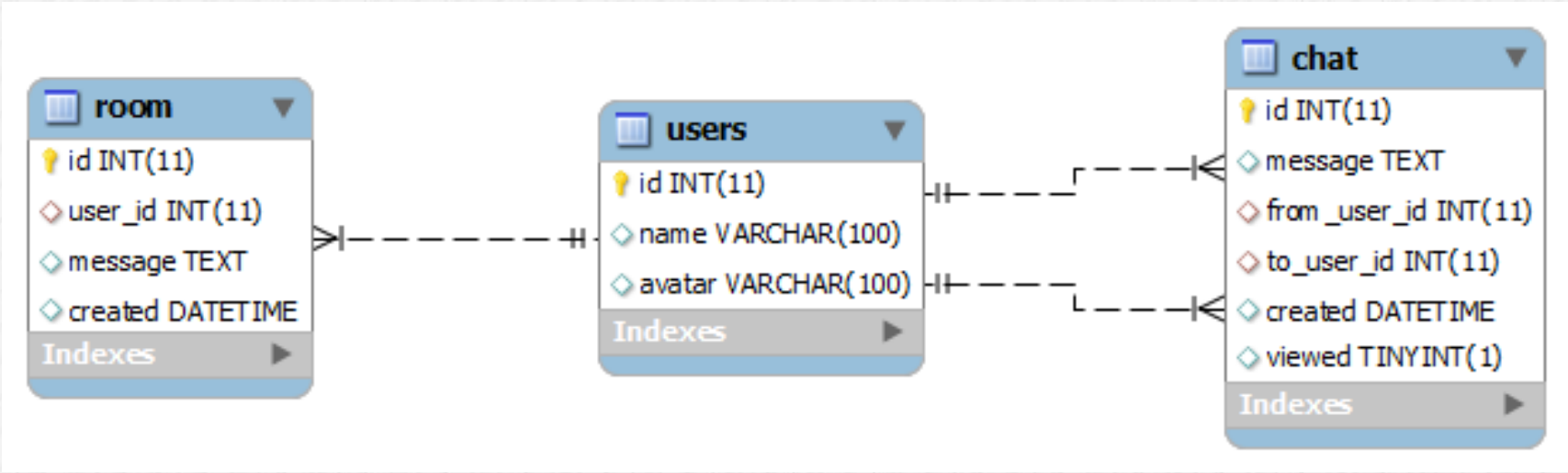
- Html
- Css
- Jquery
- Mysql
- Php
- Node.js

Cài đặt

Install ?

- xampp.
- node.js
  - socket.io package
  - mysql package
  - date-format package

Cơ sở dữ liệu


room
id INT(11)
user_id INT(11)
message TEXT
created DATETIME
Indexes
users
id INT(11)
name VARCHAR(100)
avatar VARCHAR(100)
Indexes
chat
id INT(11)
message TEXT
from _user_id INT(11)
to_user_id INT(11)
created DATETIME
viewed TINYINT(1)
Indexes

Xây dựng ứng dụng

3 nhóm chức năng

- Đăng nhập – đăng xuất
- Chat giữa nhiều người (chat room)
- Chat giữa 2 người

Chat giữa 2 người