

**ĐẠI HỌC ĐÀ NẴNG**  
**TRƯỜNG ĐẠI HỌC BÁCH KHOA**  
**KHOA CÔNG NGHỆ THÔNG TIN**

Tel. (+84.0236) 3736949, Fax. (84-511) 3842771  
Website: <http://dut.udn.vn/khoacntt>, E-mail: [cntt@dut.udn.vn](mailto:cntt@dut.udn.vn)

**BÁO CÁO**  
**ĐỒ ÁN CƠ SỞ NGÀNH MẠNG**

**ĐỀ TÀI :**

*Nguyên lý hệ điều hành: Xây dựng chương trình chia sẻ file giữa các máy tính người dùng.*

*Lập trình mạng: Xây dựng chương trình ứng dụng BKZalo cho phép người dùng có thể trò chuyện, nhắn tin, gửi hình ảnh.*

**SINH VIÊN : Nguyễn Thanh Sang**  
**MÃ SINH VIÊN : 102190037**  
**LỚP : 19TCLC\_DT1**

**SINH VIÊN : Võ Tấn Hậu**  
**MÃ SINH VIÊN : 102190012**  
**LỚP : 19TCLC\_DT1**

**SINH VIÊN : Lê Ngọc Duy**  
**MÃ SINH VIÊN : 102190010**  
**LỚP : 19TCLC\_DT1**

**CBHD : PGS.TS Nguyễn Tấn Khôi**

**Đà Nẵng, 12/2021**

## MỤC LỤC

<b>CHƯƠNG 1: CƠ SỞ LÝ THUYẾT .....</b>	<b>3</b>
1.1. CƠ SỞ LÝ THUYẾT .....	3
1.2. PHÁT BIỂU BÀI TOÁN .....	7
1.3. KẾT CHƯƠNG .....	7
<b>CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG.....</b>	<b>8</b>
2.1. PHÂN TÍCH YÊU CẦU .....	8
2.2. PHÂN TÍCH CHỨC NĂNG.....	8
2.2.1. Các chức năng của chương trình .....	8
2.2.2. Công nghệ sử dụng .....	8
2.3. THIẾT KẾ CƠ SỞ DỮ LIỆU .....	8
2.4. TỔ CHỨC CHƯƠNG TRÌNH.....	9
2.4.1. Tổ chức thư mục.....	9
2.4.2. Server.....	9
2.4.3. Client .....	11
2.5. KẾT CHƯƠNG .....	13
<b>CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ .....</b>	<b>14</b>
3.1. MÔ HÌNH TRIỂN KHAI.....	14
3.1.1. Mô hình triển khai .....	14
3.1.2. Các công cụ sử dụng.....	14
3.1.3. Cấu hình hệ thống.....	14
3.2. KẾT QUẢ THỰC NGHIỆM .....	14
3.2.1. Đăng ký, đăng nhập vào hệ thống .....	14
3.2.2. Giao diện chính chương trình.....	15
3.2.3. Giao diện quản lý file cá nhân.....	16
3.3. NHẬN XÉT ĐÁNH GIÁ KẾT QUẢ .....	16
3.4. KẾT CHƯƠNG .....	16
<b>CHƯƠNG 1: CƠ SỞ LÝ THUYẾT .....</b>	<b>17</b>
1.1. CƠ SỞ LÝ THUYẾT .....	17
1.2. PHÁT BIỂU BÀI TOÁN .....	19
1.3. KẾT CHƯƠNG .....	20
<b>CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG.....</b>	<b>21</b>
2.1. PHÂN TÍCH YÊU CẦU .....	21
2.2. PHÂN TÍCH CHỨC NĂNG.....	21

2.2.1.	Đối tượng sử dụng .....	21
2.2.2.	Các chức năng của chương trình .....	21
2.2.3.	Công nghệ sử dụng .....	22
2.3.	THIẾT KẾ CƠ SỞ DỮ LIỆU .....	22
2.4.	TỔ CHỨC CHƯƠNG TRÌNH.....	23
2.4.1.	Tổ chức thư mục.....	23
2.4.2.	Server.....	23
2.4.3.	Client .....	26
2.5.	KẾT CHƯƠNG .....	29
<b>CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ .....</b>		<b>30</b>
3.1.	MÔ HÌNH TRIỂN KHAI.....	30
3.1.1.	Mô hình triển khai .....	30
3.1.2.	Các công cụ sử dụng.....	30
3.1.3.	Cấu hình hệ thống.....	30
3.2.	KẾT QUẢ THỰC NGHIỆM .....	30
3.2.1.	Đăng ký, đăng nhập vào hệ thống .....	30
3.2.2.	Chức năng xem danh sách người dùng, cập nhật trạng thái người dùng	32
3.2.3.	Chức năng nhắn tin, gửi emoji, gửi ảnh, gửi file.....	33
3.2.4.	Giao diện server.....	36
3.3.	NHẬN XÉT ĐÁNH GIÁ KẾT QUẢ .....	37
3.4.	KẾT CHƯƠNG .....	37
<b>KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....</b>		<b>38</b>
1.	<b>KẾT QUẢ ĐẠT ĐƯỢC .....</b>	<b>38</b>
2.	<b>KIẾN NGHỊ VÀ HƯỚNG PHÁT TRIỂN .....</b>	<b>38</b>

# DANH SÁCH HÌNH ẢNH

Hình 1 Mô hình mạng máy tính.....	3
Hình 2 Mô tả kiến trúc mạng client – server .....	5
Hình 3 Sự kiện đăng nhập và đăng ký .....	10
Hình 4 Giao diện đăng nhập, đăng ký .....	15
Hình 5 Giao diện chính chương trình .....	15
Hình 6 Giao diện quản lý file cá nhân .....	16
Hình 7 Mô tả Web API.....	18
Hình 8 Table Account.....	22
Hình 9 Table User.....	22
Hình 10 Table Message .....	23
Hình 11 Đăng ký tài khoản .....	24
Hình 12 API Register.....	24
Hình 13 API Login .....	25
Hình 14 Call API Register .....	27
Hình 15 Call API Login.....	27
Hình 16 Giao diện đăng ký .....	31
Hình 17 Giao diện đăng nhập .....	31
Hình 18 Danh sách người dùng .....	32
Hình 19 Cập nhật trạng thái hoạt động .....	33
Hình 20 Giao diện nhắn tin.....	33
Hình 21 Panel more .....	34
Hình 22 Danh sách emoji loại 1.....	35
Hình 23 Danh sách emoji loại 2.....	35
Hình 24 Gửi file, gửi ảnh.....	36
Hình 25 Giao diện server .....	37

## DANH SÁCH TỪ VIẾT TẮT

Từ viết tắt	Diễn giải
IP	Internet Protocol
MD5	Message-Degist Algorithm 5
SHA	Secure Hash Algorithm
API	Application Programming Interface
URL	Uniform Resource Locator

# MỞ ĐẦU

## 1. Tổng quan về đề tài

**Nguyên lý hệ điều hành:** Xây dựng được một hệ thống theo mô hình client-server cho phép chia sẻ file giữa các máy tính người dùng.

**Lập trình mạng:** Xây dựng được một hệ thống theo mô hình client-server cho phép các người dùng có thể trao đổi dữ liệu với nhau ở dạng văn bản, hình ảnh. Việc trao đổi dữ liệu được thực hiện thông qua ứng dụng trên máy tính.

## 2. Mục đích và ý nghĩa của đề tài

Hiểu rõ được nguyên lý và cách hoạt động của mô hình client-server cũng như có thể xây dựng được các chương trình có khả năng làm việc trong môi trường mạng tích hợp nói chung và mạng máy tính nói riêng.

Trao đổi thêm được kiến thức hệ thống mạng, mô hình xây dựng chương trình ứng dụng mạng cũng như kiến thức về cơ sở dữ liệu.

Ngoài ra đề tài còn tạo điều kiện cho sinh viên:

- Đi sâu và nắm vững một cách có hệ thống kiến thức đã thu nhận được trong quá trình học lý thuyết, làm bài tập và thực hành.
- Từng bước làm quen với các công tác khoa học có định hướng của giáo viên hướng dẫn và hình thành hành vi nghiên cứu độc lập có sự trợ giúp của tài liệu tham khảo.
- Gắn quá trình với công tác nghiên cứu thực tế.
- Trình bày rõ ràng và khoa học một vấn đề thuộc lĩnh vực nghiên cứu của mình

### **3. Bố cục của đề án**

Đề án gồm 2 phần, mỗi phần bao gồm các nội dung sau:

*Mở đầu*

*Chương 1: CƠ SỞ LÝ THUYẾT*

*Chương 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG*

*Chương 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ*

*Kết luận và hướng phát triển.*

# ***Phần I: NGUYÊN LÝ HỆ ĐIỀU HÀNH***

***Đề tài:***

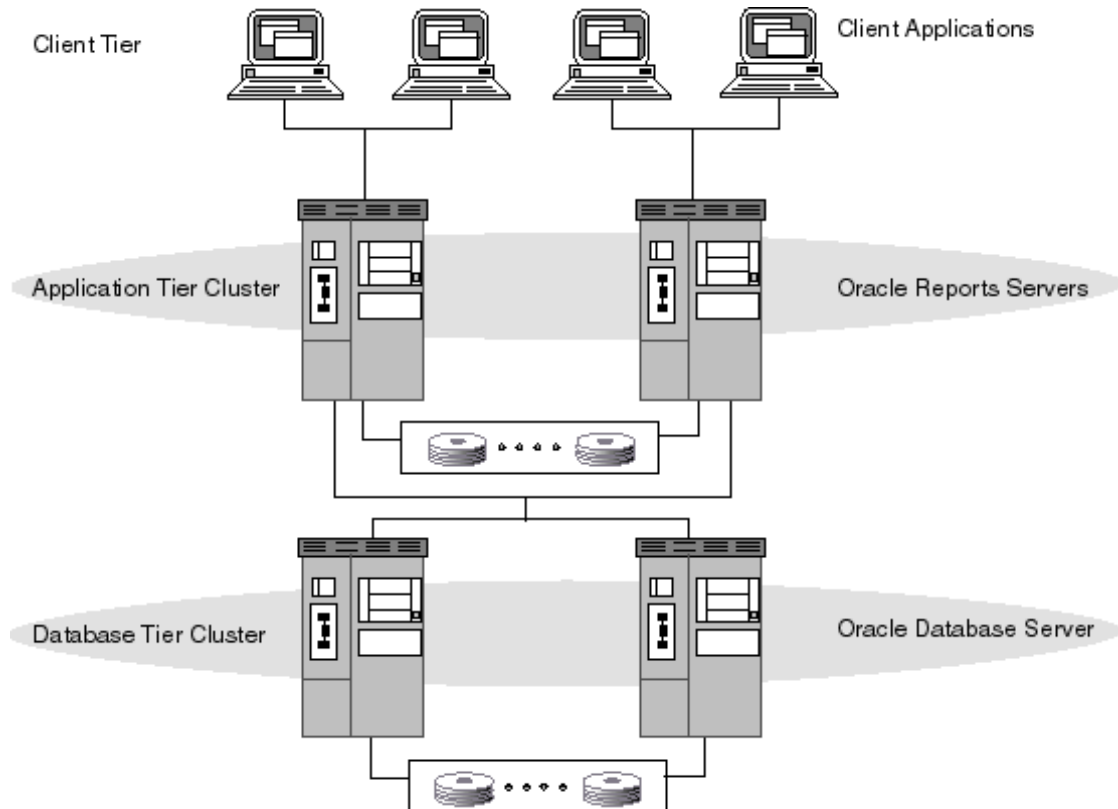
***Xây dựng chương trình chia sẻ file giữa các máy tính***

## **CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**

### **1.1. CƠ SỞ LÝ THUYẾT**

#### **a. Mạng máy tính:**

Hiểu một cách đơn giản, mạng máy tính là một hệ thống bao gồm nhiều máy tính và các thiết bị mạng được kết nối với nhau bằng đường truyền vật lý theo một kiến trúc mạng nào đó. Nhằm tăng khả năng thu thập thông tin, dữ liệu và chia sẻ tài nguyên đến cho nhiều người dùng trong cùng một thời gian



*Hình 1 Mô hình mạng máy tính*



Trên một hệ thống mạng, máy tính có thể đảm nhận một trong 3 vai trò sau:

Máy tính đóng vai trò là máy chủ – Server: Là máy tính có khả năng cung cấp tài nguyên và các dịch vụ đến các máy trạm khác trong hệ thống mạng. Server đóng vai trò hỗ trợ cho các hoạt động trên máy trạm client diễn ra hiệu quả hơn.

Máy tính đóng vai trò là máy trạm – Client: Với vai trò là máy trạm, chúng sẽ không cung cấp tài nguyên đến các máy tính khác mà chỉ sử dụng tài nguyên được cung cấp từ máy chủ. Một client trong mô hình này có thể là một server cho mô hình khác, tùy thuộc vào nhu cầu sử dụng của người dùng.

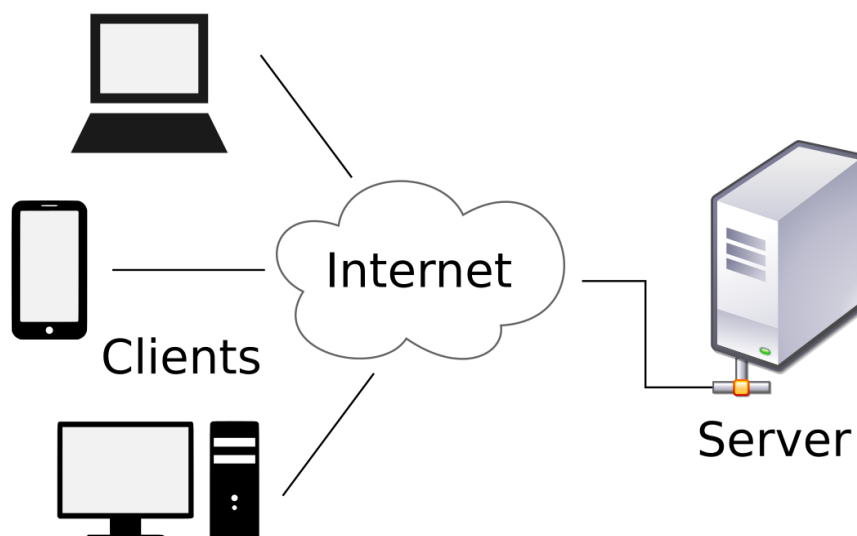
Máy tính đóng vai trò là Peer: Vừa sử dụng tài nguyên từ máy chủ cung cấp, đồng thời cũng cung cấp tài nguyên đến các máy tính khác trong mạng.

Dựa vào các vai trò máy tính cung cấp cho hệ thống mạng mà chúng được phân chia ra thành 3 loại sau:

- Mô hình máy khách chủ – Client Server
- Mô hình mạng ngang hàng – Peer-to-Peer
- Mô hình mạng lai – Hybrid

#### **b. Mô hình kiến trúc mạng client – server**

Mô hình mạng client server là mô hình nổi tiếng trong hệ thống mạng máy tính, và chúng được áp dụng rộng rãi trên hầu hết các trang web hiện có. Với mô hình này, các máy tính con được đóng vai trò như một máy khách, chúng làm nhiệm vụ gửi yêu cầu đến các máy chủ. Để máy chủ xử lý yêu cầu và trả kết quả về cho máy khách đó.



*Hình 2 Mô tả kiến trúc mạng client – server*

Trong mô hình này, server chấp nhận tất cả các yêu cầu hợp lệ từ mọi nơi khác nhau trên mạng, sau đó trả kết quả về máy tính đã gửi yêu cầu.

Máy tính được coi là máy khách khi chúng làm nhiệm vụ gửi yêu cầu đến các máy chủ và đợi câu trả lời được gửi về.

Để máy khách và máy chủ có thể giao tiếp được với nhau thì giữa chúng phải có một chuẩn nhất định, và chuẩn đó được gọi là giao thức. Một số giao thức chuẩn được sử dụng rộng rãi hiện nay như TCP/IP, OSI, ISDN, X.25, Lan-to-Lan,.. Khi đó, nếu máy khách muốn lấy được thông tin từ máy chủ, chúng phải tuân theo một giao thức mà máy chủ đó đưa ra. Nếu yêu cầu đó được chấp nhận thì máy chủ sẽ thu thập thông tin và trả về kết quả cho máy khách yêu cầu. Bởi thông thường, server luôn trong trạng thái sẵn sàng nhận yêu cầu từ các client, nên chỉ cần client gửi tín hiệu và chấp nhận yêu cầu là server sẽ trả về kết quả trong thời gian ngắn nhất có thể.

- **Ưu điểm:**

Nhờ sự xuất hiện của mô hình client server giúp chúng ta có thể làm việc trên bất kì một máy tính nào có hỗ trợ giao thức truyền thông. Giao thức chuẩn này cũng giúp

các nhà sản xuất tích hợp lên nhiều sản phẩm khác nhau mà không gặp phải khó khăn gì.

Có thể có nhiều chương server cùng làm một dịch vụ, chúng có thể nằm trên nhiều máy tính hoặc một máy tính.

Mô hình Client server chỉ mang đặc điểm của phần mềm mà không hề liên quan đến phần cứng, ngoài yêu cầu duy nhất là server phải có cấu hình cao hơn các client.

Client server hỗ trợ người dùng nhiều dịch vụ đa dạng và sự tiện dụng bởi khả năng truy cập từ xa mà các mô hình cũ không có được.

Mô hình mạng khách chủ cung cấp một nền tảng lý tưởng, cho phép cung cấp tích hợp các kỹ thuật hiện đại như mô hình thiết kế hướng đối tượng, hệ chuyên gia, hệ thông tin địa lý (GIS).

- **Nhược điểm:**

Do phải trao đổi dữ liệu giữa 2 máy tính khác nhau ở 2 khu vực địa lý cách xa nhau, nên vấn đề bảo mật dữ liệu thông tin đôi khi còn chưa được an toàn lắm. Đây là nhược điểm duy nhất của mô hình này.

### **c. Socket IO**

Để xây dựng một ứng dụng realtime cần sử dụng socketio. Socketio sẽ giúp các bên ở những địa điểm khác nhau kết nối với nhau, truyền dữ liệu ngay lập tức thông qua server trung gian. Socketio có thể được sử dụng trong nhiều ứng dụng như chat, game online, cập nhật kết quả của một trận đấu đang xảy ra,...

Socketio không phải là một ngôn ngữ, mà chỉ là 1 công cụ giúp thực hiện những ứng dụng realtime. Vì thế, không thể sử dụng socketio để thay thế hoàn toàn cho một ngôn ngữ, mà phải sử dụng kết hợp với một ngôn ngữ khác. Ngôn ngữ đó có thể là php, asp.net, nodejs,...

Cơ chế hoạt động của một ứng dụng realtime đó là thông qua server để lắng nghe (listen) data và truyền data về các máy client. Vì vậy cần cài khai báo sử dụng socketio ở cả phía server và client.

## 1.2. PHÁT BIỂU BÀI TOÁN

- Xây dựng chương trình gồm hai phần là client và server.
  - Server có chức năng tiếp nhận, lưu trữ giữ liệu và phản hồi lại với phía client khi client gửi yêu cầu đến.
  - Người dùng có thể sử dụng chương trình bên phía client để upload file lên server hoặc download file từ server xuống máy tính.
- Sử dụng Socket IO

Để cài đặt Socket.io trong dự án của mình bạn cần phải cài đặt ở 2 phía đó là server và client. Socket.io sẽ đảm nhận kết nối giữa 2 phía, thông thường các API của 2 phía sẽ tương tự giống nhau.

Cơ chế lắng nghe, truyền dữ liệu của socketio: Để bạn có thể lắng nghe data thì hãy sử dụng câu lệnh `socket.on()`, để phát dữ liệu thì sử dụng lệnh `socket.emit()`.

## 1.3. KẾT CHƯƠNG

Chương này trình bày những kiến thức tổng quát về mạng máy tính cũng như về mô hình client-server, kèm theo các ưu và nhược điểm của nó. Bên cạnh đó chương cũng giới thiệu sơ lược về Socket IO và cách sử dụng Socket IO. Để có thể xây dựng được một ứng dụng chia sẻ file giữa các máy tính thì việc sử dụng Socket IO sẽ giúp đỡ chúng ta rất nhiều về vấn đề giao tiếp và trao đổi dữ liệu giữa client và server.

## CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

### 2.1. PHÂN TÍCH YÊU CẦU

- **Nội dung đề tài:**

Xây dựng chương trình chia sẻ file giữa các máy tính

- **Hướng giải quyết**

Tìm hiểu về Socket IO, cách ghi file trên máy tính và cách sử dụng giao thức TCP/IP để chia sẻ file giữa các máy tính. Sau đó cài đặt server dựa trên những kiến thức đã tìm hiểu.

Viết chương trình client có kết nối với server đã thiết lập ở trên để giao tiếp với nhau

### 2.2. PHÂN TÍCH CHỨC NĂNG

#### 2.2.1. Các chức năng của chương trình

- Chức năng đăng nhập, đăng ký.
- Chức năng upload, download file.
- Chức năng sắp xếp, tìm kiếm file từ danh sách tất cả các file có trên server.
- Chức năng quản lý các file đã upload.

#### 2.2.2. Công nghệ sử dụng

- Socket IO.

### 2.3. THIẾT KẾ CƠ SỞ DỮ LIỆU

Chương trình sử dụng MySQL để lưu trữ dữ liệu về tài khoản người dùng và các thông tin cơ bản của file như tên file, kích thước file, đường dẫn của file trên máy server,...

## **2.4. TỔ CHỨC CHƯƠNG TRÌNH**

### **2.4.1. Tổ chức thư mục**

Chương trình được chia làm hai chương trình con là server và client với mỗi phần thực hiện một chức năng khác nhau.

### **2.4.2. Server**

Server có nhiệm vụ lắng nghe kết nối, nhận giữ liệu, lưu trữ giữ liệu và phản hồi lại client khi có yêu cầu gửi đến. Server gồm 3 module chính: Socket IO, Đọc và ghi file, truy vấn database.

#### **a. Socket IO**

Module này có chức năng chính trong chương trình, phụ trách nhiều tác vụ như: xác thực đăng nhập, đăng ký người dùng, cập nhật danh sách file, nhận dữ liệu khi người dùng upload file hay gửi dữ liệu đến client khi người dùng download file.

- Đăng nhập, đăng ký người dùng:
  - Client gửi thông tin tài khoản gồm username và password đến server.
  - Server nhận được thông tin tài khoản người dùng sau đó gửi dữ liệu đến module truy vấn database để kiểm tra và xác nhận tài khoản.
  - Nếu tài khoản hợp lệ, server sẽ gửi trả lại kết quả “true” đến client, nếu không hợp lệ, sẽ gửi trả lại kết quả “false” đến client.

```

server.addListener("register", User.class, new DataListener<User>() {
    @Override
    public void onData(SocketIOClient sioc, User data, AckRequest ar) throws Exception {
        User u = BO.Instance().Register(data);
        if(u != null) {
            ar.sendAckData(true);
            sioc.sendEvent("all_file", BO.Instance().GetAllFile().toArray());
        }
        else {
            ar.sendAckData(false);
        }
    }
});
server.addListener("login", User.class, new DataListener<User>() {
    @Override
    public void onData(SocketIOClient sioc, User data, AckRequest ar) throws Exception {
        User u = BO.Instance().Login(data);
        if(u != null) {
            ar.sendAckData(true);
            sioc.sendEvent("all_file", BO.Instance().GetAllFile().toArray());
        }
        else {
            ar.sendAckData(false);
        }
    }
});

```

*Hình 3 Sự kiện đăng nhập và đăng ký*

- Cập nhật danh sách file
  - Client gửi yêu cầu danh sách file trên server.
  - Server nhận được yêu cầu và gửi yêu cầu đến module truy vấn database để lấy danh sách file.
  - Danh sách file được gửi trả lại client.
- Nhận dữ liệu upload file
  - Client gửi dữ liệu file upload đến server.
  - Server nhận thông tin của file, gửi thông tin đến module truy vấn database để lưu lại thông tin của file.
  - Server nhận dữ liệu của file sau đó truyền dữ liệu đến module đọc ghi file để thực hiện ghi file. File sẽ được ghi với tên là id của file đó.
- Download file
  - Client gửi yêu cầu download file đến server kèm theo thông tin file cần download.

- Server nhận được yêu cầu sẽ gửi thông tin file cần download đến module đọc ghi file, module đọc ghi file sẽ thực hiện đọc file trên máy server và gửi dữ liệu cho client dưới dạng các mảng byte.

## **b. Truy vấn database**

Module này làm việc với cơ sở dữ liệu và trả về dữ liệu khi các module khác yêu cầu.

Gồm các chức năng chính:

- Xác thực đăng nhập, đăng ký.
- Lấy danh sách file.
- Thêm thông tin file mới.
- Sửa thông tin file.
- Xóa file.

## **c. Đọc ghi file**

Module này sử dụng thư viện RandomAccessFile để thực hiện đọc, ghi file. Có chức năng ghi file lên máy server khi client upload file và thực hiện đọc file và gửi nội dung file đến client khi client yêu cầu download file.

Module này thực hiện trên một luồng riêng biệt để tối ưu hóa thời gian thực hiện trên server.

### **2.4.3. Client**

Client có chức năng lấy dữ liệu từ phía server và hiển thị dữ liệu lên giao diện người dùng. Gồm các module chính: Socket IO, View, Đọc ghi file.

#### **a. Socket IO**

Module này thực hiện trao đổi thông tin với server, thực hiện một số chức năng cụ thể như: Đăng nhập và đăng ký, upload và download file, cập nhật danh sách file.



- Đăng nhập và đăng ký
  - Gửi thông tin tài khoản đến server
  - Nhận kết quả trả về từ server, từ kết quả của server thực hiện xác nhận tài khoản có chính xác hay không và gửi thông tin đến module View
- Upload file
  - Nhận thông file cần upload từ module View
  - Module đọc ghi file sẽ thực hiện đọc file
  - Module Socket IO sẽ gửi nội dung file đến server dưới dạng các mảng byte.
- Download file
  - Nhận thông tin file cần download từ module View
  - Module Socket IO sẽ gửi thông tin file cần download đến server, sau đó nhận dữ liệu dưới dạng các mảng byte từ server gửi đến.
  - Dữ liệu sẽ được chuyển sang module đọc ghi file để thực hiện ghi file trên máy.
- Cập nhật danh sách file
  - Module Socket IO gửi yêu cầu danh sách file đến server, sau đó nhận dữ liệu là một mảng danh sách thông tin các file.
  - Sau đó dữ liệu sẽ được gửi đến module View để hiển thị lên giao diện người dùng.
  - Khi có bất kì sự thay đổi nào về danh sách file chẳng hạn như có người dùng upload file mới hoặc có người dùng xóa file, module Socket IO sẽ nhận được thông báo từ server và sẽ thực hiện cập nhật lại danh sách file.

## **b. View**

Có chức năng chính là tương tác với người dùng, lấy là hiển thị dữ liệu cho người dùng.

## **c. Đọc ghi file**

Module này sử dụng RandomAccessFile để thực hiện đọc, ghi file. Có chức năng đọc file trên máy người dùng để thực hiện upload file hoặc ghi file lên máy người dùng khi download file.

## **2.5. KẾT CHƯƠNG**

Chương này trình bày hướng giải quyết của bài toán, các chức năng chính của chương trình, các công nghệ sử dụng và tổ chức chương trình. Ngoài ra chương còn trình bày các thuật toán để giải quyết vấn đề nhận và gửi tin nhắn trong ứng dụng.

## **CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ**

### **3.1. MÔ HÌNH TRIỂN KHAI**

#### **3.1.1. Mô hình triển khai**

Chương trình được triển khai theo mô hình client-server.

#### **3.1.2. Các công cụ sử dụng**

- Ngôn ngữ lập trình: java
- IDE: Eclipse

#### **3.1.3. Cấu hình hệ thống**

- Phiên bản java: jdk 16

### **3.2. KẾT QUẢ THỰC NGHIỆM**

#### **3.2.1. Đăng ký, đăng nhập vào hệ thống**

- Người dùng điền thông tin tài khoản và mật khẩu vào đầy đủ các trường sau đó nhấn đăng nhập hoặc đăng ký.
- Nếu đăng nhập hoặc đăng ký thành công, người dùng có thể truy cập vào hệ thống.
- Nếu đăng nhập hoặc đăng ký thất bại, hệ thống sẽ thông báo lỗi cho người dùng

Hình 4 Giao diện đăng nhập, đăng ký

### 3.2.2. Giao diện chính chương trình

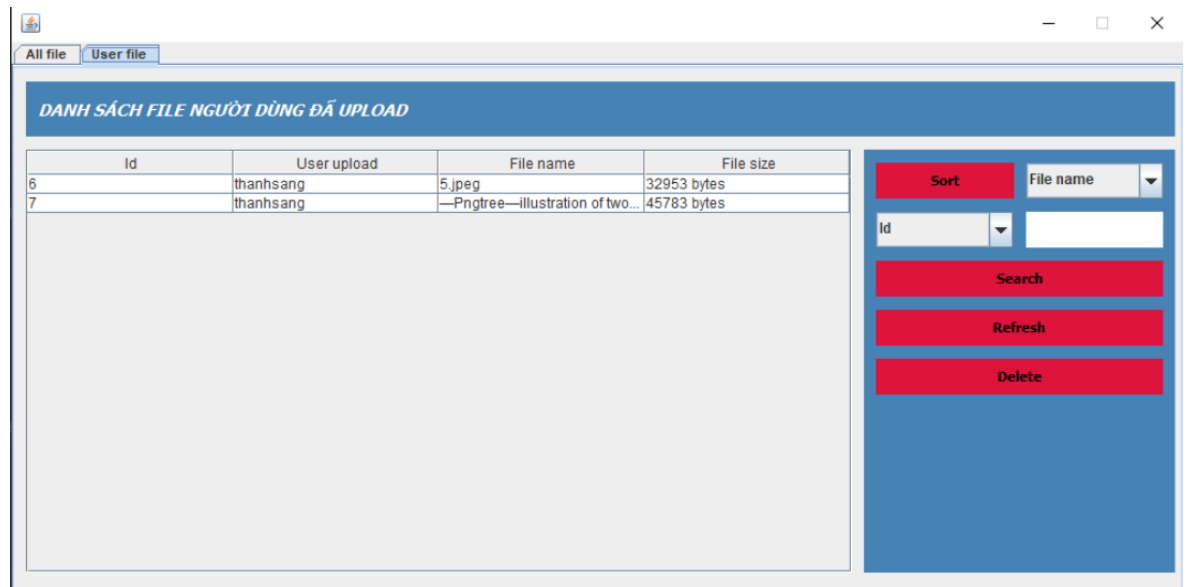
- Người dùng có thể sắp xếp, tìm kiếm thông tin file.
- Để upload file, người dùng nhấn vào button upload file sau đó chọn file cần upload, nhấn open sau đó hệ thống sẽ tiến hành upload file.
- Để download file, người dùng chọn file cần download trên danh sách sau đó nhấn vào button download file, sau đó hệ thống sẽ tiến hành download file.

Id	User upload	File name	File size
5	sang1	6.jpeg	64367 bytes
6	thahsang	5.jpeg	32953 bytes
7	thahsang	—Pngtree—illustration of two...	45783 bytes

Hình 5 Giao diện chính chương trình

### 3.2.3. Giao diện quản lý file cá nhân

- Tương tự như ở giao diện chính, người dùng cũng có thể sắp xếp hoặc tìm kiếm thông tin file mà người dùng đã upload.
- Người dùng có thể xóa file bằng cách chọn một file bên danh sách sau đó nhấn vào button delete, sau đó hệ thống sẽ tiến hành xóa file.



Hình 6 Giao diện quản lý file cá nhân

### 3.3. NHẬN XÉT ĐÁNH GIÁ KẾT QUẢ

- Chương trình đã thực hiện được các điều kiện mà bài toán đặt ra, tuy nhiên nếu muốn đưa vào sử dụng cần thêm nhiều chức năng hơn.
- Giao diện của chương trình cũng chưa tối ưu và thân thiện với người dùng, cần cải thiện, nâng cấp để chương trình có một giao diện đẹp hơn và thân thiện với người dùng hơn.

### 3.4. KẾT CHƯƠng

Chương này trình bày về môi trường đề cài đặt, kết quả triển khai và đánh giá nhận xét kết quả chương trình.

## ***Phần II: LẬP TRÌNH MẠNG***

### ***Đề tài:***

***Xây dựng chương trình BKZalo cho phép người dùng có thể trò chuyện, nhắn tin, gửi hình ảnh.***

## **CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**

### **1.1. CƠ SỞ LÝ THUYẾT**

#### **a. Mạng máy tính**

Tham khảo phần nguyên lý hệ điều hành.

#### **b. Mô hình kiến trúc mạng client-server**

Tham khảo phần nguyên lý hệ điều hành.

#### **c. Socket IO**

Tham khảo phần nguyên lý hệ điều hành.

#### **d. Tổng quan về API**

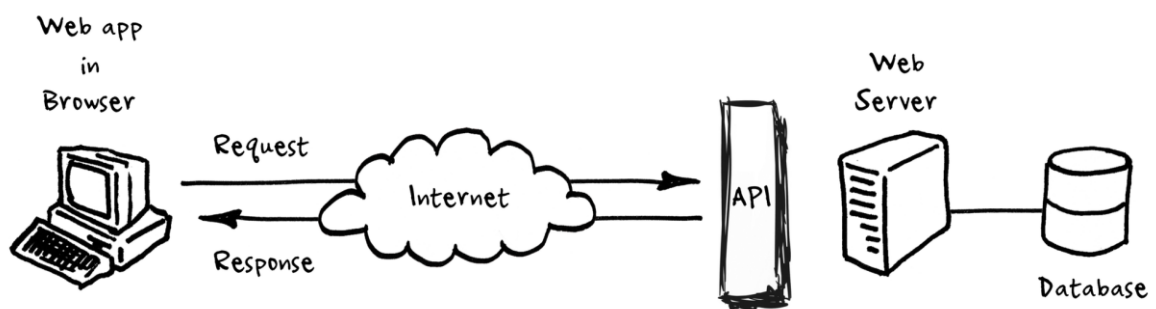
- **API là gì:**

API được viết tắt bởi cụm từ tiếng Anh Application Programming Interface, tức giao diện lập trình ứng dụng. Đây là phương tiện cho hai hoặc nhiều ứng dụng trao đổi, tương tác với nhau, tạo ra tương tác giữa người dùng với ứng dụng hiệu quả và tiện lợi hơn.

Với API, các lập trình viên có thể tiếp cận, truy xuất dữ liệu từ máy chủ thể hiện chúng trên ứng dụng phần mềm hoặc website của mình một cách dễ dàng hơn.

- **Web API là gì?**

Web API là một phương thức dùng để cho phép các ứng dụng khác nhau có thể giao tiếp, trao đổi dữ liệu qua lại. Dữ liệu được Web API trả lại thường ở dạng JSON hoặc XML thông qua giao thức HTTP hoặc HTTPS.



*Hình 7 Mô tả Web API*

Ưu và nhược điểm của Web API

- **Ưu điểm:**

- Web API được sử dụng hầu hết trên các ứng dụng desktop, ứng dụng mobile và ứng dụng website.
- Linh hoạt với các định dạng dữ liệu khi trả về client: Json, XML hay định dạng khác.
- Nhanh chóng xây dựng HTTP service: URI, request/response headers, caching, versioning, content formats và có thể host trong ứng dụng hoặc trên IIS.
- Mã nguồn mở, hỗ trợ chức năng RESTful đầy đủ, sử dụng bởi bất kì client nào hỗ trợ XML, Json.

- Hỗ trợ đầy đủ các thành phần MVC như: routing, controller, action result, filter, model binder, IoC container, dependency injection, unit test.
- Giao tiếp hai chiều được xác nhận trong các giao dịch, đảm bảo độ tin cậy cao.

▪ **Nhược điểm:**

Do web API còn khá mới nên chưa thể đánh giá nhiều về nhược điểm của mô hình này. Tuy nhiên, dễ dàng nhận thấy:

- Để sử dụng hiệu quả cần có kiến thức chuyên sâu, có kinh nghiệm backend tốt
- Tốn thời gian và chi phí cho việc phát triển, nâng cấp và vận hành
- Có thể gặp vấn đề về bảo mật khi hệ thống bị tấn công nếu không giới hạn điều kiện kỹ.

## 1.2. PHÁT BIỂU BÀI TOÁN

- Xây dựng Web API cho phép xác thực, đăng nhập, thao tác dữ liệu; ứng dụng truy xuất đến website thông qua các API cung cấp.
- Xây dựng ứng dụng kết nối với Webserver để quản lý tài khoản và lưu trữ dữ liệu. Người dùng có thể trao đổi thông tin qua ứng dụng này. Hệ thống có cơ chế xác thực client và lưu nhật ký.

### a. Sử dụng Web API

4 lệnh CRUD cơ bản:



CRUB bao gồm: Create, Read, Update, Delete. Đây là 4 chức năng cơ bản của bất kỳ cơ sở dữ liệu nào. Web API đều hỗ trợ đầy đủ 4 lệnh, giúp bạn có thể thao tác với dữ liệu lấy từ máy chủ dễ dàng hơn.

- Post: Có chức năng tạo dữ liệu, thông tin mới
- Get: Lệnh đọc/lấy dữ liệu, thông tin mới
- Put: Cập nhật thông tin và dữ liệu đã có
- Delete: Xóa thông tin và dữ liệu đã có

#### **b. Sử dụng Socket**

Tham khảo phần nguyên lý hệ điều hành

### **1.3. KẾT CHƯƠng**

Chương này trình bày những kiến thức tổng quát về mạng máy tính cũng như về mô hình client – server, kèm theo các ưu và nhược điểm của nó. Bên cạnh đó chương cũng giới thiệu sơ lược về Web API và Socket IO. Để có thể xây dựng được một ứng dụng chat real-time thông qua mô hình client - server thì việc sử dụng Web API và Socket IO sẽ giúp đỡ chúng ta rất nhiều về vấn đề real-time cũng như backend server.

## CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

### 2.1. PHÂN TÍCH YÊU CẦU

- **Nội dung đề tài:**

Xây dựng một ứng dụng theo mô hình client – server giúp trao đổi dữ liệu văn bản và hình ảnh với nhau.

- **Hướng giải quyết:**

Tìm hiểu về cách sử dụng Web API và Socket IO để trao đổi dữ liệu giữa client và server. Cài đặt server dựa trên công nghệ Web API và Socket IO đã tìm hiểu để lắng nghe và lấy thông tin từ người dùng gửi đến mà các thông tin đấy sẽ được lưu trữ trong cơ sở dữ liệu sau đó server sẽ lấy dữ liệu đấy hiển thị lên máy của người dùng khác.

Viết ứng dụng chat cho nhiều client mà thông qua server có kết nối mạng mà ta đã thiết lập ở trên để giao tiếp với nhau.

### 2.2. PHÂN TÍCH CHỨC NĂNG

#### 2.2.1. Đối tượng sử dụng

- Những người có nhu cầu trò chuyện, giao tiếp với nhau nhưng không thể trao đổi trực tiếp
- Người có nhu cầu chia sẻ tài nguyên như file, ảnh,...

#### 2.2.2. Các chức năng của chương trình

- Chức năng đăng nhập, đăng ký
- Chức năng xem danh sách người dùng, người dùng đang online
- Chức năng gửi, nhận tin nhắn real-time, gửi emoji cảm xúc
- Chức năng gửi file, gửi ảnh.

### 2.2.3. Công nghệ sử dụng

- Web API
- Socket IO

## 2.3. THIẾT KẾ CƠ SỞ DỮ LIỆU

Chương trình sử dụng công cụ MySQL để thiết kế cơ sở dữ liệu.

Cơ sở dữ liệu gồm 3 table chính:

- Table Account: Dùng để lưu trữ thông tin tài khoản người dùng.

#	Tên	Kiểu	Bảng mã đối chiếu	Thuộc tính	Null	Mặc định	Ghi chú	Thêm	Hành động
<input type="checkbox"/> 1	id	int(11)			Không	Không		AUTO_INCREMENT	Thay đổi  Xóa  Thêm
<input type="checkbox"/> 2	username	varchar(50)	utf8_bin		Không	Không			Thay đổi  Xóa  Thêm
<input type="checkbox"/> 3	password	varchar(50)	utf8_bin		Không	Không			Thay đổi  Xóa  Thêm

Hình 8 Table Account

- Table User: Dùng để lưu trữ thông tin người dùng

#	Tên	Kiểu	Bảng mã đối chiếu	Thuộc tính	Null	Mặc định	Ghi chú	Thêm	Hành động
<input type="checkbox"/> 1	id	int(11)			Không	Không			Thay đổi  Xóa  Thêm
<input type="checkbox"/> 2	username	varchar(50)	utf8_bin		Không	Không			Thay đổi  Xóa  Thêm
<input type="checkbox"/> 3	gender	tinyint(4)			Không	0			Thay đổi  Xóa  Thêm
<input type="checkbox"/> 4	image	varchar(500)	utf8mb4_bin		Có				Thay đổi  Xóa  Thêm
<input type="checkbox"/> 5	status	tinyint(4)			Không	0			Thay đổi  Xóa  Thêm

Hình 9 Table User

- Table Message: Dùng để lưu trữ nội dung tin nhắn của người dùng.

#	Tên	Kiểu	Bảng mã đối chiếu	Thuộc tính	Null	Mặc định	Ghi chú	Thêm	Hành động
<input type="checkbox"/>	1 id	int(11)		Không	Không		AUTO_INCREMENT		Thay đổi  Xóa  Thêm
<input type="checkbox"/>	2 idUserSend	int(11)		Không	Không				Thay đổi  Xóa  Thêm
<input type="checkbox"/>	3 idUserReceive	int(11)		Không	Không				Thay đổi  Xóa  Thêm
<input type="checkbox"/>	4 type	int(11)		Không	1				Thay đổi  Xóa  Thêm
<input type="checkbox"/>	5 text	text	utf8_bin	Không	"				Thay đổi  Xóa  Thêm
<input type="checkbox"/>	6 filepath	varchar(500)	utf8_bin	Không					Thay đổi  Xóa  Thêm
<input type="checkbox"/>	7 filename	varchar(500)	utf8_bin	Không					Thay đổi  Xóa  Thêm

Hình 10 Table Message

## 2.4. TỔ CHỨC CHƯƠNG TRÌNH

### 2.4.1. Tổ chức thư mục

Chương trình được chia làm hai chương trình con là server và client với mỗi phần thực hiện một chức năng khác nhau.

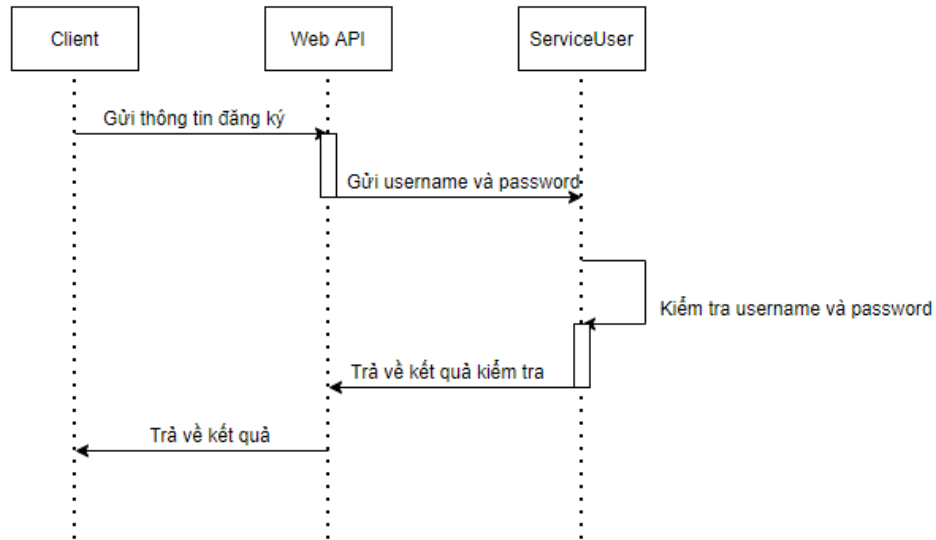
### 2.4.2. Server

Server có nhiệm vụ lắng nghe kết nối, nhận dữ liệu, lưu trữ thông tin và phản hồi client. Server gồm có 3 module chính: Web API, Socket IO, truy vấn database.

#### b. Web API

Module này có chức năng xác thực và xử lý thông tin tài khoản của người dùng khi người dùng thực hiện đăng ký hoặc đăng nhập vào hệ thống

- Đăng ký



Hình 11 Đăng ký tài khoản

Khi người dùng đăng ký tài khoản ở phía client, client sẽ thực hiện gọi Web API ở phía server để truyền dữ liệu tài khoản đăng ký (dữ liệu được truyền ở dạng **json**) về phía server. Ở phía server, hàm **Register** tương ứng với sự kiện đăng ký sẽ nhận dữ liệu sau đó thực hiện truy vấn đến cơ sở dữ liệu rồi trả lại kết quả cho client.

```

@PostMapping("/Register")
public Model_Error Register(@RequestBody Model_Account data) {
    System.out.println(data.getUsername());
    System.out.println(data.getPassword());
    return ServiceUser.Instance().Register(data);
}
  
```

Hình 12 API Register

**Dữ liệu dạng Json:**

```

{
    "username": "thanhlang",
    "password": "123"
}
  
```

- **Đăng nhập**

Việc đăng nhập về cơ bản giống như đăng kí người dùng khi mà server cũng lấy thông tin tài khoản người dùng từ client gửi về sau đó truy cập vào database để xác thực tài khoản. Sau đó trả kết quả lại cho client.

Hàm xử lý Đăng nhập ở server:

```
@PostMapping("/Login")
public ResponseEntity<Model_User> Login(@RequestBody Model_Account login) {
    try {
        Model_User user = ServiceUser.Instance().login(login);
        if(user != null) {
            return new ResponseEntity<Model_User>(user, HttpStatus.OK);
        }
        else {
            return new ResponseEntity<Model_User>(HttpStatus.NOT_FOUND);
        }
    } catch (Exception e) {
        return new ResponseEntity<Model_User>(HttpStatus.NOT_FOUND);
    }
}
```

*Hình 13 API Login*

### c. Socket IO

Module Socket IO có chức năng chính trong chương trình ở phía server. Phụ trách nhiều tác vụ như: cập nhật danh sách người dùng, cập nhật trạng thái hoạt động của người dùng, xử lý nhắn tin real-time, xử lý gửi nhận file và ảnh.

- **Cập nhật danh sách người dùng**

Khi người dùng đăng nhập vào hệ thống, client sẽ tự động gửi 1 yêu cầu lấy danh sách người dùng, server lắng nghe và nhận được yêu cầu từ phía client. Sau đó, server sẽ truy cập vào database để lấy danh sách người dùng rồi trả ngược lại phía client.

- **Cập nhật trạng thái hoạt động của người dùng**

Khi bất kì một người dùng nào đăng nhập hoặc đăng ký vào hệ thống, chương trình client ở phía người dùng vừa đăng nhập đây sẽ gửi một yêu cầu cập nhật trạng thái đến server. Khi server nhận được yêu cầu sẽ thay đổi trạng thái

người dùng vừa đăng nhập hoặc đăng ký từ offline sang online và gửi lại cho tất cả người dùng khác thông qua phương thức **getBroadcastOperations().sendEvent()**.

- **Sử lý nhắn tin real-time**

Khi user1 gửi tin nhắn đến user2, chương trình client bên phía user1 sẽ gửi dữ liệu đến phía server. Khi server nhận được dữ liệu sẽ gửi ngược lại user2, sau đó truy cập database để lưu dữ liệu tin nhắn vừa được gửi.

- **Sử lý gửi nhận file hoặc ảnh**

Khi user1 gửi file hoặc ảnh đến user2, chương trình client bên phía user1 sẽ gửi dữ liệu đến phía server. Khi server nhận được dữ liệu sẽ thực hiện gửi dữ liệu đến user2 đồng thời tạo ra 1 luồng riêng biệt để thực hiện ghi file dữ liệu vừa nhận được và sau khi ghi xong sẽ truy cập database để lưu trữ thông tin file vừa lưu.

#### **d. Truy vấn database**

Module này chủ yếu truy xuất database để lưu trữ dữ liệu hoặc lấy dữ liệu của người dùng.

### **2.4.3. Client**

Client có chức năng lấy dữ liệu từ phía server và hiển thị dữ liệu lên giao diện người dùng. Gồm các module chính: Call API, Socket IO, View.

#### **a. Call API**

Module này có chức năng gọi Web API từ phía server để xác thực tài khoản người dùng khi người dùng thực hiện đăng ký hoặc đăng nhập.

Module sử dụng lớp **HttpClient**, **HttpRequest**, **HttpResponse** để gửi và nhận dữ liệu.

- **Đăng ký**

Lấy dữ liệu tài khoản khi người dùng đăng ký sau đó gửi dữ liệu đến server. Sau đó nhận lại kết quả từ phía server.

Hàm gọi API Register:

```
public void Register(Model_Account data, EventErrorMessage error) {
    try {
        HttpClient client = HttpClient.newHttpClient();
        HttpRequest request = HttpRequest.newBuilder().uri(URI.create("http://" + IP + ":8081/Register"))
            .header("Content-Type", "application/json; charset=UTF-8")
            .POST(BodyPublishers.ofString(data.ToObjectJson().toString()))
            .build();

        HttpResponse<String> response = client.send(request, BodyHandlers.ofString());
        JSONObject obj = new JSONObject(response.body());
        Model_Error message = new Model_Error(obj.getBoolean("action"), obj.getString("message"));
        if(message.isAction()) {
            Model_User user = new Model_User(obj.getJSONObject("data"));
            this.user = user;
            this.client.emit("login", user.toJsonObject());
        }
        error.callErrorMessage(message);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

*Hình 14 Call API Register*

- **Đăng nhập**

Việc đăng nhập về cơ bản giống như đăng kí người dùng khi mà client cũng lấy thông tin tài khoản người dùng từ client để gửi về server sau đó nhận lại phản hồi từ server.

Hàm gọi API Login:

```
public void Login(Model_Account data) {
    try {
        HttpClient client = HttpClient.newHttpClient();
        HttpRequest request = HttpRequest.newBuilder().uri(URI.create("http://" + IP + ":8081/Login"))
            .header("Content-Type", "application/json; charset=UTF-8")
            .POST(BodyPublishers.ofString(data.ToObjectJson().toString()))
            .build();

        HttpResponse<String> response = client.send(request, BodyHandlers.ofString());
        if(response.statusCode() == 200) {
            JSONObject obj = new JSONObject(response.body());
            Model_User user = new Model_User(obj);
            this.user = user;
            this.client.emit("login", user.toJsonObject());
            PublicEvent.getInstance().getEventMain().showLoading(false);
            PublicEvent.getInstance().getEventMain().initChat();
        }
        else {
            PublicEvent.getInstance().getEventMain().showLoading(true);
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

*Hình 15 Call API Login*



## **b. Socket IO**

Module Socket IO có chức năng chính trong chương trình ở phía client. Phụ trách nhiều tác vụ như: cập nhật danh sách người dùng, cập nhật trạng thái hoạt động của người dùng, xử lý nhắn tin real-time, xử lý gửi nhận file và ảnh.

- **Cập nhật danh sách người dùng**

Sau khi đăng nhập hoặc đăng ký thành công, client sẽ gửi yêu cầu cập nhật danh sách người dùng đến server thông qua phương thức **socket.emit()**. Sau đó nhận được danh sách người dùng từ server thông qua phương thức **socket.on()** sau đó hiển thị danh sách người dùng lên giao diện người dùng.

- **Cập nhật trạng thái hoạt động**

- **Dưới góc nhìn của người dùng vừa đăng nhập hoặc đăng kí**

Chương trình client sẽ tự động gửi yêu cầu cập nhật trạng thái hoạt động ngay khi người dùng vừa thực hiện đăng nhập hoặc đăng ký thành công vào hệ thống.

- **Dưới góc nhìn của người dùng đang online**

Chương trình client sẽ có một phương thức lắng nghe sự kiện cập nhật trạng thái người dùng (sử dụng **socket.on()**). Phương thức này sẽ nhận dữ liệu từ phía server gửi lên sau đó cập nhật trạng thái người dùng trên giao diện.

- **Sử lý nhắn tin real-time**

- **Dưới góc nhìn của người dùng gửi tin nhắn**

Chương trình client sẽ nhận dữ liệu tin nhắn sau đó gửi đến sever

- **Dưới góc nhìn của người dùng nhận tin nhắn**

Chương trình client sẽ có một phương thức lắng nghe để nhận dữ liệu từ phía server gửi đến, sau đó sẽ tải tin nhắn vừa nhận được lên giao diện người dùng.

- **Sử lý gửi nhận file hoặc ảnh**

- **Dưới góc nhìn của người dùng gửi**

Client sẽ đọc dữ liệu dạng byte sau đó gửi tuần tự nhiều mảng byte về phía client (Vì phương thức socket giới hạn độ lớn dữ liệu truyền tải nên khi truyền dữ liệu lớn như file cần phải gửi từng mảng về phía server sử dụng phương thức RandomAccessFile)

- **Dưới góc nhìn của người nhận**

Client sẽ có một phương thức lắng nghe để nhận dữ liệu file từ phía server. Sau khi nhận được dữ liệu sẽ thực hiện ghi file đồng thời tải thông tin file lên giao diện người dùng.

### **c. View**

Module này có chức năng chính là tương tác với người dùng, lấy và hiển thị dữ liệu cho người dùng.

## **2.5. KẾT CHƯƠng**

Chương này trình bày hướng giải quyết của bài toán, các chức năng chính của chương trình, các công nghệ sử dụng và tổ chức chương trình. Ngoài ra chương còn trình bày các thuật toán để giải quyết vấn đề nhận và gửi tin nhắn trong ứng dụng.

## CHƯƠNG 3: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

*Chương này trình bày về kết quả triển khai hệ thống, cấu hình hệ thống và các thành phần chức năng. Kết quả được đánh giá thông qua các kịch bản thực nghiệm khác nhau nhằm thể hiện các ưu/nhược của giải pháp đề xuất.*

### 3.1. MÔ HÌNH TRIỂN KHAI

#### 3.1.1. Mô hình triển khai

Chương trình được triển khai theo mô hình client – server.

#### 3.1.2. Các công cụ sử dụng

- Ngôn ngữ lập trình: java
- IDE: Eclipse

#### 3.1.3. Cấu hình hệ thống

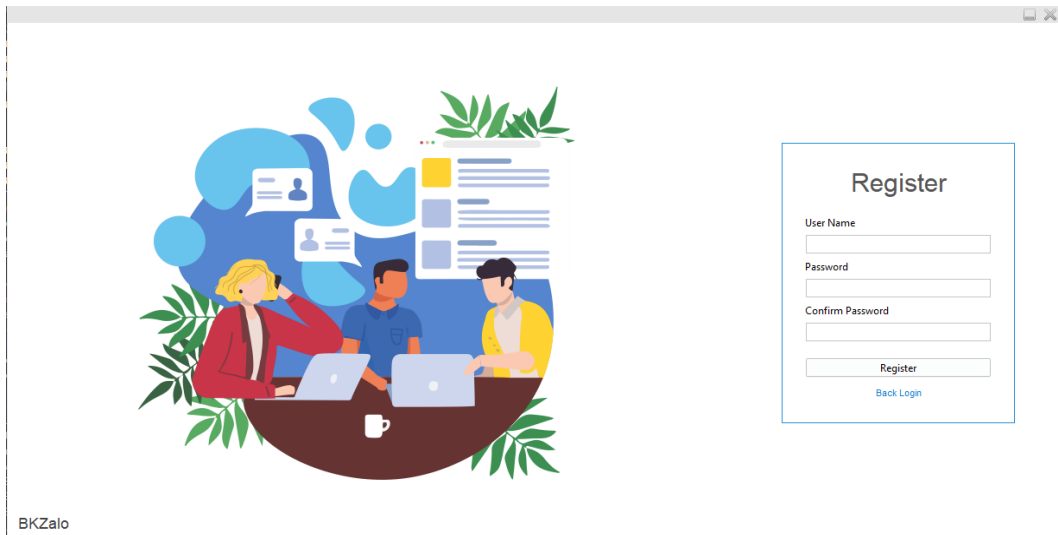
- Phiên bản java: jdk 16
- Tomcat: v8.5

### 3.2. KẾT QUẢ THỰC NGHIỆM

#### 3.2.1. Đăng ký, đăng nhập vào hệ thống

##### a. Đăng ký:

- Người dùng điền thông tin tài khoản đăng ký đầy đủ vào các trường sau đó nhấn đăng ký.
- Nếu đăng ký thành công, người dùng có thể truy cập vào hệ thống.
- Nếu đăng ký không thành công, hệ thống sẽ thông báo lỗi cho người dùng
- Người dùng có thể thực hiện đăng ký lại

The image shows a web browser window with a registration form titled "Register". On the left side of the page is a large, colorful illustration of three people (a woman and two men) sitting around a table with laptops, surrounded by green foliage and blue abstract shapes. The registration form on the right contains the following fields and elements:

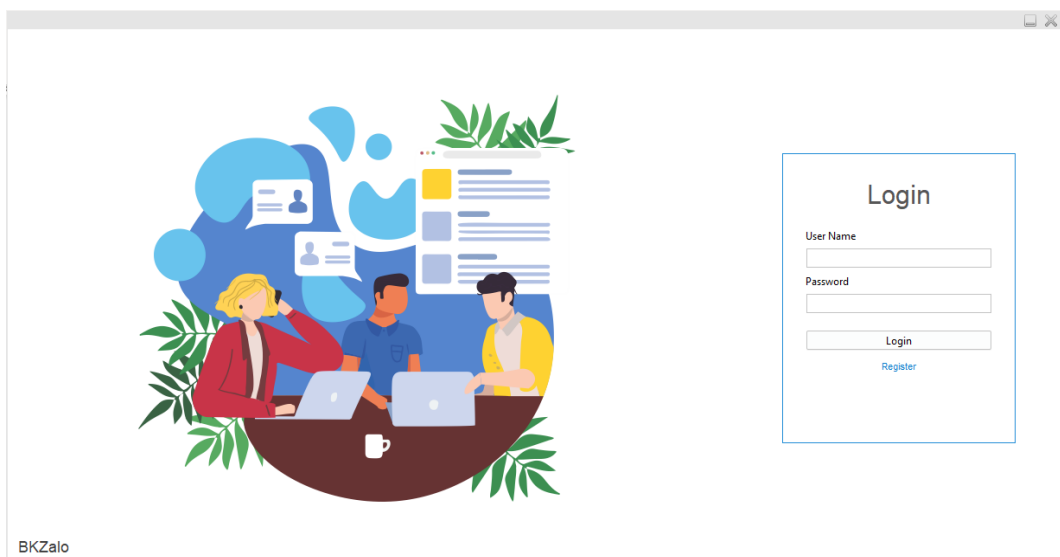
- Register** (Section Header)
- User Name** (Label) followed by a text input field.
- Password** (Label) followed by a text input field.
- Confirm Password** (Label) followed by a text input field.
- Register** (Submit Button)
- [Back Login](#) (Link)

The browser's address bar shows "BKZalo".

Hình 16 Giao diện đăng ký

## b. Đăng nhập

- Người dùng điền thông tin tài khoản vào đầy đủ các trường sau đó nhấn đăng nhập
- Nếu thông tin chính xác, người dùng có thể truy cập vào hệ thống
- Nếu thông tin không chính xác, hệ thống sẽ thông báo đến người dùng.

The image shows a web browser window with a login form titled "Login". It features the same large illustration on the left as the registration page. The login form on the right contains the following fields and elements:

- Login** (Section Header)
- User Name** (Label) followed by a text input field.
- Password** (Label) followed by a text input field.
- Login** (Submit Button)
- [Register](#) (Link)

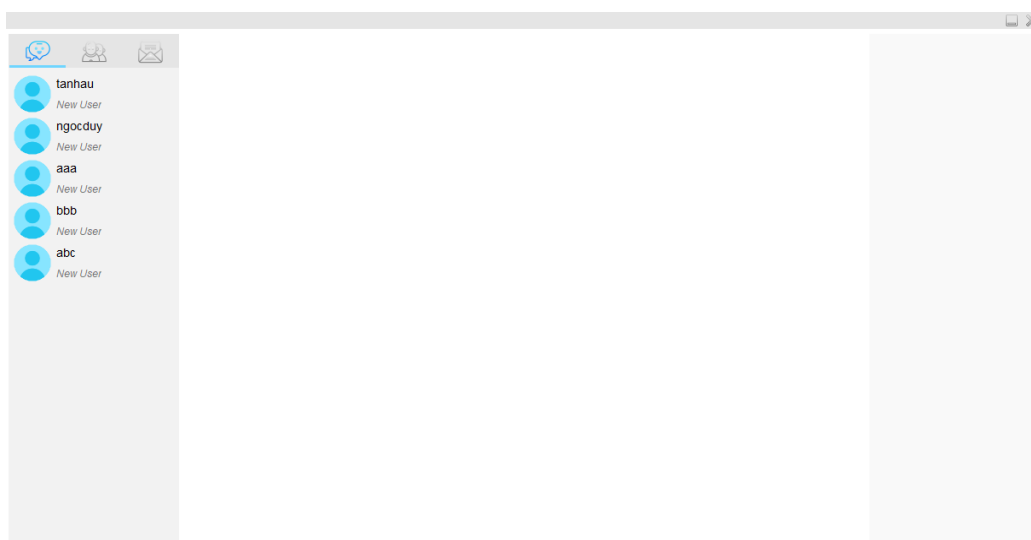
The browser's address bar shows "BKZalo".

Hình 17 Giao diện đăng nhập

### 3.2.2. Chức năng xem danh sách người dùng, cập nhật trạng thái người dùng

#### a. Chức năng xem danh sách người dùng

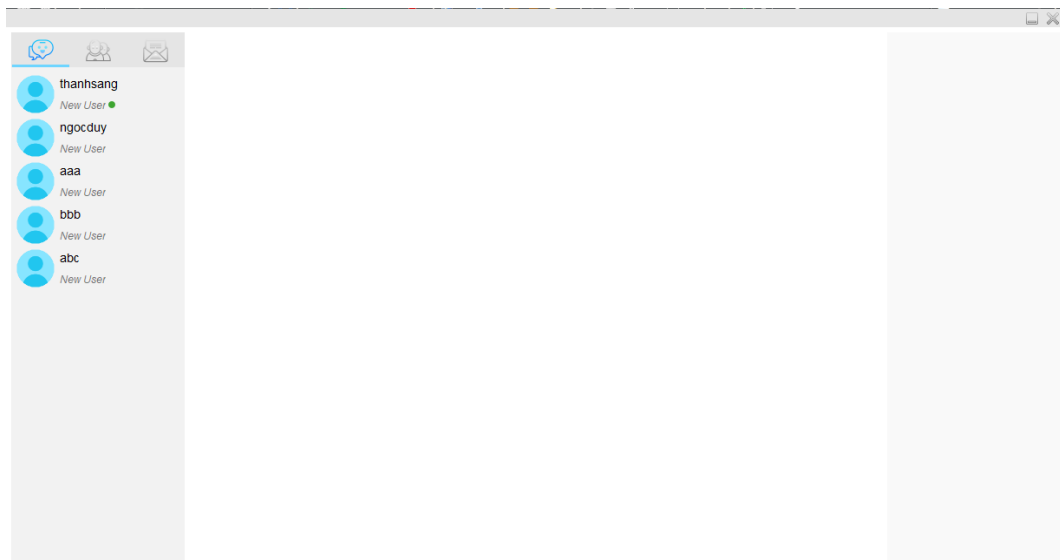
- Sau khi đăng nhập vào hệ thống, người dùng sẽ có thể xem được danh sách người dùng.
- Khi có bất kì người dùng nào đăng ký mới vào hệ thống, danh sách sẽ được tự động làm mới.



Hình 18 Danh sách người dùng

#### b. Cập nhật trạng thái người dùng

- Sau khi đăng nhập hoặc đăng ký thành công vào hệ thống, người dùng có thể xem được danh sách người dùng đồng thời là trạng thái hoạt động của họ.
- Nếu có người dùng nào đăng nhập hoặc đăng ký vào hệ thống, trạng thái người dùng sẽ tự động cập nhật lại.

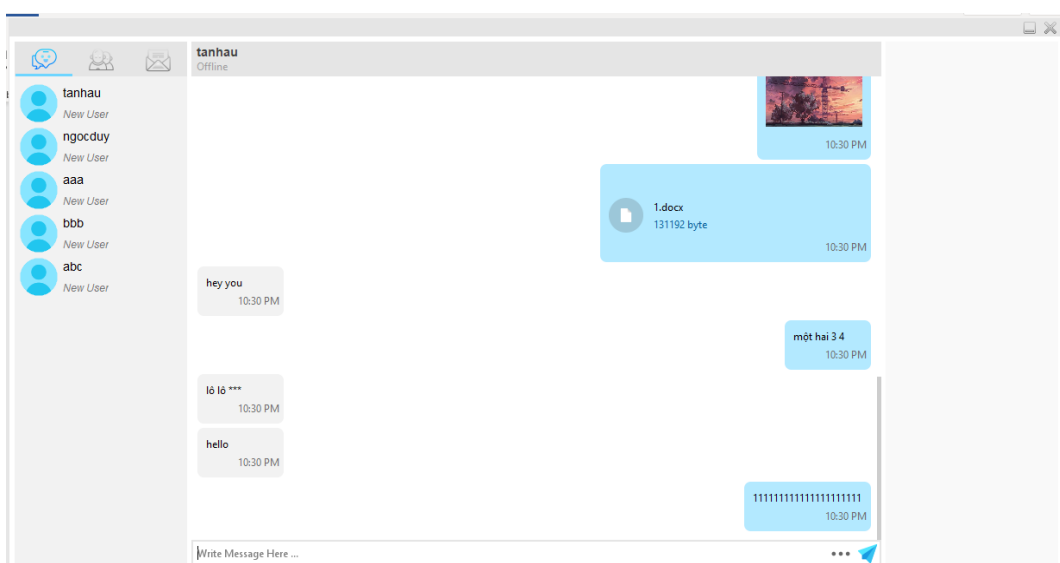


*Hình 19 Cập nhật trạng thái hoạt động*

### 3.2.3. Chức năng nhắn tin, gửi emoji, gửi ảnh, gửi file

#### a. Nhắn tin

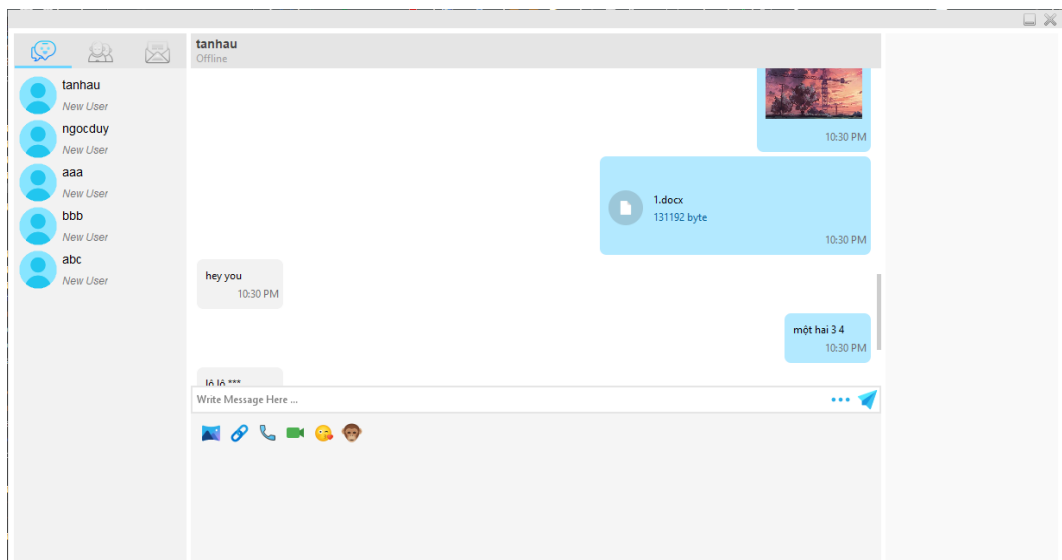
- Người dùng nhập tin nhắn vào boxchat sau đó nhấn gửi
- Giao diện người gửi sẽ tự động cập nhật tin nhắn lên khung chat
- Giao diện người nhận sau khi nhận được tin nhắn sẽ hiển thị lên khung chat



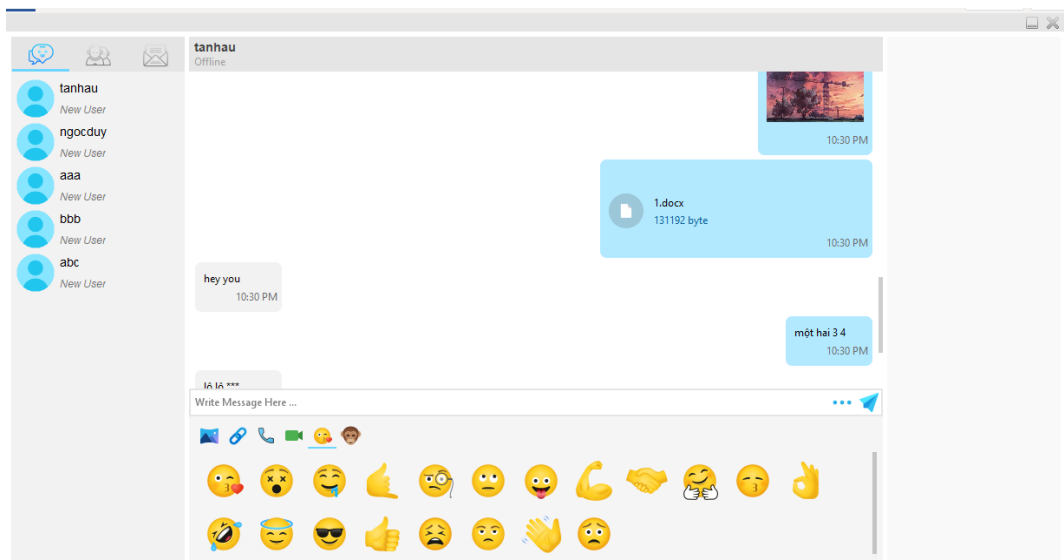
*Hình 20 Giao diện nhắn tin*

## b. Nhấn Emoji

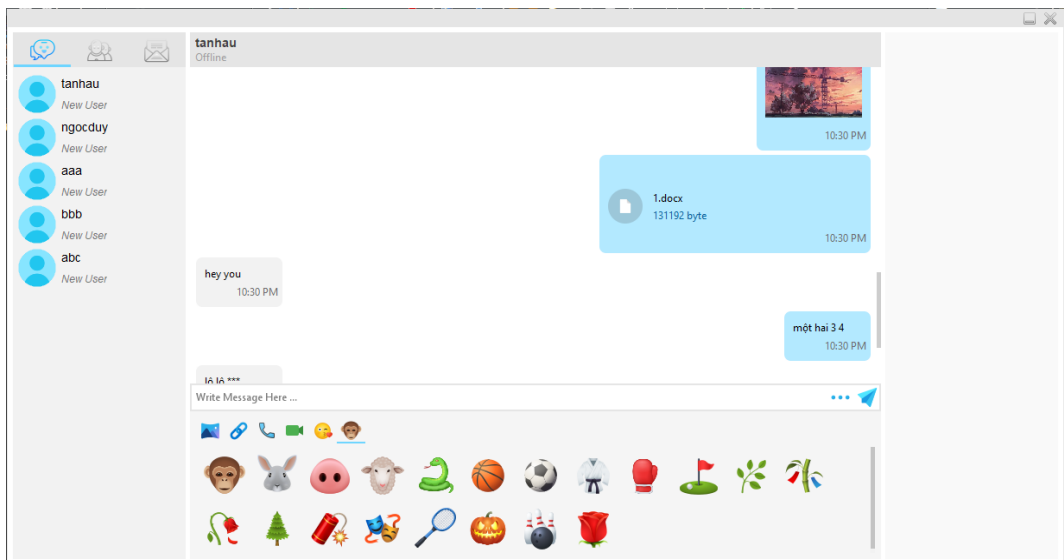
- Người dùng nhấn vào biểu tượng dấu 3 chấm trên boxchat để hiển thị panel more
- Nhấn vào biểu tượng emoji trên panel more để tải danh sách emoji
- Nhấn vào biểu tượng emoji muốn gửi, sau đó hệ thống sẽ thực hiện gửi emoji



*Hình 21 Panel more*



Hình 22 Danh sách emoji loại 1



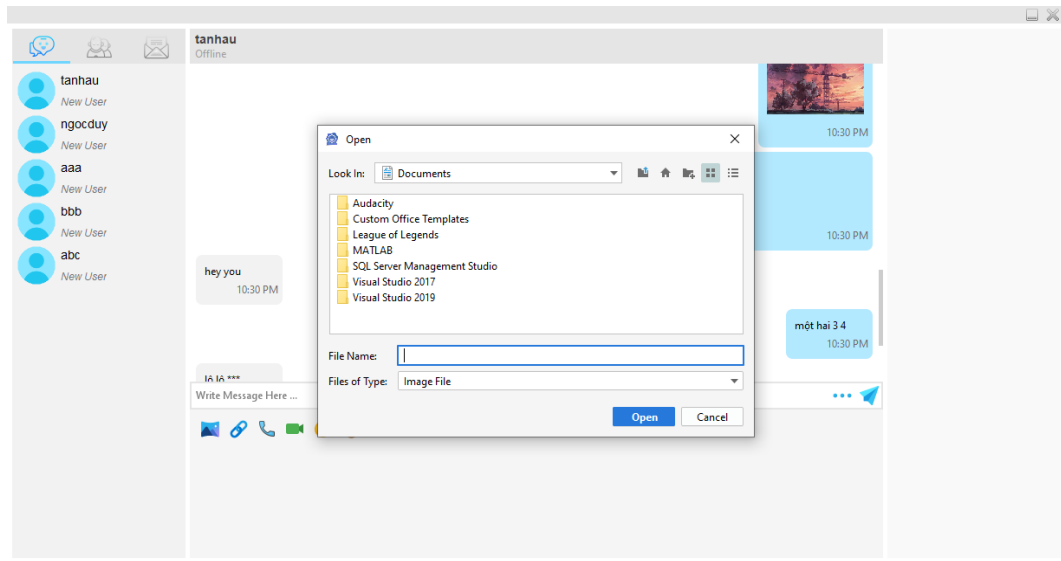
Hình 23 Danh sách emoji loại 2

### c. Gửi ảnh, gửi file

- Trên panel more, chọn biểu tượng hình ảnh hoặc biểu tượng file để gửi ảnh hoặc gửi file.
- Nếu nhấn vào gửi ảnh, người dùng chỉ được chọn các file có đuôi là ảnh như **jpg, png, jpeg, gif, ...**



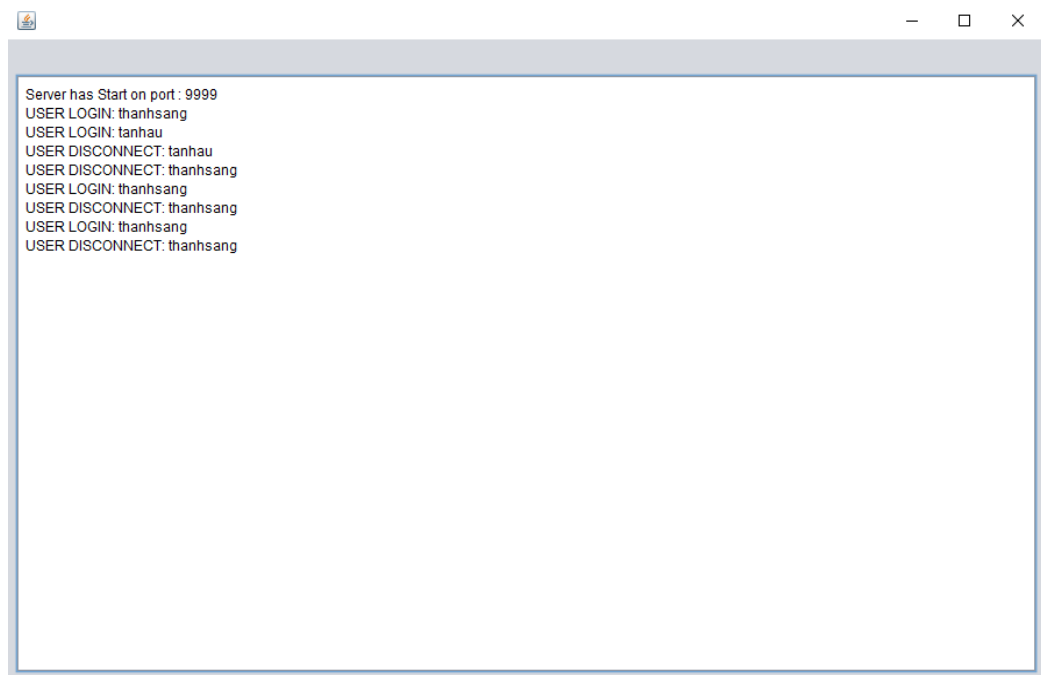
- Nếu nhấn vào gửi file, người dùng chỉ được chọn các file có đuôi là: **doc, docx, pdf, xlsx, txt, ...**
- Sau khi nhấn vào một trong hai biểu tượng, một cửa sổ mới sẽ hiện ra cho phép lựa chọn file cần gửi.
- **Nhấn đúp chuột** vào file cần gửi hoặc **click open**, sau đó hệ thống sẽ thực hiện gửi file.



*Hình 24 Gửi file, gửi ảnh*

### 3.2.4. Giao diện server

- Khi chạy server, giao diện server sẽ hiển thị trạng thái của server và port server.
- Khi có người đăng nhập hoặc đăng ký, giao diện sẽ hiển thị hành động của người dùng và tên người dùng thực hiện hành động đó.
- Khi có người dùng ngắt kết nối, giao diện sẽ hiển thị tên người dùng vừa ngắt kết nối.



*Hình 25 Giao diện server*

### **3.3. NHẬN XÉT ĐÁNH GIÁ KẾT QUẢ**

- Chương trình đã thực hiện được các điều kiện mà bài toán đặt ra, tuy nhiên nếu muốn đưa vào sử dụng cần thêm nhiều chức năng hơn.
- Giao diện của chương trình cũng chưa tối ưu và thân thiện với người dùng, cần cải thiện, nâng cấp để chương trình có một giao diện đẹp hơn và thân thiện với người dùng hơn.

### **3.4. KẾT CHƯƠNG**

Chương này trình bày về môi trường để cài đặt, kết quả triển khai và đánh giá nhận xét kết quả chương trình.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 1. KẾT QUẢ ĐẠT ĐƯỢC

Trong thời gian tìm hiểu, nghiên cứu cơ sở lý thuyết và triển khai ứng dụng công nghệ, đồ án đã đạt được những kết quả sau:

Về mặt lý thuyết, đồ án đã giúp hiểu rõ hơn về cách tạo ra và cơ chế hoạt động của một Web API, hiểu được phương thức hoạt động của Socket IO cũng như hiểu rõ hơn về mô hình client – server.

Về mặt thực tiễn ứng dụng, đồ án đã tạo được ứng dụng trao đổi file và ứng dụng nhắn tin đơn giản đồng thời phát hiện được nhiều vấn đề phát sinh khi lập trình với Java, từ đó tích góp thêm được kinh nghiệm khi làm việc với ngôn ngữ này.

Tuy nhiên, vẫn còn tồn tại các vấn đề như sau:

- Ứng dụng còn thiếu sót nhiều chức năng.
- Giao diện còn khá đơn giản và nhiều phần vẫn chưa được tối ưu.

## 2. KIẾN NGHỊ VÀ HƯỚNG PHÁT TRIỂN

Một số hướng nghiên cứu và phát triển của đề tài như sau:

- Bổ sung và hoàn thiện một số chức năng của hệ thống
- Đánh giá hiệu năng trên các môi trường khác nhau
- Kiểm thử các chức năng của chương trình.
- Bổ sung các giải pháp bảo mật và an toàn cho hệ thống.

## TÀI LIỆU THAM KHẢO

### Tiếng Việt

- [1] Đặng Văn Đức (2001), *Hệ thống thông tin địa lý*, Nhà xuất bản Khoa học và Kỹ Thuật Hà Nội.
- [2] Phạm Hữu Đức (2005), *Cơ sở dữ liệu và hệ thống thông tin địa lý GIS*, Nhà xuất bản Xây dựng.

### Tiếng Anh

- [3] Senthil Shanmugan (2004). “*Digital urban management programme - Evolution of Bangalore GIS model*”, Proceedings of the third International Conference on Environment and Health, India.
- [4] Teemu Nuortio, Harri Niska (2003), “*Improved route planning and scheduling of waste collection and transport*”, Department of Environmental Sciences, University of Kuopio, Finland.

### Internet

- [5] [http:// www.mapreduce.org](http://www.mapreduce.org)
- [6] [http:// www.vre.cse.hcmut.edu.vn](http://www.vre.cse.hcmut.edu.vn)