ỦY BAN NHÂN DÂN THÀNH PHỐ HỒ CHÍ MINH

**TRƯỜNG ĐẠI HỌC SÀI GÒN**

**TIỂU LUẬN MÔN HỌC**

**LẬP TRÌNH MẠNG (841307)**

**ĐỀ TÀI 1**

|  |  |
| --- | --- |
| **Sinh viên thực hiện:** | **Trần Nhật Duy** |
| **MSSV:** | **3118410061** |
|  |  |
| **Ngành:** | **CÔNG NGHỆ THÔNG TIN** |
| **TRÌNH ĐỘ ĐÀO TẠO:** | **ĐẠI HỌC** |

**THÀNH PHỐ HỒ CHÍ MINH, THÁNG 1 NĂM 2022**

ỦY BAN NHÂN DÂN THÀNH PHỐ HỒ CHÍ MINH

**TRƯỜNG ĐẠI HỌC SÀI GÒN**

**TIỂU LUẬN MÔN HỌC**

**LẬP TRÌNH MẠNG (841307)**

**ĐỀ TÀI: 1**

|  |  |  |
| --- | --- | --- |
| **PHẦN THÔNG TIN CÁ NHÂN CỦA SINH VIÊN** | | |
| Họ và tên sinh viên | **TRẦN NHẬT DUY** | |
| Mã số sinh viên | **3118410061** | |
| Lớp | **DCT11810** | |
| Phòng thi | **002** | |
| Ngày thi | **15/01/2021** | |
| Giờ thi | **13g00** | |
| **PHẦN THÔNG TIN DÀNH CHO CÁN BỘ CHẤM THI** | | |
| **CBCT1** | | **CBCT2** |
|  | |  |

MỤC LỤC

[CÂU 1: 1](#_Toc92576765)

[CÂU 2: 2](#_Toc92576766)

[Phần I: Giới thiệu ứng dụng “Chat với người lạ” 2](#_Toc92576767)

[1.1 Giới thiệu chung 2](#_Toc92576768)

[1.2 Lý do chọn đề tài 2](#_Toc92576769)

[1.3 Chức năng của đề tài 2](#_Toc92576770)

[Phần II: Phân tích và thiết kế 3](#_Toc92576771)

[2.1 Các tín hiệu truyền tin và cách hoạt động 3](#_Toc92576772)

[2.1.1 Khởi động chương trình 3](#_Toc92576773)

[2.1.2 Ghép đôi chat 3](#_Toc92576774)

[2.1.3 Thoát khỏi phiên chat 4](#_Toc92576775)

[2.1.4 Gửi tin nhắn 5](#_Toc92576776)

[2.1.5 Thoát ứng dụng 5](#_Toc92576777)

[2.1.5 Cập nhật số lượng người online 5](#_Toc92576778)

[2.2 Sơ đồ Flowchart 6](#_Toc92576779)

[2.2.1 Khởi động chương trình: 6](#_Toc92576780)

[2.2.2 Ghép đôi chat 9](#_Toc92576781)

[2.2.3 Thoát khỏi phiên chat 12](#_Toc92576782)

[2.2.4 Gửi tin nhắn 13](#_Toc92576783)

[2.2.5 Thoát ứng dụng 15](#_Toc92576784)

[Phần III: Kết quả thực thi 16](#_Toc92576785)

[Lời cảm ơn: 21](#_Toc92576786)

[Nhận xét của giảng viên: 21](#_Toc92576787)

[Tham khảo: 21](#_Toc92576788)

PHẦN BÀI LÀM

# CÂU 1:

STREAM SOCKET

Stream Socket là socket hướng kết nối, là socket hoạt động qua giao thức TCP (Transmisstion Control Protocol). Stream Socket chỉ hoạt động khi server và client đã kết nối với nhau.

Ưu điểm:

* Dữ liệu truyền đi được đảm bảo truyền đến đúng nơi nhận, đúng thứ tự với thời gian nhanh chóng.
* Mỗi thông điệp gửi đi đều có xác nhận trả về để thông báo cho người dùng thông tin về quá trình truyền tải.

Nhược điểm:

* Giữa máy chủ và máy nhận chỉ có 1 IP, nên khi kết nối, 1 máy phải chờ máy còn lại chấp nhận kết nối.

DATAGRAM SOCKET

Datagram socket là socket không hướng kết nối, là socket hoạt động thông qua giao thức UDP (User Datagram Protocol). Datagram socket có thể hoạt động kể cả khi không có sự thiết lập kết nối giữa 2 máy với nhau.

Ưu điểm:

* Quá trình kết nối và truyền tải thông tin đơn giản, không cần thực hiện nhiều thao tác.
* Thời gian truyền tải dữ liệu nhanh.

Nhược điểm:

* Quá trình truyền thông tin không đảm bảo tin cậy, thông tin có thể truyền sai thứ tự hoặc bị lặp.

So sánh TCP và UDP:

* Giống nhau: đều là các giao thức mạng, có chức năng kết nối các máy tính lại với nhau.
* Khác nhau:

|  |  |
| --- | --- |
| TCP | UDP |
| Dùng cho mạng WAN | Dùng cho mạng LAN |
| Không cho phép mất gói tin | Cho phép mất gói tin |
| Đảm bảo việc truyền dữ liệu | Không đảm bảo |
| Tốc độ thấp | Tốc độ cao |
| Hoạt động theo hướng kết nối | Hoạt động theo hướng không kết nối |

# CÂU 2:

# Phần I: Giới thiệu ứng dụng “Chat với người lạ”

## Giới thiệu chung

Ứng dụng chat với người lạ được lấy ý tưởng từ tên bài hát “Tâm sự với người lạ” là một ứng dụng chat đơn giản sử dụng kỹ thuật multithreading và Stream Socket, hoạt động theo mô hình Client – Server. Dù chỉ là ứng dụng đơn giản nhưng các giai đoạn từ phân tích đến xây dựng và thực thi chương trình đều được thực hiện một cách tỉ mỉ nhất, chính xác nhất.

Đúng với tên gọi, “Chat với người lạ” sẽ mang đến cho bạn những trải nghiệm thú vị nhất khi trò chuyện với người lạ để biết được những kiểu cách giao tiếp với những người khác nhau, hoặc hơn nữa là trải lòng, chia sẽ những tâm sự của bản thân với một người hoàn toàn xa lạ. Đôi khi, điều đấy lại là “có vẻ ổn nhất” khi chúng ta cần một ai đó để trút bầu tâm sự của mình nhưng không phải với những người mà ta quen biết (vì biết đâu là chuyện khó nói, hoặc là sau khi nói ra lại trở thành điều không hay hoặc có lẽ là ta muốn nói với ai đó và sau khi nói xong lại muốn rút lại). Chính vì lẽ đó, mà ứng dụng “Chat với người lạ” được xây dựng để thực hiện việc đấy.

## Lý do chọn đề tài

Nắm vững kỹ năng, kiến thức liên quan đến môn Lập trình mạng nói chung và kỹ thuật multithreading, socket nói riêng.

Biết được cách hoạt động của socket, thread.

Tìm hiểu kiến thức mới, ứng dụng multithread kết hợp với giao diện. Từ đó biết cách xây dựng được ứng dụng hoàn chỉnh.

## Chức năng của đề tài

+ Chức năng chính: chat.

+ Hiển thị số người tham gia.

+ Random người dùng ngẫu nhiên để ghép đôi chat.

# Phần II: Phân tích và thiết kế

## 2.1 Các tín hiệu truyền tin và cách hoạt động

### 2.1.1 Khởi động chương trình

Tín hiệu bên Server: 0, 1.

+ Server:

Khi chạy chương trình, server sẽ mở port, chờ và chấp nhận khi có client kết nối.

+ Client:

Khi chạy chương trình, hiện lên giao diện để cho người dùng nhập nickname, khi nhập xong, người dùng sẽ nhấn “OK”. Lúc đó sẽ gửi tín hiệu kết nối với dữ liệu là nickname đã nhập đến server.

Server đồng ý và sẽ kiểm tra nickname mà người dùng đã nhập:

- Nếu nickname chưa tồn tại thì Server sẽ gửi về “1”, đồng thời sẽ thêm vào list. Tạo luồng mới để Server có thể phục vụ (giao tiếp riêng) với Client này. Và phía Client cũng sẽ mở luồng nhận riêng để nhận tin từ server.

- Nếu nickname đã tồn tại thì Server sẽ gửi về “0”, để yêu cầu người dùng nhập lại.

### 2.1.2 Ghép đôi chat

Tín hiệu bên Server: WANT TO CHAT, USERISCHATTING, WAIT, DECLINE, ACCEPTED.

Tín hiệu bên Client: CREATE, IACCEPT, NO.

+ Client (gửi):

Khi client đăng nhập thành công, client sẽ ở trạng thái chờ. Nếu muốn chat, client sẽ gửi click button “Bắt đầu chat” và sẽ gửi tín hiệu “CREATE” đến server.

Client sẽ nhận các phản hồi từ Server:

- WAIT: là chờ người khác tham gia.

- USERISCHATTING: là người dùng đã được ghép đôi chat.

- DECLINE: người dùng được chọn đã từ chối chat với bạn.

- ACCEPTED: người dùng được chọn ngẫu nhiên đồng ý chat với bạn. Hai bên sẽ bắt đầu chat.

+ Server:

Server nhận tín hiệu “CREATE”, server sẽ kiểm tra:

- Nếu chỉ có một người client đó tham gia, Server sẽ trả về “WAIT”. Client sẽ phải chờ người khác tham gia.

- Nếu người dùng đang bận (đang được ghép cặp với người khác) thì Server sẽ gửi lại tín hiệu “USERISCHATTING”. Client sẽ chờ người người tham gia mới hoặc là chờ những người đang được ghép đôi thoát chat.

- Nếu ngoài hai trường hợp trên, Server sẽ chọn ngẫu nhiên một người (Client) trong danh sách và gửi tín hiệu “WANT TO CHAT” đến người đó.

Sau khi nhận được phản hồi từ Client ngẫu nhiên:

- Nếu là “NO”: Server sẽ gửi lại cho Client yêu cầu tín hiệu “DECLINE”.

- Nếu là “IACCEPT”: Server sẽ gửi lại cho Client yêu cầu tín hiệu “ACCEPTED”.

+ Client (nhận):

Client nhận được tín hiệu “WANT TO CHAT”, có thể đồng ý hoặc từ chối:

- Nếu đồng ý chat: gửi về cho Server tín hiệu “IACCEPT” hai bên sẽ bắt đầu chat.

- Nếu từ chối: gửi về cho Server tín hiệu “NO”.

### 2.1.3 Thoát khỏi phiên chat

Tín hiệu bên Server: EXIT

Tín hiệu bên Client: EXITCHAT\_\_

+ Client (gửi): Khi Client đóng frame chat (thoát khỏi phiên chat) sẽ gửi “EXITCHAT\_\_”, kèm theo port của Client đang được ghép đôi cùng, đến Server.

+ Server: Nhận tín hiệu “EXITCHAT\_\_”, sẽ dò theo port, tìm đến Client tương ứng và gửi tín hiệu “EXIT”.

+ Client (nhận): Nhận tín hiệu “EXIT” và đóng frame chat, thoát phiên chat

### 2.1.4 Gửi tin nhắn

Tín hiệu bên Client: TEXTINPUT

Tín hiệu bên Server: TEXT

+ Client: gửi tin nhắn theo cú pháp TEXTINPUT:port:tin nhắn. Trong đó:

- TEXTINPUT: là tín hiệu nhận biết.

- Port : port của socket của bên nhận.

- Tin nhắn: phần tin nhắn mà Client gửi.

+ Server: khi Server nhận được yêu cầu từ Client với tín hiệu đầu là “TEXTINPUT” thì sẽ gửi tin cho Client nhận tín hiệu là “TEXT” dựa vào số port.

### 2.1.5 Thoát ứng dụng

+ Client thoát ứng dụng: đóng cửa sổ ứng dụng, đồng thời gửi cho Server tín hiệu “BYEBYEBYE”.

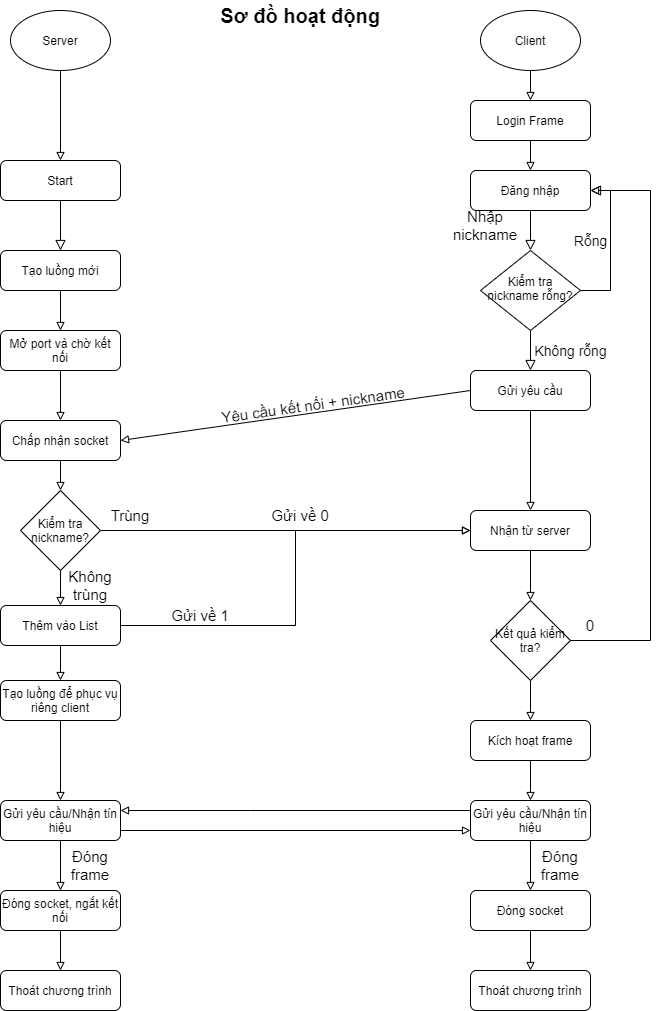
+ Server nhận được tín hiệu thì sẽ xóa Client ra hỏi list workers, sau đó cập nhật lại số lượng online cho các Client khác

### 2.1.6 Cập nhật số lượng người online

Mỗi khi có Client được Server accpet thì Server gửi tín hiệu “ONLINE” đến cho tất cả các Client có trong list workers. Client nhận được thì sẽ cập nhật số lượng người hiện có trong list.

## 2.2 Sơ đồ Flowchart

### Khởi động chương trình:



Code phía Client:

* Client kết nối, gửi đi nickname để kiểm tra.

socket = new Socket("localhost", port);

in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));

out.write(txtNickname.getText());

out.newLine();

out.flush();

* Nếu nhận về 0 thì thông báo nickname đã có. Client sẽ nhập lại nickname.
* Nếu khác 0 và thì tạo luồng để giao tiếp riêng.

if (line.equals("0"))

{

JOptionPane.showMessageDialog(null, "Nickname này đã có, hãy tạo nickname khác!!!", "Thông báo", JOptionPane.INFORMATION\_MESSAGE);

return;

} else {

//nếu khác 0 và thì tạo luồng để giao tiếp riêng

if (!line.contains("WAIT")) {

t = new Thread(LoginFrame.this);

t.start();

}

}

Code phía Server:

* Nhận nickname từ phía Client. Sau đó kiểm tra trong mapSocket (kiểm tra xem nickname có trùng hay không), checkMODE (kiểm tra tín hiệu đầu vào là gì).

while (true) {

socket = server.accept();

in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));

String message = in.readLine();

String mode = "";

String name = "";

If ((mapSocket.containsKey(message)) && (checkMODE(mode) == false)) {

out.write("0"); //nếu nickname đã tồn tại

out.newLine();

out.flush();

}

else {

if (!mapSocket.containsKey(message) && (checkMODE(mode) == false)) {

String temp = "1"; //nếu nickname chưa tồn tại

out.write(temp + “\n”);

out.flush();

mapSocket.putIfAbsent(message, socket);

listName.add(message);

Worker2 client = new Worker2(socket, message);

workers.add(client); //thêm vào list workers

for (Worker2 w : workers) { //cập nhật số lượng online

w.out.write("ONLINE:" + workers.size());

w.out.newLine();

w.out.flush();

}

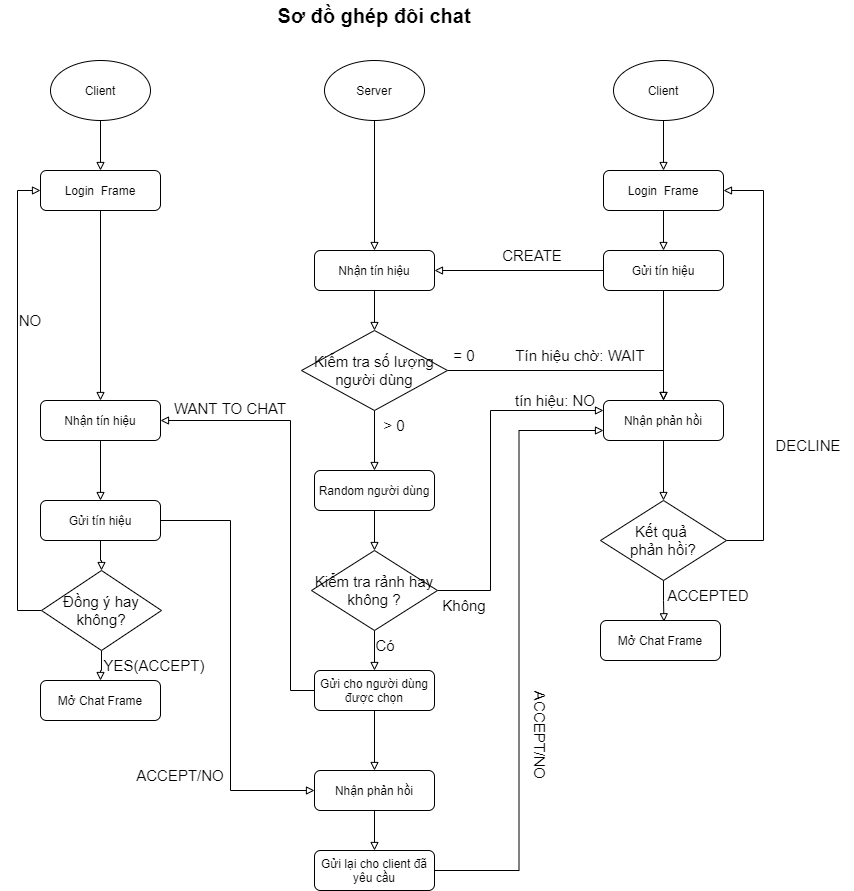
Thread t = new Thread(client); //tạo luồng riêng để phục vụ

t.start();

}

}

### Ghép đôi chat



Code phía Client gửi: gửi tín hiệu CREATE.

try {

out.write("CREATE:" + txtNickname.getText());

out.newLine();

out.flush();

} catch (Exception e) {

}

Code nhận phản hồi (đồng ý hoặc không đồng ý chat):

* Nếu là “DECLINE” thì vẫn ở trạng thái chờ.
* Nếu là “ACCEPTED” thì mở frame chat lên và bắt đầu chat.

if (receive.equals("DECLINE")) {

JOptionPane.showMessageDialog(null, "Người dùng đã từ chối!!!", "Thông báo", JOptionPane.INFORMATION\_MESSAGE);

btnCre.setEnabled(true);

}

if (receive.contains("ACCEPTED:")) {

String split[] = receive.split(":");

btnCre.setEnabled(false);

c = new ChatFrame(socket, txtNickname.getText(), split[2], Integer.parseInt(split[1]), Thread.currentThread().getName());

c.setVisible(true);

btnCre.setEnabled(false);

}

Code phía Server:

* Kiểm tra số lượng Client đang có.
* Kiểm tra Client được random có bận không?
* Gửi tín hiệu đến Client random.

if (ServerFrame.workers.size() < 2) { // nếu Client chỉ có 1 gửi lại WAIT

out.write("WAIT" + '\n');

out.flush();

System.out.println("Server write: WAIT" + " to " + myName);

} else { //nếu có nhiều hơn 2 Client thì sẽ random

Random r = new Random();

int i = r.nextInt(ServerFrame.workers.size());

Worker2 w1 = ServerFrame.workers.get(i);

while (w1.myName.equals(myName)||ServerFrame.workerTemp.contains(w1)){

i = r.nextInt(ServerFrame.workers.size());

w1 = ServerFrame.workers.get(i);

if (!w1.myName.equals(myName) && !map.containsKey(w1.myName)) {

break; //random đến khi không trùng với Client gửi yêu cầu

}

}

if (ServerFrame.workerTemp.contains(w1)){ //nếu Client được ghép đổi rồi thì gửi lại USERISCHATTING.

out.write("USERISCHATTING" + '\n');

out.flush();

} else { //nếu ngoài hai trường hợp trên.

ServerFrame.workerTemp.add(w1);

w1 = ServerFrame.workers.get(i); // random ra người chat là B

w1.out.write(myName + " WANT TO CHAT:" + socket.getPort() + ":" + myName + '\n'); //gửi đến B là A WANT TO CHAT

w1.out.flush();

map.putIfAbsent(myName, socket); //thêm tạm vào map

map.putIfAbsent(w1.myName, w1.socket);

}

}

Bên Client nhận:

* Nhận tín hiệu “WANT TO CHAT”
* Nếu đồng ý thì gửi “IACCEPT”, đồng thời mở frame chat và bắt đầu chat.
* Nếu không thì gửi lại “NO”.

if (receive.contains("WANT TO CHAT")) {

String temp[] = receive.split(":");

temp[0] = temp[0].replaceAll("WANT TO CHAT", "muốn trò chuyện với bạn.");

int choose = JOptionPane.showOptionDialog(null, temp[0], "Thông báo",

JOptionPane.YES\_NO\_OPTION, JOptionPane.INFORMATION\_MESSAGE, null, null, null);

if (choose == JOptionPane.YES\_OPTION) { //đồng ý

out.write("IACCEPT:" + temp[1] + ":" + temp[2]);

out.newLine();

out.flush(); c = new ChatFrame(socket, txtNickname.getText(), temp[2], Integer.parseInt(temp[1]), Thread.currentThread().getName());

c.setVisible(true);

btnCre.setEnabled(false);

}

if (choose == JOptionPane.NO\_OPTION) { //từ chối

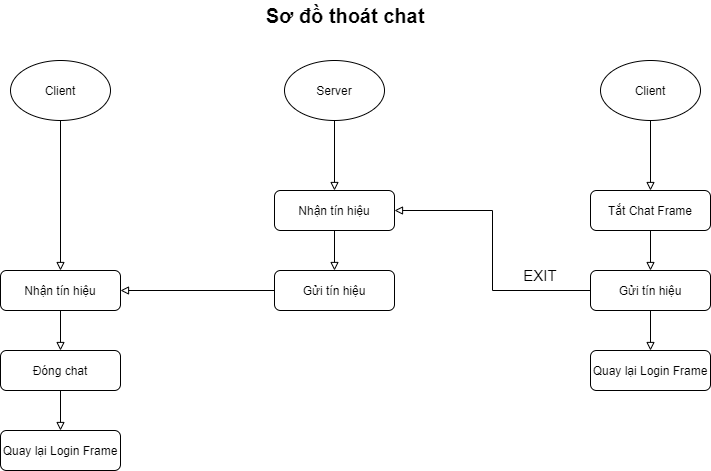
out.write("NO:" + temp[1] + ":" + temp[2] + “\n”);

out.flush();

}

}

### Thoát khỏi phiên chat



Khi 1 trong 2 người dùng thoát phiên chat:

private void formWindowClosing(java.awt.event.WindowEvent evt) {

// TODO add your handling code here:

try {

int choose = JOptionPane.showConfirmDialog(null, "Bạn chắc chắc muốn thoát?", "Thông báo", JOptionPane.YES\_NO\_OPTION);

if (choose == JOptionPane.YES\_OPTION) {

this.dispose();

System.out.println("port partner: " + portOfPartner);

out.write("EXITCHAT\_\_:" + portOfPartner);

out.newLine();

out.flush();

}

} catch (Exception e) {

}

}

Server nhận và gửi:

* Xóa Client khỏi list workerTemp (đây chỉ là list tạm thời, chứa Client chat với nhau, nếu Client thoát khỏi ứng dụng thì xóa hoàn toàn Client đó khỏi list worker).
* Sau đó gửi tín hiệu “EXIT” đến Client còn lại.

if (input.contains("EXITCHAT\_\_:")) {

String split[] = input.split(":");

for (Worker2 worker : ServerFrame.workers) {//xóa Client khỏi li

if (worker.socket.getPort() == socket.getPort()) {

ServerFrame.workerTemp.remove(worker);

}

}

for (Worker2 worker : ServerFrame.workers) { //gửi EXIT đến Client còn lại

if (worker.socket.getPort() == Integer.parseInt(split[1])) {

worker.out.write("EXIT: " + myName + '\n');

worker.out.flush();

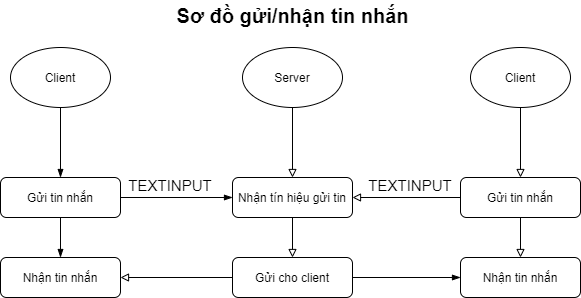
ServerFrame.workerTemp.remove(worker);

}

}

}

### Gửi tin nhắn



Client gửi tin nhắn:

* Gửi tin nhắn theo cú pháp TEXTINPUT:port:tin nhắn.
* Tin nhắn được hiển thị lên màn hình chat.

try {

out.write("TEXTINPUT:" + portOfPartner + ":" + str);

out.newLine();

out.flush();

txtArea.setText(txtArea.getText() + "You send: " + str + "\n");

txtChat.setText("");

txtChat.requestFocus();

} catch (IOException ex) {

Logger.getLogger(ChatFrame.class.getName()).log(Level.SEVERE, null, ex);

}

Server nhận tín hiệu và gửi đến cho Client tương ứng theo port:

* Cú pháp bên Server gửi Client: TEXT;From:tên Client:nội dung tin nhắn.
* Server sẽ dò theo list workers tìm ra Client có port tương ứng.

if (input.contains("TEXTINPUT:")) {

String split[] = input.split(":");

for (Worker2 worker : ServerFrame.workers) {

if (worker.socket.getPort() == Integer.parseInt(split[1])) {

worker.out.write("TEXT;From: " + myName + ": " + split[2] + '\n');

worker.out.flush(); break;

}

}

}

Client nhận tin nhắn:

* Client nhận tin nhắn và hiển thị lên màn hình chat.

if (receive.contains("TEXT;From:")) {

String split[] = receive.split(";");

if (line.contains("TEXT;From: ")) {

ChatFrame.txtArea.setText(ChatFrame.txtArea.getText() + split[1] + "\n");

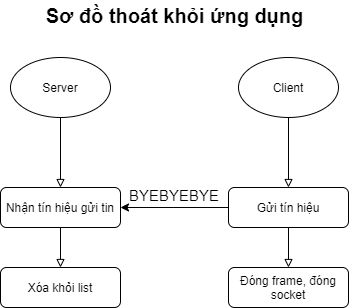
ChatFrame.txtChat.setText("");

ChatFrame.txtChat.requestFocus();

}

}

### 2.2.5 Thoát ứng dụng



Client gửi tín hiệu “BYEBYEBYE”:

private void formWindowClosing(java.awt.event.WindowEvent evt) {

// TODO add your handling code here:

try {

out.write("BYEBYEBYE");

out.newLine();

out.flush();

socket.close();

} catch (Exception e) {

}

}

Server nhận và xóa Client khỏi list workers. Sau đó gửi lại số lượng Client online để cập nhập lại số lượng.

if (input.equals("BYEBYEBYE")) {

ServerFrame.listName.remove(myName);

ServerFrame.mapSocket.remove(myName);

//Xóa client khỏi tất cả các list

for (Worker2 w : ServerFrame.workers) {

if (w.myName.equals(myName)) {

ServerFrame.workers.remove(w);

ServerFrame.workerTemp.remove(w);

break;

}

}

//Gửi lại danh sách online cho các Client còn hoạt động để cập nhật

for (Worker2 w : ServerFrame.workers) {

w.out.write("ONLINE:" + ServerFrame.workers.size());

w.out.newLine();

w.out.flush();

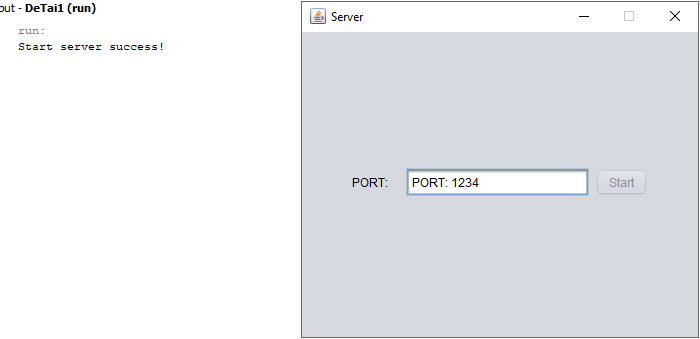
}

break;

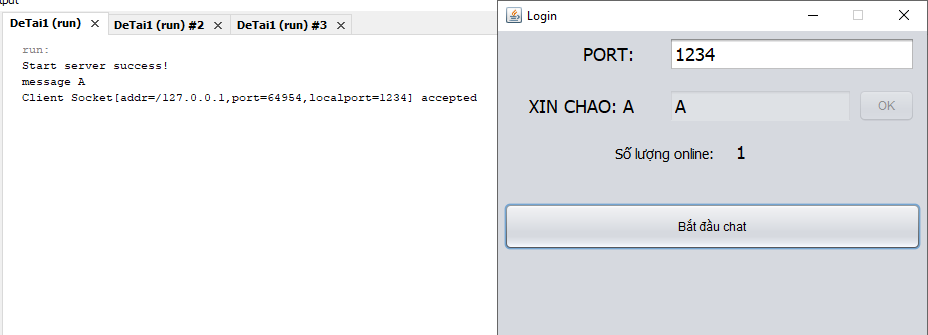
}

# Phần III: Kết quả thực thi

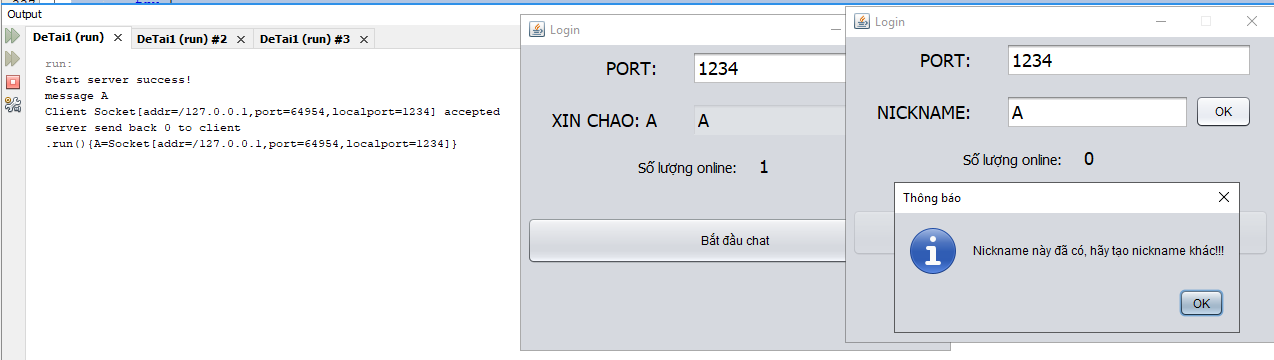
Start server.



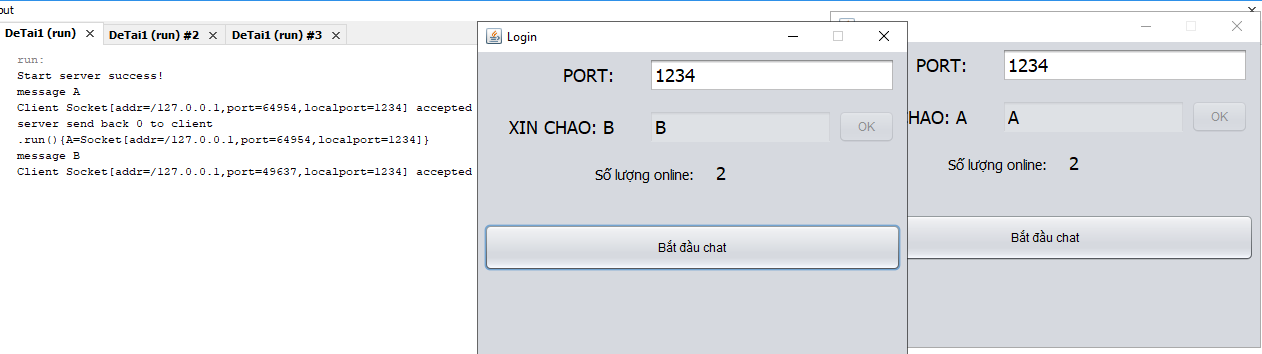
Client A kết nối.



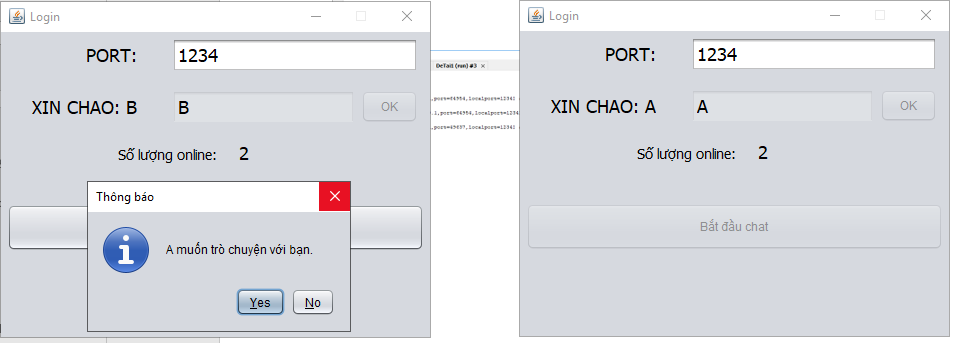
Client B kết nối, nếu trùng nickname.



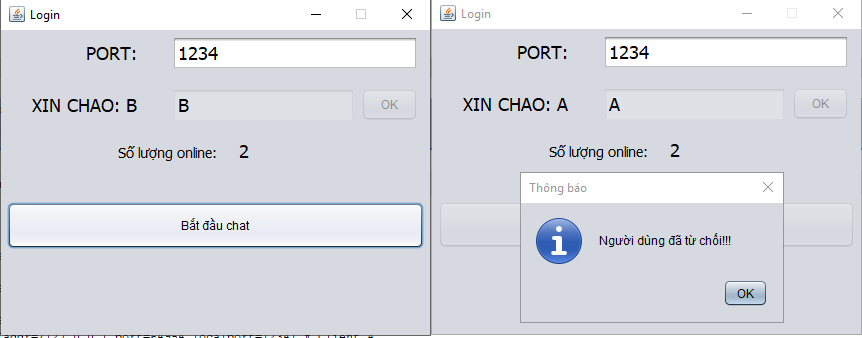
Nếu không trùng thì kết nối thành công.



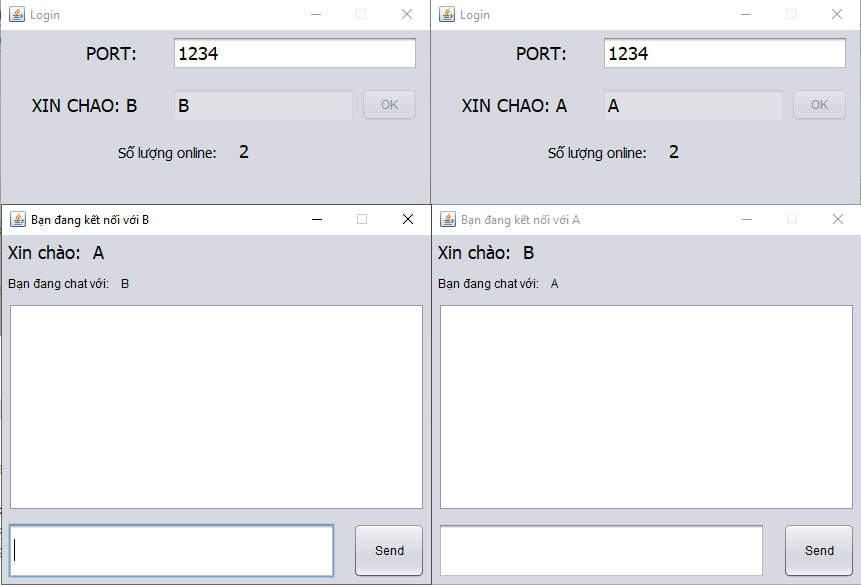
A gửi yêu cầu chat đến B.

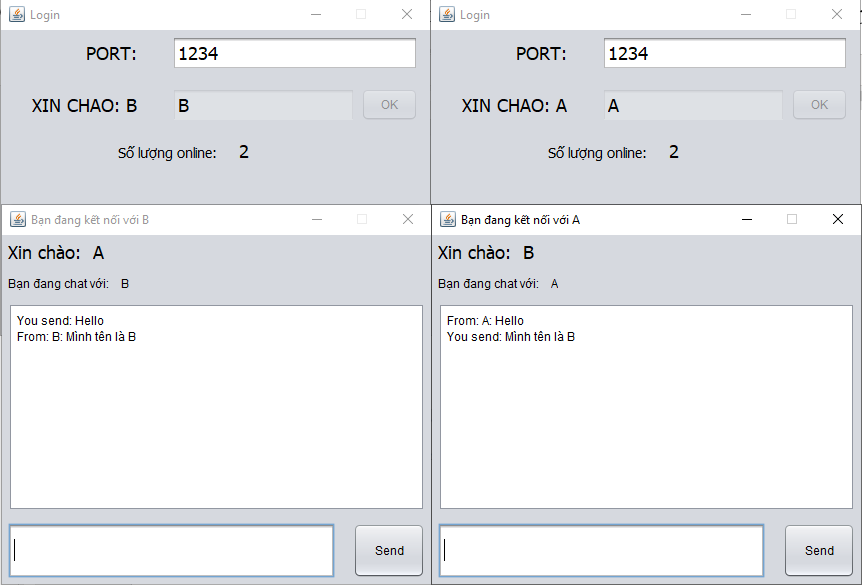


Nếu B từ chối.

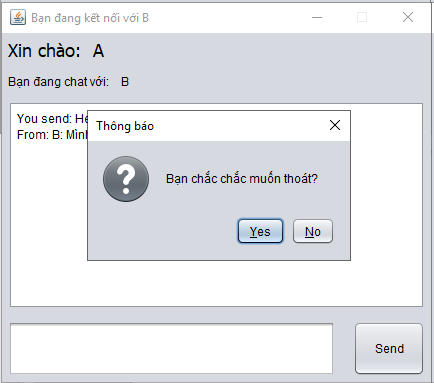


Nếu B đồng ý -> cả hai bên tiến hành chat.

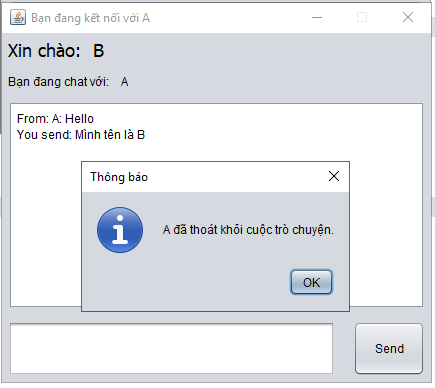




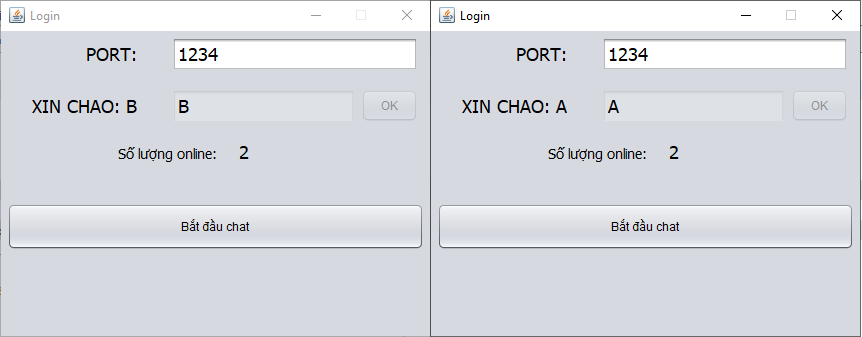
Khi 1 Client thoát phiên chat.



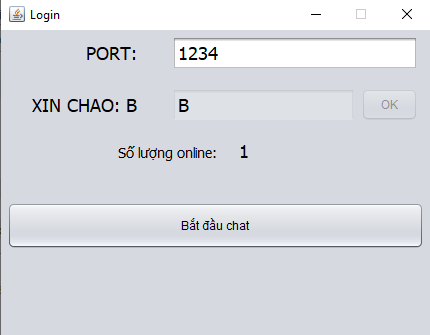
Bên còn lại cũng nhận được thông báo.



Cả hai trở lại trạng thái chờ



Khi có một Client thoát, sẽ cập nhật lại số lượng online.



# Lời cảm ơn:

Để hoàn thành tiểu luận này, em xin gửi lời cảm ơn chân thành đến:

Xin cảm ơn giảng viên bộ môn - Thầy Nguyễn Võ Lam Giang đã giảng dạy tận tình, chi tiết để em có đủ kiến thức và vận dụng chúng vào bài tiểu luận này.

Do thời gian có hạn, trong bài tiểu luận chắc chắn sẽ không tránh khỏi những thiếu sót. Rất mong nhận được sự nhận xét, ý kiến đóng góp, phê bình từ phía các thầy (cô) phát vấn để bài tiểu luận được hoàn thiện hơn.

Lời cuối cùng, em xin kính chúc các thầy (cô) nhiều sức khỏe, thành công và hạnh phúc.

# Nhận xét của giảng viên:

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

# Tham khảo:

[1] <https://www.youtube.com/watch?v=NmGGV5w9acI>, java3 Java Swing Chương trình chat client server sử dụng socket Lab7 P2, tác giả: Nam Nguyen, ngày 5 tháng 8, 2020.

[2] <https://wiki.matbao.net/socket-la-gi-khai-niem-can-biet-ve-giao-thuc-tcp-ip-va-udp/#:~:text=Datagram%20Socket%20hay%20c%C3%B2n%20g%E1%BB%8Di,%E2%80%9C>, Socket là gì? Khái niệm cần biết về giao thức TCP/IP và UDP, Mắt Bão, ngày 27 tháng 12 năm 2021.