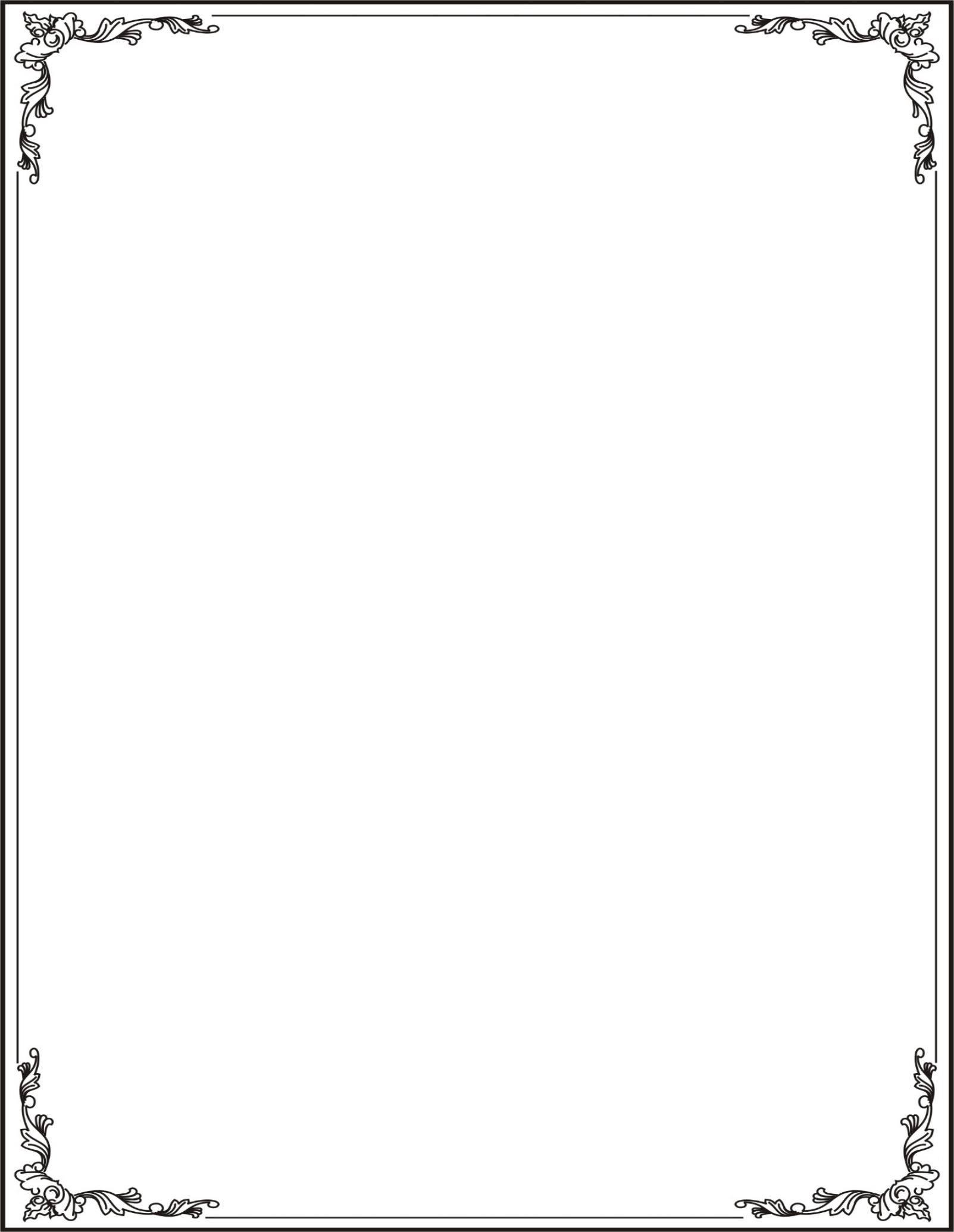
****

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA TOÁN – CƠ – TIN HỌC**

--------------\*\*\*\*---------------



**BÁO CÁO THỰC TẬP**

*Môn: Đồ án phần mềm*

LẬP TRÌNH CHAT ROOM

KHÔNG GIAO DIỆN

**Người hướng dẫn: Nguyễn Phú Tuấn**

**Sinh viên thực hiện: Vũ Thế Huy MSV: 16001796**

**Hà Nội, tháng 8 năm 2018**

**Lời cảm ơn**

*Đây là lần đầu tiên chúng em đi thực tập, trong lòng rất nhiều băn khoăn lo lắng. Có người nói đi thực tập dễ lắm toàn chơi thôi, có người thì khuyên cố gắng hết sức không thì sẽ không theo kịp sự hướng dẫn của các anh chị, nhưng các anh chị trong công ty đã phá tan mọi sự lo lắng của chúng em. Mọi người trong công ty rất vui vẻ và hòa đồng, đặc biệt là sự hướng dẫn nhiệt tình của anh Tuấn. Trong quá trình thực tập, dù bận nhưng anh vẫn đưa ra kế hoạch để hướng dẫn bọn em từng phần nhỏ để bọn em dễ hiểu và dễ dàng áp dụng vào trong bài làm cũng như trong bản thân.*

*Báo cáo này kết quả của quá trình gần 2 tháng thực tập hè tại công ty CP Trung tâm Nghiên cứu ANLAB, đây cũng là khoảng thời gian giúp em có cái nhìn mới về ngành mình đang theo đuổi. Nó giúp chúng em có thêm các kĩ năng mềm khi đi phỏng vấn, khả năng thuyết trình trước đám đông, cách trình bày ý kiến của mình một cách logic và dễ hiểu. Về mặt chuyển môn, 2 tháng thực tập là khoảng thời gian em tiếp xúc với ngôn ngữ lập trình mới, giúp cho em hiểu về cách hoạt động của một chương trình chat đơn giản. Có được những thành quả trên, ngoài sự lỗ nực và cố gắng của bản thân, sự chỉ bảo nhiệt tính của các anh chị trong công ty ANLAB, thì không thể kể đến sự giúp đỡ nhiệt tính của các thầy cô. Em chân thành gửi lời cảm ơn chân thành, sâu sắc tới thầy cô vì đã giới thiệu em đến với ANLAB, đã tạo điều kiện cho em có môi trường học tập cũng như làm việc hiệu quả nhất.*

*Em xin chân thành cảm ơn!*

MỤC LỤC

[**I. Giới thiệu chung** 4](#_Toc523327313)

[**1.** **Giới thiệu về đơn vị thực tập** 4](#_Toc523327314)

[**2.** **Công việc được giao, công cụ sử dụng** 4](#_Toc523327315)

[**3.** **Các công nghệ sử dụng** 5](#_Toc523327316)

[**II.** **Nội dung thực tập chính** 6](#_Toc523327320)

[**1.** **Đề tài** 6](#_Toc523327321)

[**2.** **Cách thức thực hiện** 6](#_Toc523327323)

[**3.** **Phân tích và thiết kế bài toán** 8](#_Toc523327325)

[3.1 Bài toán 1: Viết chương trình gửi nhận dữ liệu giữa server và một client. 8](#_Toc523327326)

[3.2 Bài toán 2: Sử dụng Multithread viết chương trình kết nối từ server đến nhiều client cùng một lúc. 12](#_Toc523327330)

[3.3 Bài toán 3: Sử dụng kiến thức OOP, thêm class User xác định từng client kết nối đến server…………………………………………...……………………………...…13](#_Toc523327331)

[3.4 Bài toán nhỏ 4: Cấp quyền quản lý cho admin 14](#_Toc523327332)

[3.5 Bài toán nhỏ 5: Tạo class Session ứng với từng phiên kết nối. 15](#_Toc523327333)

[**4.** **Kết quả đạt được** 15](#_Toc523327334)

[**5.** **Phụ lục và tham khảo** 25](#_Toc523327336)

[**III.** **Nhận xét của công ty thực tập** 26](#_Toc523327337)

# **I. Giới thiệu chung**

1. **Giới thiệu về đơn vị thực tập**

Công ty Cổ phần Trung tâm Nghiên cứu ANLAB được thành lập tháng 4/2007. A.N.Lab là tên viết tắt của chữ **Advance** and **Nurture Laboratory**, thể hiện phương châm và văn hoá của công ty. Chúng tôi theo đuổi các tiến bộ khoa học kĩ thuật (**Advance**) và nuôi dưỡng ý tưởng (**Nurture**). Môi trường ở ANLAB sẽ gần giống với mô hình phòng nghiên cứu (**Laboratory) mà ở đó các thành viên tự do chia sẻ hoài bão, đốt chạy đam mê.**

1. **Công việc được giao, công cụ sử dụng**

- Công việc: dựa vào những hiểu biết C/C++ về client-server, multi-threading và socket viết một chương trình chat room và nâng cấp dần**.**

**- Công cụ được sử dụng: Visual Studio 2017.**

**Sơ bộ về Microsoft Visual Studio**

**Microsoft Visual Studio là môi trường phát triển tích hợp (IDE) tử Microsoft. Nó được sử dụng để phát triển chương trình máy tính cho Microsoft Window, cũng như các trang web, các ứng dụng web và các dịch vụ web. Visual Studio sử dụng nền tảng phát triển phần mềm của Microsoft như Windows API, Windows Form, Window Presentation Foundation, Windows Store và Microsoft Siverlight.**

**Visual Studio bao gồm trình soạn thảo mã hỗ trợ IntelliSence cũng như cái tiến mã nguồn. Trình gỡ lỗi tích hợp hoạt động cả về trình gỡ lỗi mức độ mã nguồn và gỡ lỗi mức độ máy. Công cụ tích hợp bao gồm một mẫu thiết kế các hình thức xây dựng giao thức ứng dụng, thiết kế web, thiết kế lớp và thiết kế giản đồ cơ sở dữ liệu. Nó chấp nhận các plug-in nâng cao các chức năng ở hầu hết các cấp bao gồm thêm hỗ trợ cho các hệ thống quản lí phiên bản (như Subversion) và bổ sung thêm các công cụ mới như biên tập và thiết kế trực quan cho các miền ngôn ngữ cụ thể hoặc bộ công cụ dành cho các khía cạnh trong quy trình phát triển phần mềm. Visual Studio hỗ trợ nhiều ngôn ngữ lập trình khác nhau và cho phép trình biên dịch mã và gỡ lỗi để hỗ trợ hầu như mọi ngôn ngữ lập trình.**

- Lí do sử dụng: trong bộ thư viện chuẩn (STL) do Microsoft cung cấp có một số thư viện hỗ trợ về lập trình socket rất tốt cũng như trình gỡ lỗi hoạt động hiệu quả giúp phát hiện các lỗi nhanh chóng đồng thời cũng tích hợp thêm công cụ Github quản lí source code rất hiệu quả mà không cần thông Git Bash và sử dụng command.

1. **Các công nghệ sử dụng**

* C++ là một ngôn ngữ lập trình. Đây là một dạng ngôn ngữ đa mẫu hình tự do có kiểu tĩnh và hỗ trợ lập trình thủ tục, dữ liệu trừu tượng, lập trình hướng đối tượng và lập trình đa hình, đồng thời cung cấp các công cụ để can thiệp sâu vào bộ nhớ.

Ưu điểm:

* Mang những ưu điểm của ngôn ngữ lập trình C: có tính tương thích rất cao, thường dùng cho một loại máy hoặc hệ điều hành có thể chuyển một cách dễ dàng sang các loại hệ điều hành khác, nó hoàn toàn có thể nạp nhanh và tương thích cao cho đa dạng các bộ vi xử lí.
* Dễ hiểu và dễ sử dụng.
* Truy cập đến phần tử trong mảng nhanh.
* Sử dụng hướng đối tượng tốt.
* TCP/IP là một bộ các giao thức truyền thông cài đặt các chồng giao thức mà internet và hầu hết các mạng máy tính thương mại đang chạy trên đó. Bộ giao thức này được đặt tên theo hai giao thức chính của nó là TCP (Giao thức điều khiển giao vận) và IP (Giao thức Liên mạng). Bộ giao thức có thể coi là tập hợp các tầng, mỗi tầng giải quyết một tập hợp các vấn đề có liên quan đến việc truyền dữ liệu, và cung cấp cho các giao thức tầng trên một dịch vụ được định nghĩa rõ ràng dựa trên việc sử dụng các dịch vụ của các tầng thấp hơn. Về mặt logic, các tầng trên gần với người dùng hơn và làm việc với dữ liệu trừu tượng hơn, chúng dựa vào các giao thức tầng cấp dưới để biến đổi dữ liệu thành các dạng mà cuối cùng có thể truyền đi một cách vật lý.
* Socket network là một thiết bị đầu cuối nội bộ để gửi hoặc nhận dữ liệu trong vòng một nút trên một mạng máy tính. Nó là một đại diện của điểm cuối trong phần mềm mạng và là một tài nguyên hệ thống. Socket có nhiệm vụ tương tự như vật lí kết nối nữ, thông tin liên lạc giữa hai nút thông qua một kênh được hình dung như một hình cáp với hai đầu đực cắm vào ổ cắm tại mỗi nút. Tương tự, cổng term (thuật ngữ khác cho connector nữ) được sử dụng cho các thiết bị đầu cuối bên ngoài tại nút và socket term cũng được sử dụng cho một điểm cuối nội bộ của truyền thông nội bộ (IPC). Tuy nhiên do sự tương tự bị căng thẳng vì truyền thông mạng không cần phải là một-một hoặc có một kênh truyền thông chuyên dụng.

1. **Nội dung thực tập chính**
2. **Đề tài****:** Lập trình mạng.

Bài toán: Lập trình chat room.

1. **Cách thức thực hiện**

Sau khi nhận được yêu cầu từ anh hướng dẫn, ở mức độ đơn giản thì em có thể tự tìm hiểu và tư làm nhưng ở mức độ khó hơn thì anh hướng dẫn vạch ra logic để em tự tìm hướng giải quyết. Triển khai chi tiết:

|  |  |
| --- | --- |
| ***Tuần*** | ***Công việc*** |
| Tuần 1 | * Tìm hiểu về client-server, socket * Viết một chương trình simple chat với một client một server * Tìm hiểu về thread, multithread * Nâng cấp simple chat với multithread cho phép nhiều client kết nối tới server |
| Tuần 2 | * Tìm hiểu sâu hơn về lập trình socket * Bổ sung thêm kiến thức về Network Programming, multithread |
| Tuần 3 | * Làm bài tập traing * Nâng cấp simple chat với hướng đối tượng |
| Tuần 4 | * Tìm hiểu về memory pointer C++ * Hoàn thiện nâng cấp simple chat |
| Tuần 5 | * Làm bài tập training * Nâng cấp simple chat với phân chia truy cập admin, mod, normal |
| Tuần 6 | * Làm bài tập mid term 1 * Nâng cấp simple chat với cung cấp quyền thao tác cho admin, mod và normal |
| Tuần 7 | * Hoàn thiện nâng cấp * Làm bài tập training * Viết lại server bằng cách phân chia thành các class sao cho hợp lí |
| Tuần 8 | * Hoàn thiện server. |

1. **Phân tích và thiết kế bài toán**
   1. Bài toán 1: Viết chương trình gửi nhận dữ liệu giữa server và một client.

* Cách viết server

Server được tạo nên thông qua BLAB: bind, listen, accept, begin.

1. Bind: liên kết với một cổng

Một máy tính có thể cần phải chạy một số chương trình máy chủ cùng một lúc. Nó có thể được gửi ra các trang web, gửi email và chạy mộ máy chủ trò chuyện tất cả cùng một lúc. Để ngăn các cuộc hội thoại khác nhau bị nhầm lẫn, mỗi máy chủ sử dụng một cổng khác nhau. Một số cổng giống như một kênh trên TV. Các cổng khác nhau được sử dụng cho các dịch vụ mạng khác nhau, giống như các kênh khác nhau được sử dụng cho các nội dung khác nhau. Khi server khởi động thì nó cần thông báo cho hệ điều hành biết cổng nào sẽ sử dụng. Điều này được gọi là ràng buộc cổng.

VD: Server sử dụng cổng 1111 và để liên kết với nó cần hai thứ: bộ mô tả socket và tên socket.

#include <WS2tcpip.h>

### #pragma comment (lib, "ws2\_32.lib")

…..

sockaddr\_in serAddr;

serAddr.sin\_family = AF\_INET;

serAddr.sin\_port = htons(2203);

serAddr.sin\_addr.S\_un.S\_addr = INADDR\_ANY;

int bin = bind(server, (SOCKADDR \*)&serAddr, sizeof(serAddr));

if (bin == SOCKET\_ERROR)

std::cout << "Can't bind to socket." << std::endl;

1. Nghe

Sau khi mở cổng, việc tiếp theo của server sẽ là yêu cầu kết nối của client. Nếu server trở nên phổ biến, có thể server sẽ nhận được rất nhiều client kết nối với nó cùng một lúc. Listen() cho hệ điều hành biết bạn muốn hàng đợi bao nhiêu:

int lis = listen(server, 10);

if (lis == SOCKET\_ERROR) std::cout << "Can't listen." << std::endl;

Gọi listen() với độ dài hàng đợi là 10 có nghĩa là tối đa 10 máy khách có thể thử kết nối với server cùng một lúc. Họ sẽ không được trả lời ngay lập tức, nhưng họ sẽ có thể chờ đợi. Khách hàng thứ 11 sẽ được thông báo rằng server quá bận.

1. Chấp nhận kết nối

Khi đã ràng buộc một cổng và thiết lập hàng đợi, chỉ cần chờ đợi server dành phần lớn thời gian chờ đợi client kết nối. Server chấp nhận kết nối của client và sau đó nó trả về một bộ mô tả socket thứ hai mà bạn có thể sử dụng để giao tiếp.

#### sockaddr\_in cli;

#### int cliAddr = sizeof(cli);

#### SOCKET client = accept(server, (SOCKADDR \*)&cli, &cliAddr);

#### if (client == SOCKET\_ERROR) {

std::cout << " Can't open secondary socket!" << std::endl;

}

1. Bắt đầu giao tiếp

Cho đến nay, tất cả các luồng dữ liệu đều giống nhau. Cho dù kết nối với file hay Input/Output chuẩn, có thể sử dụng các hàm như fprintf() và fscanf() để nói chuyện với chúng. Nhưng socket hơi khác một chút. Một socketcó hai phương thức có thể được sử dụng cho đầu vào và đầu ra. Điều đó có nghĩa là nó cần các chức năng khác nhau để nói chuyện với nó.

* Nếu muốn xuất dữ liệu trên ổ cắm, không thể sử dụng fprintf(). Thay vào đó, bạn sử dụng một hàm gọi là send():

char msg[] = "Hello world!";

if (send(client, msg, strlen(msg), 0) == SOCKET\_ERROR)

std::cout << "Can't send mess!"<<std::endl;

* Cũng tương tự hàm send() đặc biệt để ghi dữ liệu, cũng có một hàm recv() để đọc dữ liệu.

int bytesRec = recv(client, msg, sizeof(int), 0);

if (bytesRec == SOCKET\_ERROR)

{

std::cout << "Error recv!" << bytesRec << endl;

break;  
}

* Cách viết client

Client và server giao tiếp bằng cách sử dụng socket. Khi client biết địa chỉ và cổng của server, nó có thể tạo một client socket. Client socket và server socket được tạo theo cùng một cách:

SOCKET client = socket(AF\_INET, SOCK\_STREAM, 0);

Nhưng cách mà mỗi bên sử dụng socket có một chút khác nhau. Như đã biết các server sử dụng trình tự BLAB: bind, listen, accept, begin. Một server dành phần lớn thời gian chờ một kết nối mới từ một client. Cho đến khi một client kết nối, một server thực sự không thể làm bất cứ điều gì. Client thì không như vậy, tuy nhiên chũng cũng có những điểm tương đồng. Một client có thể kết nối và bắt đầu nói chuyện với một server bất cứ khi nào nó thích. Đây là trình tự cho một client:

* Kết nối với cổng từ xa.
* Bắt đầu nói chuyện.
* Đóng kết nối.

Sự khác biệt giữa mã client và server là những gì họ làm với các socket khi chúng được tạo ra. Một server sẽ kết nối socket với một cổng cục bộ, nhưng một client sẽ kết nối socket với một cổng từ xa:

### string ipAddr = “170.0.0.1”;

sockaddr\_in cliAddr; //

cliAddr.sin\_family = AF\_INET;

cliAddr.sin\_port = htons(port);

### inet\_pton(AF\_INET, ipAddr.c\_str(), &cliAddr.sin\_addr);

int con = connect(client, (sockaddr\*)&cliAddr, sizeof(cliAddr));

if (con == SOCKET\_ERROR) {

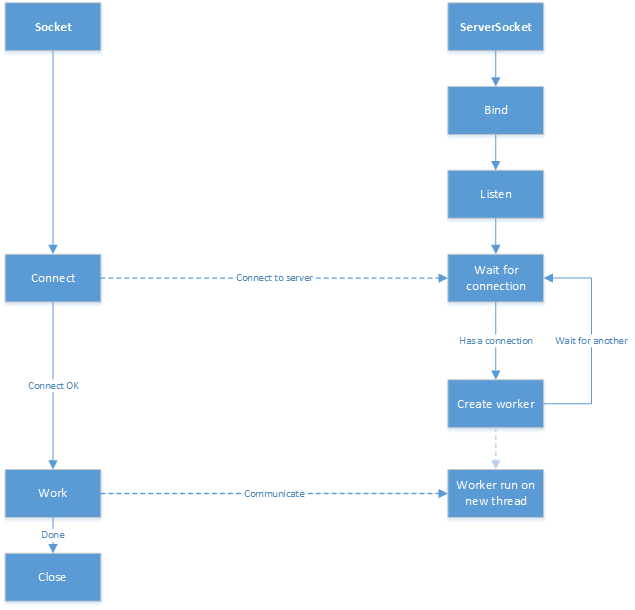
std::cout << "Connect fail!!!" << con << std::endl;

return 1;

}

Cách hoạt động của server-client như sau:

* Client sẻ sử dụng một socket để làm việc với server theo 3 bước:
* Kết nối tới server.
* Trao đổi dữ liệu với server.
* Đóng kết nối.
* Server sẽ làm việc với client theo 5 bước:
* Bind tới một endpoint(một địa chỉ IP và port) trên server.
* Bắt đầu lắng nghe kết nối.
* Đợi các kết nối.
* Tạo một worker khi có kết nối mới.



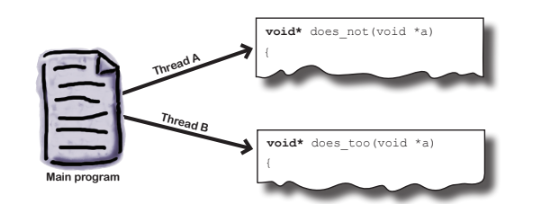
* 1. Bài toán 2: Sử dụng Multithread viết chương trình kết nối từ server đến nhiều client cùng một lúc.

Trong kiến trúc máy tính cũng như phần mềm, mọi câu lệnh hay mỗi đơn vị xử lý sẽ được thực hiện một cách độc lập và tuần tự. Tuy nhiên, điều đó khiến việc thực thi chương trình sẽ chậm hơn và nhiều khi không đáp ứng đủ yêu cầu. Đó là lý do chúng ta cần đến multithread.

|  |  |
| --- | --- |
| Multithread (đa luồng) là một mô hình lập trình được thực hiện rộng rãi, cho phép nhiều luồng tồn tại trong bối cảnh của một tiến trình. Có rất nhiều cách sử dụng multithread. Tuy nhiên, mấu chốt của bài toán này là ở việc xử ly dữ liệu giữa các luồng. | C:\Users\Thuy\AppData\Local\Microsoft\Windows\INetCache\Content.Word\1200px-Multithreaded_process.svg.png |

Với bài toán này, ta sử dụng multithread trong server để tạo các luồng ứng với mỗi client kết nối đến. Bởi thực ra khi server chấp nhận một kết nối, socket server sẽ accept và tạo ra một client socket mới. Đó là khi ta bắt đầu tạo luồng mới cho chương trình.

CreateThread(NULL,NULL,(LPTHREAD\_START\_ROUTINE)clientHandlerThread, (LPVOID)i, NULL, NULL);



Với client, nên tạo các thread. Thread có thể không cần chứa cả hàm send của client. Bởi việc gửi dữ liệu là duy nhất của client đó, nhưng nó lại có thể nhận dữ liệu từ rất nhiều client khác cùng lúc.

* 1. Bài toán 3: Sử dụng kiến thức OOP, thêm class User xác định từng client kết nối đến server.

OOP là lập trình hướng đối tượng, nó giúp cho chương trình của chúng ta đơn giản hóa độ phức tạp khi bảo trì cũng như mở rộng phần mềm, giúp tăng năng suất và còn giảm nhẹ các thao tác viết mã cho người lập trình.

Trong yêu cầu này, thêm class User với các thuộc tính name, age, gender, type. Khi server accept các client thì nó tạo ra một socket mới ứng với user mà client gửi đến. Về phía client, khi kết nối với server, đầu tiên phải gửi đến server username của nó. Sau dó server sẽ check tên đó trong list các user đã lưu. Nếu username không trùng một trong số tên của list user thì server sẽ thông báo tới client username sai hoặc username không tồn tại. Như vậy, những lần kết nối sau của user này chỉ việc lấy ra thông tin của mình trong dữ liệu đã lưu bên server theo username.

Khi user tham gia vào room thì sẽ phân chia theo type, người đăng nhập đầut tiên là amdi, còn các đăng nhập sau đó đều set là normal.

* 1. Bài toán nhỏ 4: Cấp quyền quản lý cho admin

Cũng như nhiều chương trình chat khác, các phòng chat thường sẽ có người quản lý được gọi là admin. Yêu cầu của bài toán này, admin có các quyền sau:

* + Xem thông tin của một user.
  + Ban một user khỏi chương trình chat
  + Set quyền mod cho user.
  + Tạo các filter word và hiển thị các filter word thành các kí tự \*.

User đăng nhập vào room đầu tiên là admin. Chi tiết các quyền của admin:

* Xem thông tin của một user

Sau khi admin gửi command này đến server, admin sẽ phải gửi thêm một username. Sau đó server tìm và lấy ra các thông tin của user đó. Command này chỉ đơn giản là lưu thông tin đã lấy vào một chuỗi rồi gửi cho admin.

* Ban một user khỏi chương trình chat

Command này, admin gửi đến server một username. Username đó được lưu vào trong danh sách kickout. Những username nằm trong danh sách đó không nhắn cũng như nhận tin nhắn được.

* Tạo các filter word và hiển thị các filter word thành các kí tự \*

Sẽ mất thời gian hơn khá nhiều trong việc xử lý các tin nhắn của server nếu bạn phải check từng tin nhắn để tìm kiếm một số từ. Tuy nhiên, để phòng chat của bạn trở nên văn minh thì đó là điều cần thiết. Các từ cần xử lý sẽ là các từ mà admin yêu cầu. Sau khi admin gửi, từ đó sẽ được lưu vào file. Để thực hiện command này, ta phải tạo một bộ lọc từ. Khi server nhận tin nhắn từ các client, bộ lọc sẽ tìm các từ có trong file. Sau đó, bộ đệm lưu tin nhắn sẽ được thay đổi trước khi gửi đi các client khác bằng cách thay mỗi kí tự của từ thành các kí tự \*.

* Set quyền mod cho user.

User được admin set quyền mod có một số quyền nhưng sẽ không đầy như admin.

Ngoài ra các user trong room có thể gửi tin nhắn riêng cho bất kì ai và tin nhắn all.

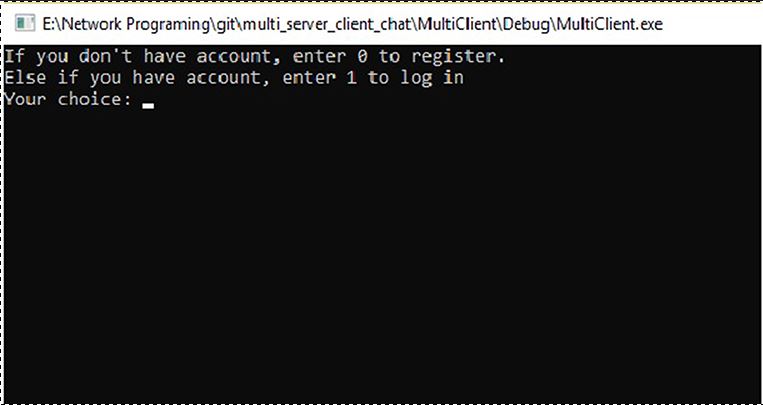
* 1. Bài toán nhỏ 5: Tạo class Session ứng với từng phiên kết nối.

Thật khó để một chương trình dài như vậy mà server và client chỉ là tập hợp của các hàm đơn lẻ. Bây giờ thay vào đó, ta sẽ chuyển chúng thành các class và hướng đối tượng chúng để dễ dàng tra cứu và sử dụng. Các client được đặt khóa bằng một username theo như các bài toán trước. Tuy nhiên, nếu như vậy, bạn sẽ không thể kết nối với server bằng một cái tên nhưng ở nhiều phiên khác nhau được. Bởi vậy, client cần có một khóa mới ứng với từng socket. Chính vì vậy, ta phải tạo ra class Session, mỗi session là một phiên kết nối của client. Như vậy, mỗi session ứng với một socket sẽ được tạo ra khi server chấp nhận kết nối socket đó. Và mỗi client sẽ có thể kết nối với server bằng nhiều session nhưng cùng chung một user.

1. **Kết quả đạt được**

Kết quả thu được là một phần mềm chat room ở mức đơn giản chưa có giao diện (sử dụng CMD).

Màn hình đăng nhập như hình dưới đây:

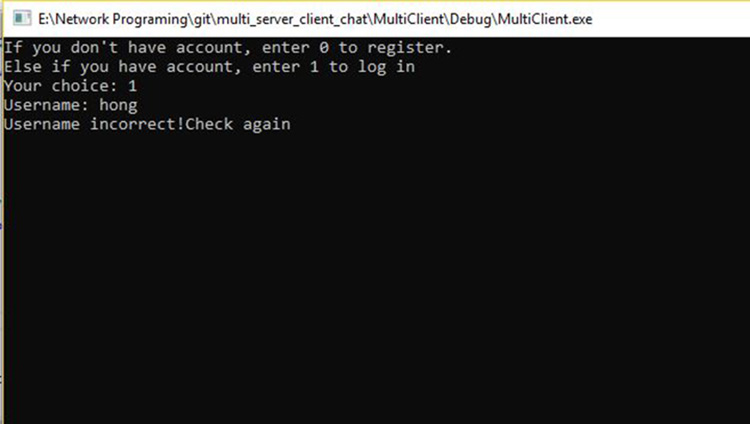


*Hình 1. Màn hình đăng nhập*

**Sau khi đăng nhập:**



*Hình 2. Màn hình sau khi đăng nhập*

Khi sai username thì sẽ nhận được thông báo như hình dưới:

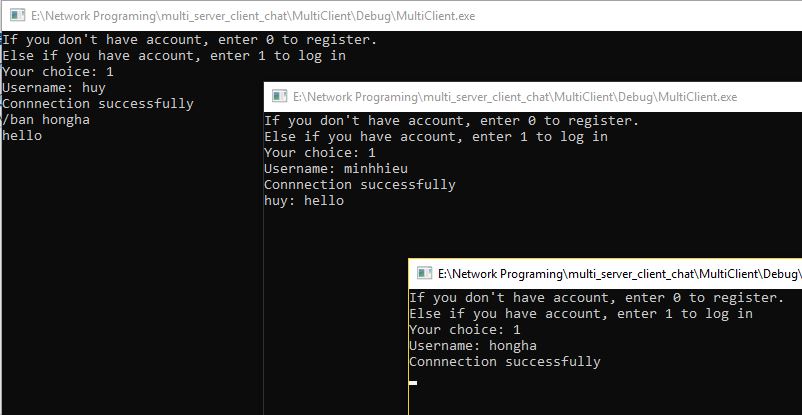
Hình 3. Thông báo khi đăng nhập sai username

Người đầu tiên đăng nhập vào là admin của phòng chat, admin sẽ có một số quyền thao tác tới các user ở trong phòng như: ban, mod, filter, xem thông tin user…

Kickout 1 user, xem thông tin 1 user, thêm và xóa từ khóa đều thông qua command.  
VD: /ban abcd,…

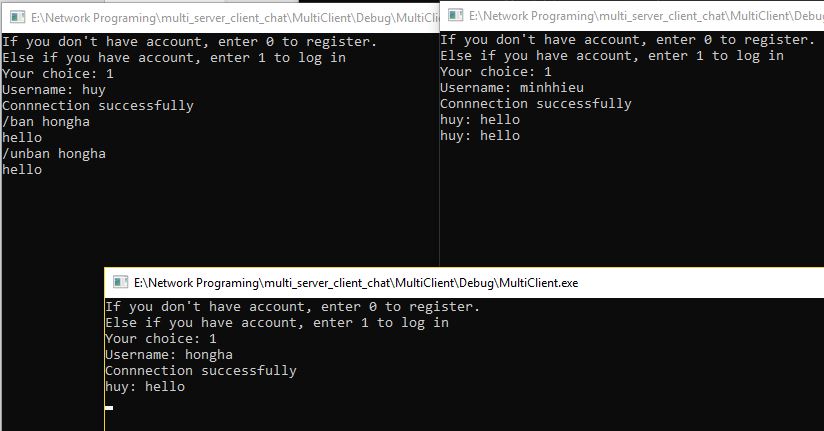
Các quyền của admin, mod, user sử dụng trong phòng chat:

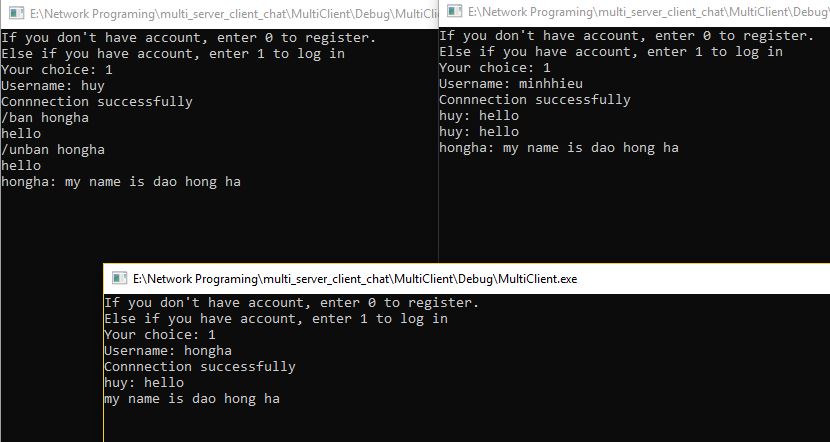
* /ban username: user bị ban không nhận được tin nhắn gửi đến và không có quyền gửi tin nhắn.



Hình 4. Mô tả user bị ban

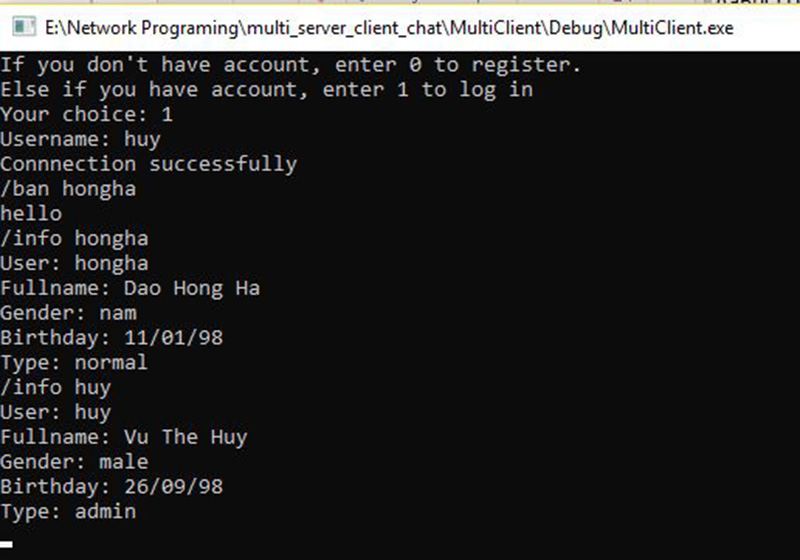
* /unban username: user đó nhận và gửi tin nhắn như bình thường.



Hình 5. User sau khi unban nhận được tin nhắn từ các user khác

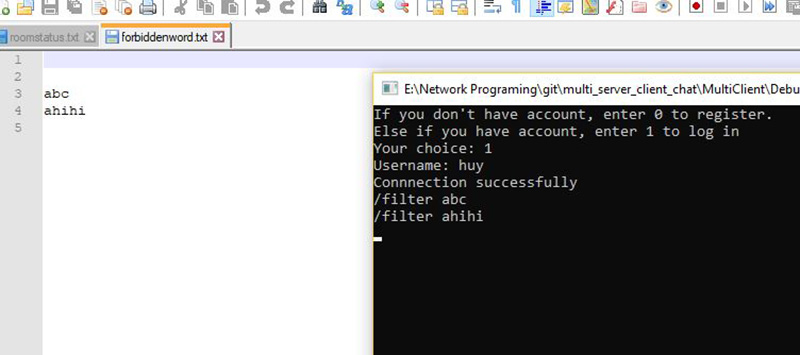
Hình 6. User sau khi ban gửi được tin nhắn

* /info username: xem thông tin của user:

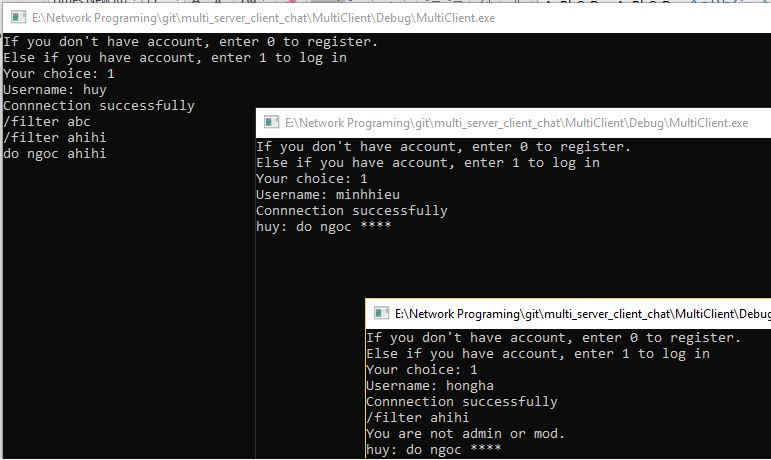


Hình 7. Thông tin chi tiết của user

* /filter word: thêm từ vào danh sách các từ xấu, được lưu xuống tệp và được nạp khi chạy server. Trong tin nhắn gửi đi có từ trong blacklist thì từ đó được chuyển thành “\*\*\*\*” trong tin nhắn gửi đến các user khác.

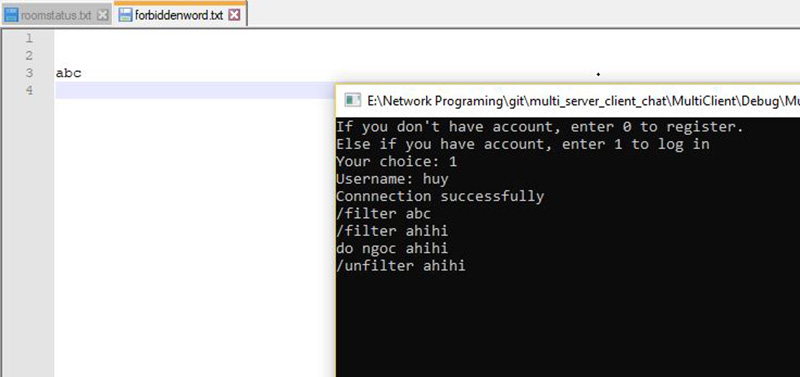


*Hình 8. Thêm từ cấm vào tệp chứa blacklist*



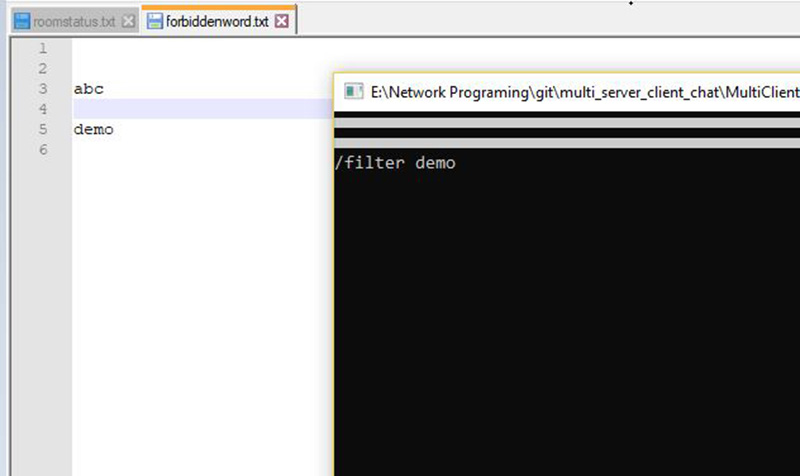
Hình 9. Bad word trong tin nhắn được thay bằng “\*\*\*\*”

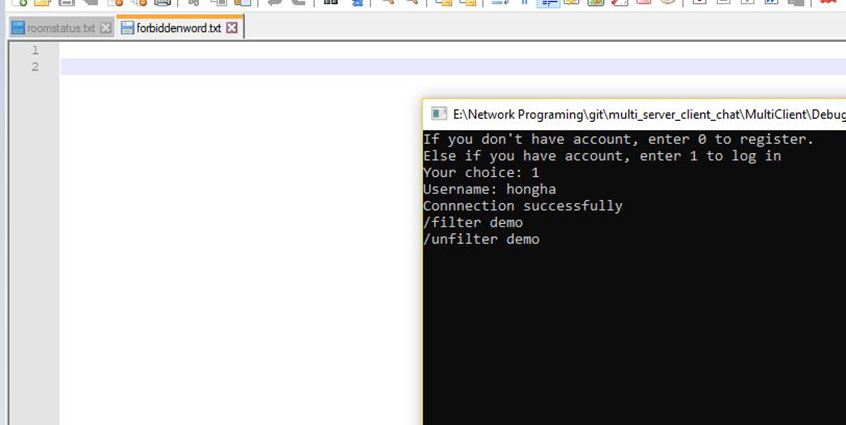
* /unfilter word: xóa word khỏi danh sách từ cấm.



Hình 10. Xóa bad word khỏi blaklist..

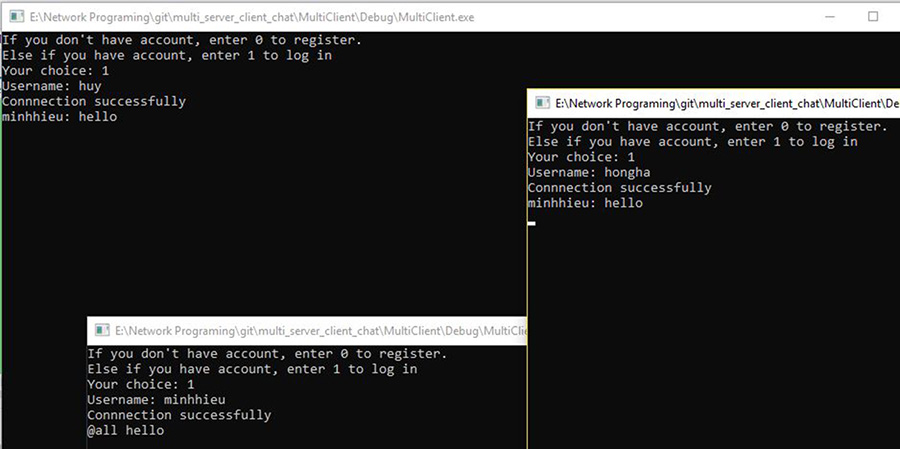
* /mod username: set quyền mod cho user đó. User có quyền mod được thêm từ cấm và xóa từ cấm.



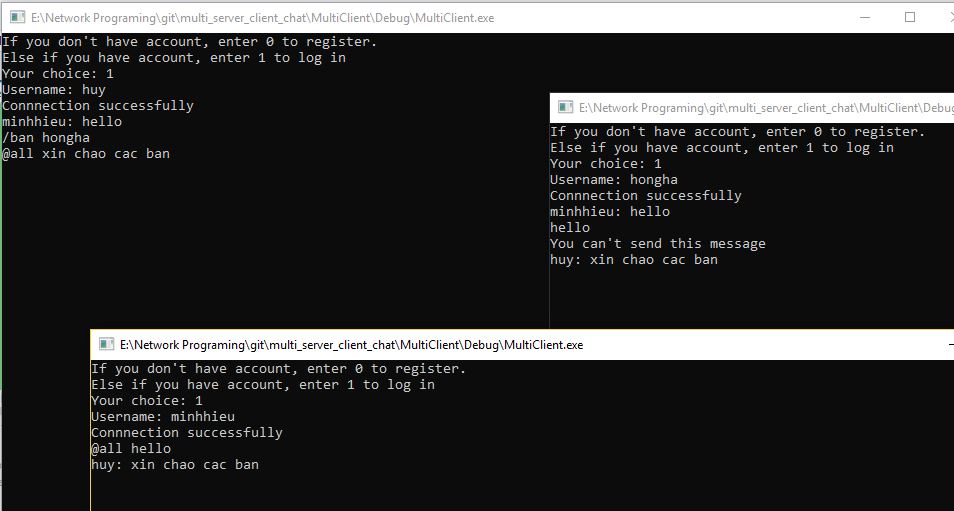
Hình 11. User được set mod thêm được bad word

*Hình 12. User được set mod xóa được bad word*

* @all: gửi tới tất cả các user trong phòng chat kể cả user bị ban, các user đều có quyền này.

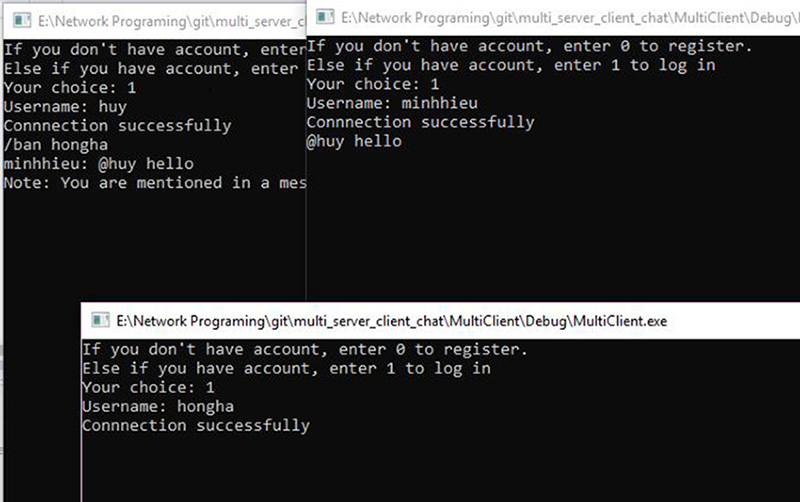


Hình 13. Tin nhắn all được gửi đến tất cả user trong phòng



Hình 14. User bị ban vẫn nhận được tin nhắn all

* @username message: tin nhắn được gửi đến user trong phòng ngoại trừ user bị ban đồng thời username được nhắc đến trong tin nhắn nhận thêm một thông báo.



Hình 15. Username nhắc đến nhận được thông báo và user bị ban không nhận được tin nhắn

Admin có đầy đủ các quyền được mô tả như các ảnh trên.

Mod có những quyền thêm và xóa từ khóa (mô tả như hình 11, hình 12).

User chỉ được gửi tin nhắn all và riêng cá nhân nào đó.

1. **Phụ lục và tham khảo**

* <https://msdn.microsoft.com/en-us/library/windows/desktop/ms738545(v=vs.85).aspx>
* <https://www.codeproject.com/Articles/13071/Programming-Windows-TCP-Sockets-in-C-for-the-Begin>
* <https://www.codeproject.com/Articles/16774/Multi-Threaded-Client-Server>
* <http://diendan.congdongcviet.com/threads/t42977::tim-hieu-ban-chat-cua-con-tro-tu-co-ban-den-nang-cao.cpp>
* <https://www.tutorialspoint.com/cplusplus/cpp_pointers.htm>
* <https://stackoverflow.com/>
* <https://gist.github.com/oleksiiBobko/43d33b3c25c03bcc9b2b>
* <https://www.youtube.com/watch?v=AjG4jcVn6QE&t=9s>
* <http://www2.cs.uregina.ca/~hilder/cs330/FYI/Head%20First%20C.pdf>
* <https://en.wikipedia.org/wiki/Network_socket>
* <https://vi.wikipedia.org/wiki/TCP/IP>

1. **Nhận xét của công ty thực tập**

* Công ty CP Trung tâm Nghiên cứu ANLAB: tầng 8 tòa nhà 18/4, số 8 Phạm Hùng, Mai Dịch, Cầu Giấy, Hà Nội.
* Người hướng dẫn: Nguyễn Phú Tuấn

Số điện thoại: 0986617948 Email: tuan.np@anlab.info

* Nhận xét về quá trình thực tập:
* **Xác nhận về công việc được giao trong báo cáo:**

* **Xác nhận về kết quả đạt được:**

* **Ý thức và thái độ làm việc trong quá trình thực tập tại công ty:**

* **Góp ý với sinh viên nhà trường để công việc có hiệu quả hơn:**

Những ý kiến khác (Tiếp nhận các sinh viên này thực tập tại công ty không,…)

* **Điểm số đánh giá sinh viên cụ thể:**

# Bằng chữ:……………………………………………… Bằng số: ……………

*Hà Nội, ngày tháng 08 năm2018*

|  |  |
| --- | --- |
| Xác nhận thực tập của công ty | Người hướng dẫn  ***(Ký và ghi rõ họ tên)*** |