

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Môn học:

Thực hành lập trình mạng



BÁO CÁO BÀI TẬP MÔN HỌC
ĐỀ TÀI: FUNGAME

Nhóm sinh viên thực hiện:

Họ tên	MSSV
1. Vũ Đức Nguyễn	20162999
2. Nguyễn Việt Hưng	20166256
3. Phùng Thị Trang	20165645

Giảng viên hướng dẫn: TS. Đặng Tuấn Linh

Hà Nội, tháng 12 năm 2019

Mục lục

Lời mở đầu.....	2
1. Giới thiệu đề tài.....	3
2. Phân tích thiết kế	4
3. Chức năng	5
4. Công nghệ sử dụng.....	13
5. Đánh giá.....	18
6. Phân công công việc	20
7. Kết luận.....	21
8. Tài liệu tham khảo.....	22

Lời mở đầu

Lời đầu tiên, chúng em xin được chân thành cảm ơn sự giúp đỡ, giảng dạy nhiệt tình của thầy giáo Đặng Tuấn Linh. Nhờ những kiến thức trong môn học thực hành lập trình mạng mà thầy đã nhiệt tình chỉ dạy trên những giờ học trên lớp, chúng em đã hoàn thành được đề tài **Fun Game** trong khuôn khổ báo cáo môn học.

Mặc dù chúng em đã cài đặt và triển khai đầy đủ các yêu cầu đề ra, tuy nhiên những thiếu sót trong quá trình thực hiện và tổng hợp là không thể tránh khỏi, rất mong nhận được những ý kiến đóng góp, chỉ bảo từ thầy để chúng em có thể nắm vững hơn các vấn đề đặt ra của môn học.

Chúng em xin chân thành cảm ơn!

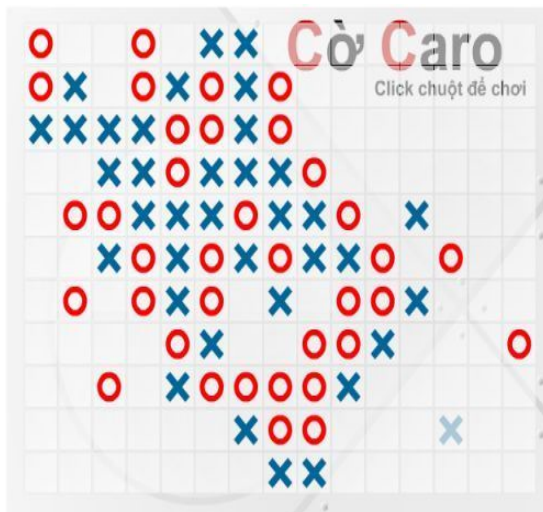
Hà Nội, tháng 12 năm 2019

1. Giới thiệu đề tài

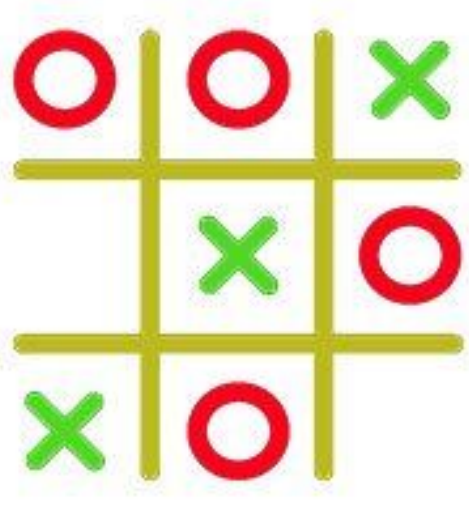
Thực trạng hiện nay, rất nhiều người mệt mỏi sau những giờ học và làm việc căng thẳng. Từ đó chúng em nảy ra ý định làm đề tài "fungame" này. Fungame tức là những trò chơi vui vẻ, giải trí, mục đích để giúp mọi người thư giãn, giảm thiểu những căng thẳng mệt mỏi, ngoài ra cũng giúp cải thiện khả năng tư duy của người chơi.

Ứng dụng "fungame" này được xây dựng dựa trên các kỹ năng và kiến thức đã được học trong bộ môn thực hành lập trình mạng. Ứng dụng này cho phép nhiều người chơi cùng lúc, sau khi đăng nhập, người chơi có thể đánh cờ caro trực tiếp với máy hoặc có thể chơi game tictactoe (phiên bản thu nhỏ của cờ caro truyền thống với 3 hàng dọc và 3 hàng ngang), sau khi chơi xong, người chơi có thể xem lịch sử trận vừa chơi. Ngoài ra, người chơi có thể xem bảng xếp hạng, bảng xếp hạng bao gồm toàn bộ thông tin của người chơi (tài khoản, số trận thắng, thua, hòa, điểm, thứ hạng,...) cùng với đó là thứ hạng của tất cả người chơi.

Ngoài ra ứng dụng cũng đã giải quyết được bài toán "client và server bị chờ đợi mãi mãi" bằng kỹ thuật đặt giá trị timeout.



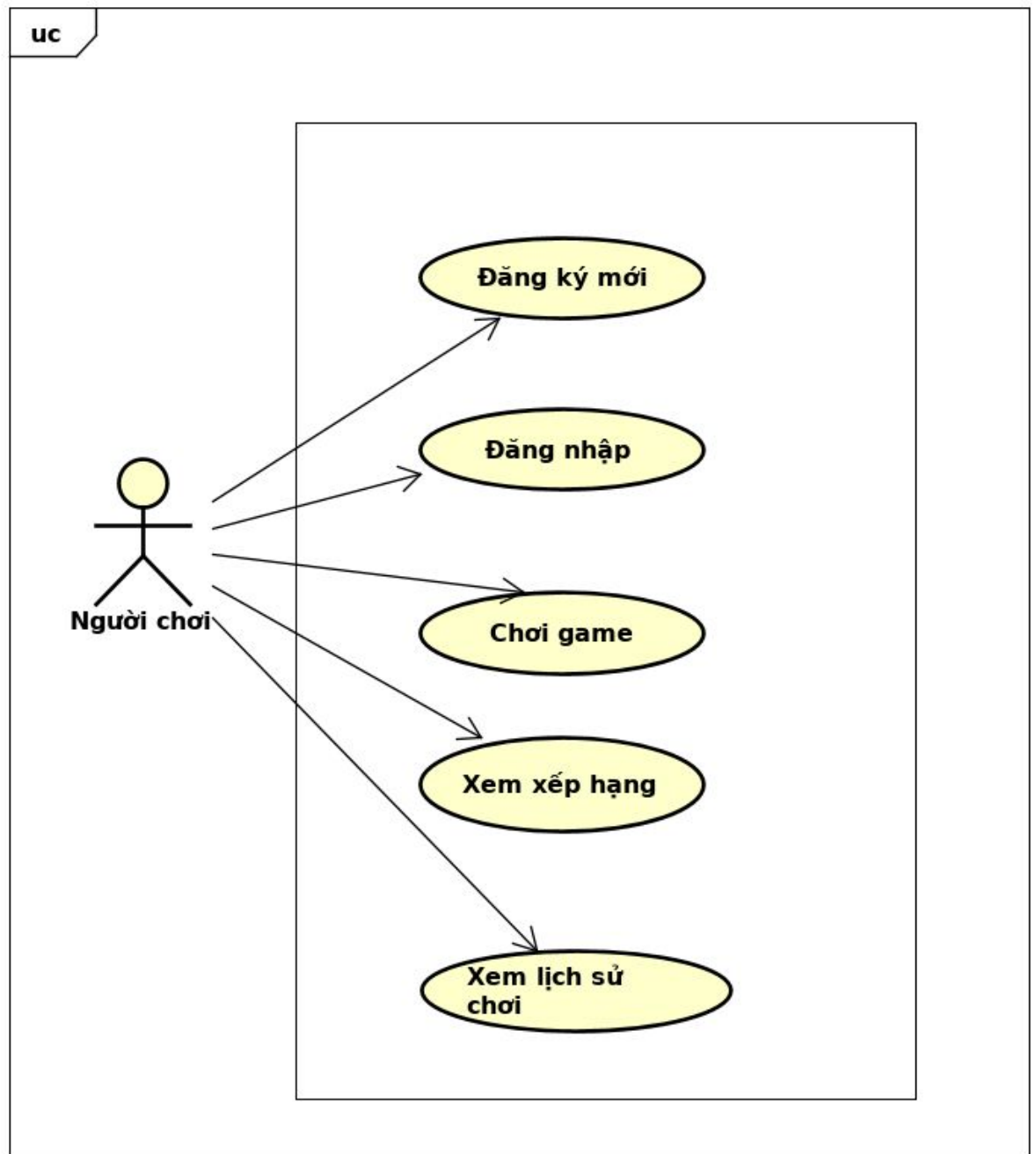
Hình 1: Trò chơi caro



Hình 2: Trò chơi Tictactoe

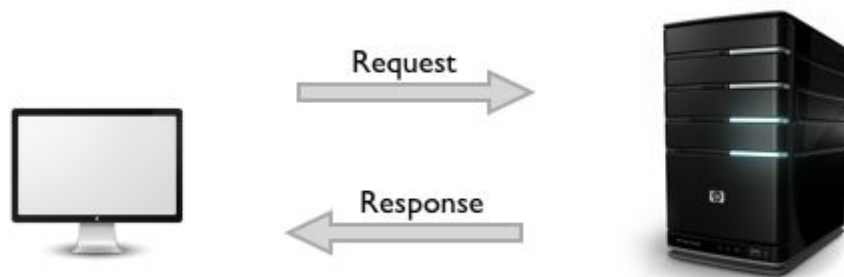
2. Phân tích thiết kế

✓ Biểu đồ Use case



Hình 3: Biểu đồ Use Case

✓ Kiến trúc sử dụng cho ứng dụng là: client/server

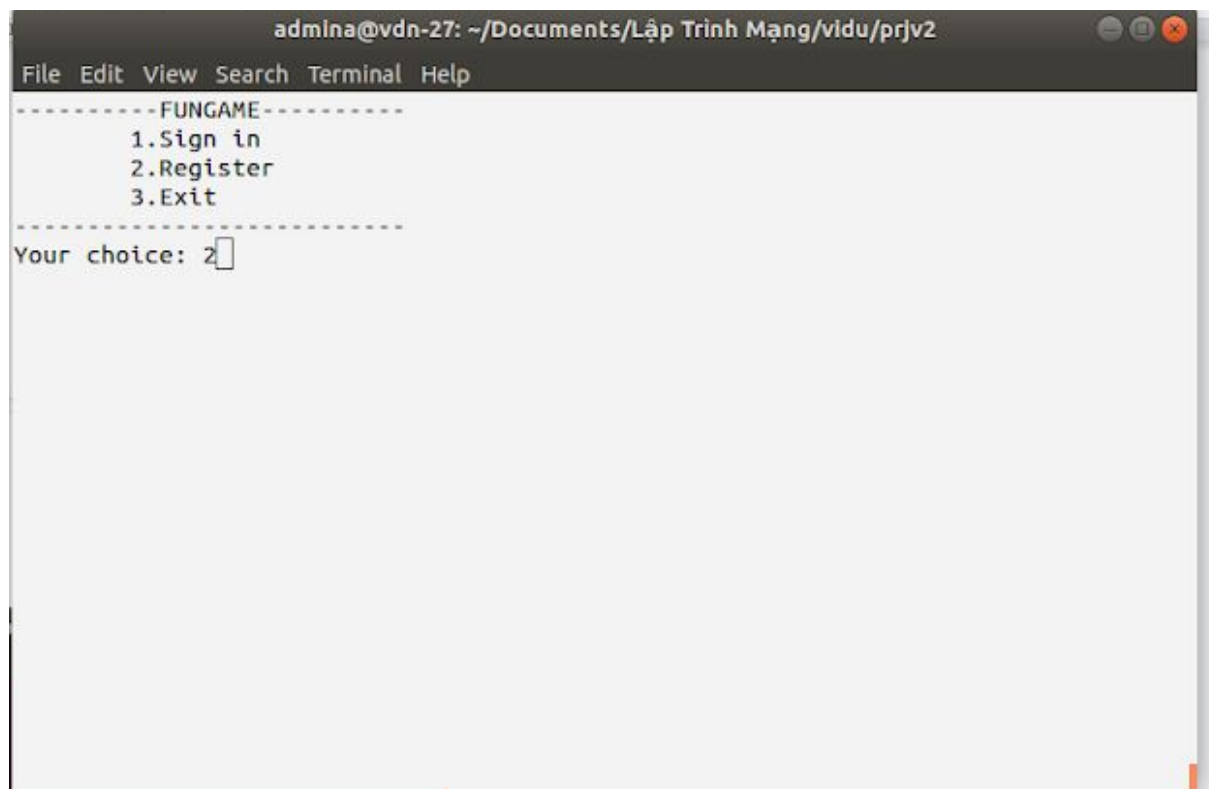


Hình 4: Kiến trúc Client/Server.

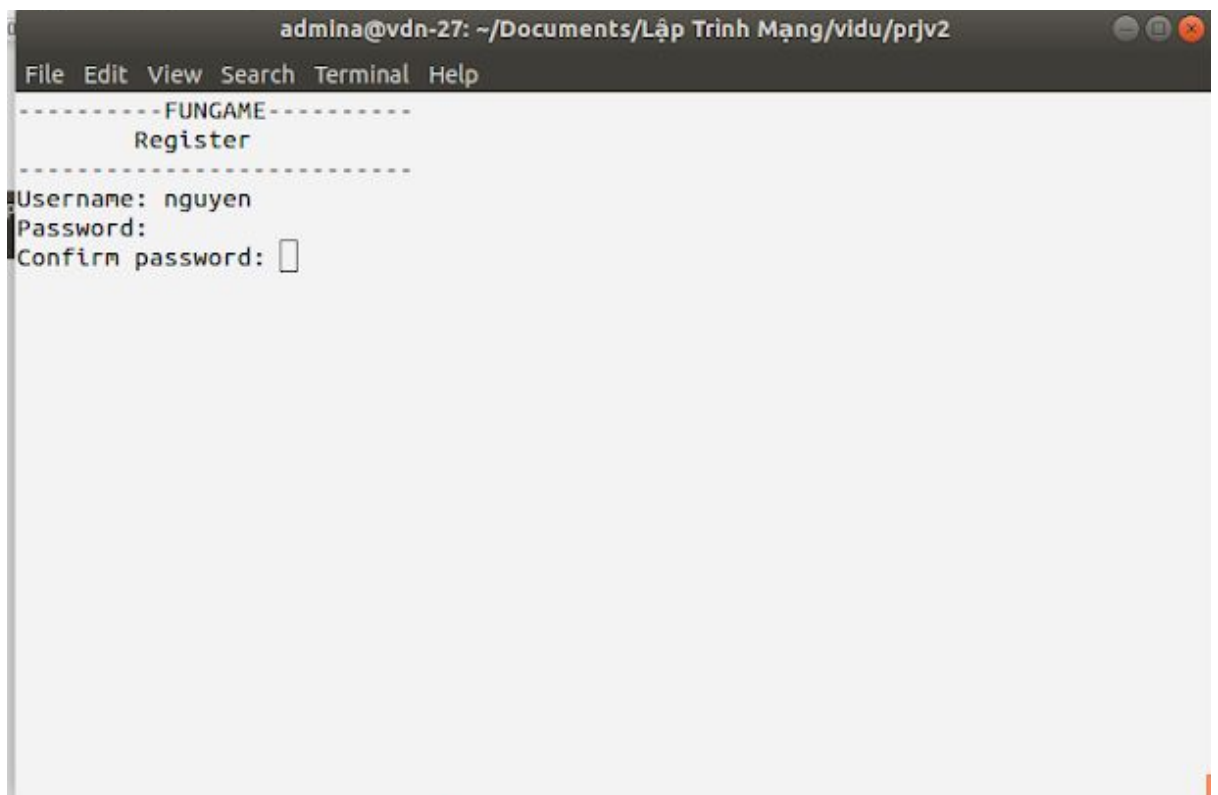
- ✓ Mô hình cấu trúc client/ server: nhiều client kết nối tới 1 server
- ✓ Chương trình được xây dựng trên kiến trúc này, client sẽ trao đổi dữ liệu của mình thông qua server. Server có nhiệm vụ quản lý thông tin đăng nhập và trao đổi thông tin với client.

3. Chức năng

3.1: Chức năng đăng ký mới



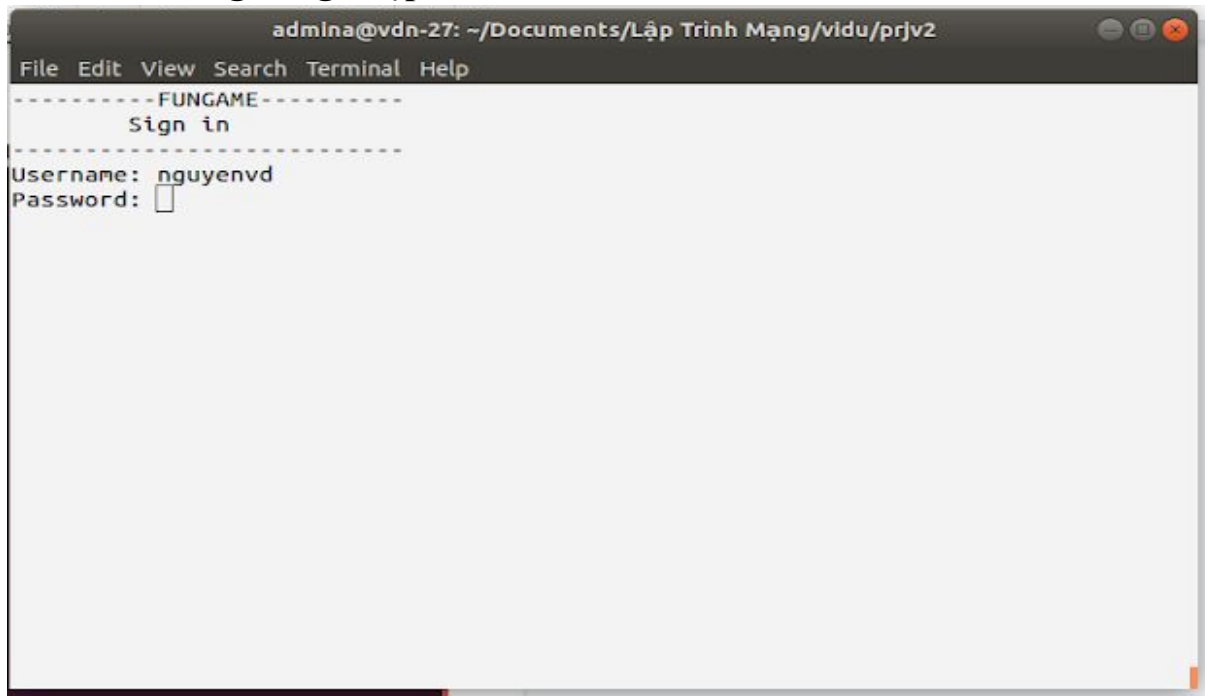
Hình 5: Giao diện chức năng trước khi chơi game



Hình 6: Giao diện đăng ký tài khoản mới

Sau khi chọn 2 để tạo tài khoản mới(Hình 2), người chơi được chuyển hướng sang cửa sổ màn hình khác(Hình 3). Tại cửa sổ mới, người chơi tạo tên tài khoản và mật khẩu, sau đó xác nhận lại mật khẩu mình đã đặt. Điều đặc biệt mà nhóm em đã làm được là khi nhập mật khẩu thì mật khẩu này sẽ không hiển thị để tạo tính bảo mật cao cho người chơi.

3.2: Chức năng đăng nhập

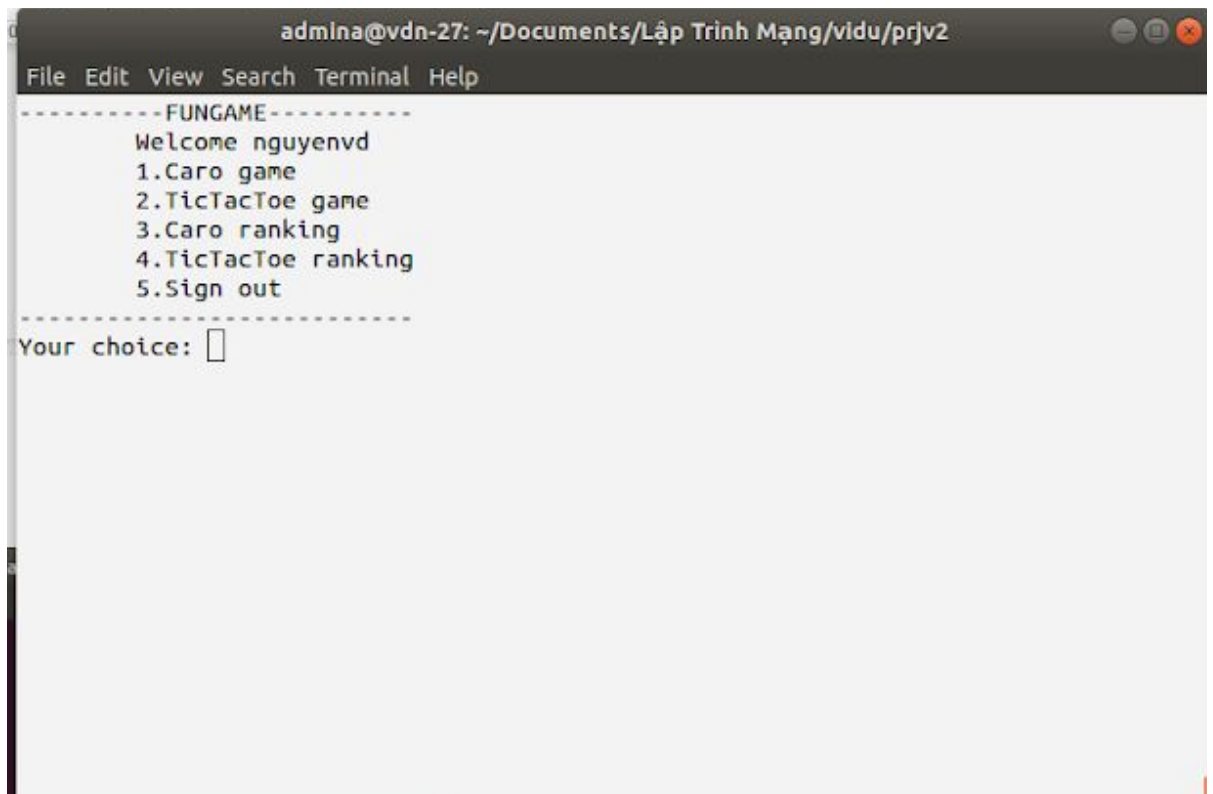


Hình 7: Giao diện đăng nhập

Bằng cách chọn 1 để đăng nhập ở hình 2, người dùng sẽ nhập tên tài khoản và mật khẩu tương ứng mà đã đăng ký trước đó để đăng nhập.

3.3: Chức năng chơi game

Sau khi đăng nhập thành công(Hình 4) hoặc đăng ký tài khoản mới thành công (Hình 3), người chơi được chuyển hướng sang cửa sổ mới(Hình 5). Tại cửa sổ này, người chơi có thể chọn các lựa chọn bên dưới theo ý mình.



Hình 8: Giao diện chính của fungame



Hình 9: Giao diện lựa chọn kích thước bảng chơi game caro

```
admina@vdm-27: ~/Documents/Lập Trình Mạng/vidu/prjv2
File Edit View Search Terminal Help
-----CARO GAME-----
Username: nguyenvd - GameID = 1577123309515685
      Press w,a,s,d to move
      Press enter to select
      Press 'q' to quit
-----
 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14
0  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
1  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
2  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
3  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
4  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
5  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
6  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
7  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
8  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
9  [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
10 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
11 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
12 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
13 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
14 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
You Win. Do you want to view log (y or n)?
Your choice: 
```

Hình 10: Giao diện sau khi kết thúc 1 lượt chơi game caro

```
admina@vdm-27: ~/Documents/Lập Trình Mạng/vidu/prjv2
File Edit View Search Terminal Help
13 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
14 [ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ][ ]
You Win. Do you want to view log (y or n)?
Your choice: y
Username: nguyenvd - SessionID: 1577123309515685
Day: 24-12-2019 - Time: 0:48:29
You: column:7 - row:7
Computer: column:6 - row:6
You: column:9 - row:5
Computer: column:9 - row:7
You: column:11 - row:7
Computer: column:9 - row:6
You: column:10 - row:6
Computer: column:12 - row:8
You: column:8 - row:4
Computer: column:7 - row:3
You: column:8 - row:6
Computer: column:10 - row:4
You: column:8 - row:7
Computer: column:6 - row:2
You: column:8 - row:5
Computer: column:8 - row:9
You: column:8 - row:8
Press 'q' to quit: 
```

Hình 11: Giao diện xem lịch sử các nước đi game caro

Sau khi đăng nhập thành công và chọn chơi game caro, người chơi được chuyển hướng đến cửa sổ mới để chơi(Hình 6). Tại cửa sổ này, người chơi tiến hành chọn kích thước cho độ rộng của lượt cờ caro muốn chơi. Sau đó người chơi tiến hành chơi theo độ rộng mình đã chọn. Sau khi chơi xong một lượt chơi, người chơi có quyền xem lại lịch sử các nước đi của mình bằng cách nhấn 'y' hoặc 'n' . Nếu nhấn 'y' thì người chơi sẽ xem được lịch sử các nước đi(Hình 8). Sau đó, người chơi nhấn 'q' để quay về giao diện chọn game để chơi(Hình 5).



```
admina@vdn-27: ~/Documents/Lập Trình Mạng/vidu/prjv2
File Edit View Search Terminal Help
-----FUNGAME-----
Username: nguyenvd
      NEW TicTacToe GAME
-----
Computer: 0, You: X
You want to Play (1)st or (2)nd: 1

 1 | 2 | 3
---+---+---
 4 | 5 | 6
---+---+---
 7 | 8 | 9

Input move ([1..9]): 
```

Hình 12: Giao diện chọn chơi game tictactoe

Sau khi đăng nhập thành công và chọn chơi game tictactoe, người chơi được chuyển hướng đến cửa sổ mới để chơi(Hình 9). Tại cửa sổ này, người chơi tiến hành chọn xem mình muốn đi trước hay không ? Sau đó người chơi tiến hành chơi. Sau khi chơi xong một lượt chơi, người chơi xem được kết quả thắng/thua/hòa của lượt chơi đó và có quyền chơi tiếp hay không bằng cách nhấn 'y' hoặc 'n'(Hình 10). Nếu nhấn 'y' thì người chơi sẽ quay lại chơi(Hình 9). Trường hợp người chơi nhấn 'n' thì sẽ quay về giao diện chọn game để chơi(Hình 5).

```
admina@vdm-27: ~/Documents/Lập Trình Mạng/vidu/prjv2
File Edit View Search Terminal Help
-----FUNGAME-----
Username: nguyenvd!
NEW TicTacToe GAME
-----
Computer: O, You: X
You play (1)st

  X | O | X
  +---+
  X | O | O
  +---+
  O | X | 9

Input move ([1..9]): 9

A draw. How droll.
Do you want to play again? (y or n)
Your choice: 
```

Hình 13: Giao diện khi kết thúc 1 lượt chơi game tictactoe

```
admina@vdm-27: ~/Documents/Lập Trình Mạng/vidu/prjv2
File Edit View Search Terminal Help
-----FUNGAME-----
Username: nguyenvd
-----Caro Ranking-----
Your information:
  Username-ID      : nguyenvd
  Number Of Wins   : 62
  Number Of Losses : 2
  Number Of Draws  : 0
  Point            : 60.0
-----
Caro Ranking
-----
TOP  ID           Win    Lose   Draw   Point
1    nguyenvd      62     2      0     60.0
2    nnn           11     2      3     10.5
3    manhvdm       5      1      0      4.0
4    bbb           2      1      1      1.5
5    aaa           1      1      1      0.5
Press 'q' to quit: 
```

Hình 14: Giao diện xem bảng xếp hạng game caro

```
admina@vdm-27: ~/Documents/Lập Trình Mạng/vdu/prjv2
File Edit View Search Terminal Help
-----FUNGAME-----
Username: nguyenvd
-----TicTacToe Ranking-----
Your information:
    Username-ID      : nguyenvd
    Number Of Wins   : 3
    Number Of Losses : 0
    Number Of Draws  : 11
    Point            : 8.5
-----
TicTacToe Ranking
-----
TOP  ID           Win    Lose   Draw   Point
1    mmm           10     0      3     11.5
2    nnn           11     2      3     10.5
3    nguyenvd      3      0     11     8.5
4    manhvdm       5      1      0     4.0
5    ttt           3      1      2     3.0
6    bbb           2      1      1     1.5
7    aaa           1      1      1     0.5
8    vuducnguyen   0      1      1    -0.5
9    c             0      1      0    -1.0
10   b             0      3      1    -2.5
Press 'q' to quit: 
```

Hình 15: Giao diện xem bảng xếp hạng game tictactoe

Sau khi đăng nhập thành công, người chơi có thể xem thành tích của mình thông qua bảng xếp hạng bằng cách chọn 3 hoặc 4. Người chơi được chuyển hướng đến cửa sổ mới (Hình 14 hoặc Hình 15) để xem thành tích chơi của mình. Tại đây, người chơi có thể biết được số trận thắng, thua, hòa, điểm số và xem mình đang đứng thứ mấy trong tổng số những người chơi. Sau khi xem xong, người chơi nhấn 'q' để ra màn hình ban đầu(Hình 8).

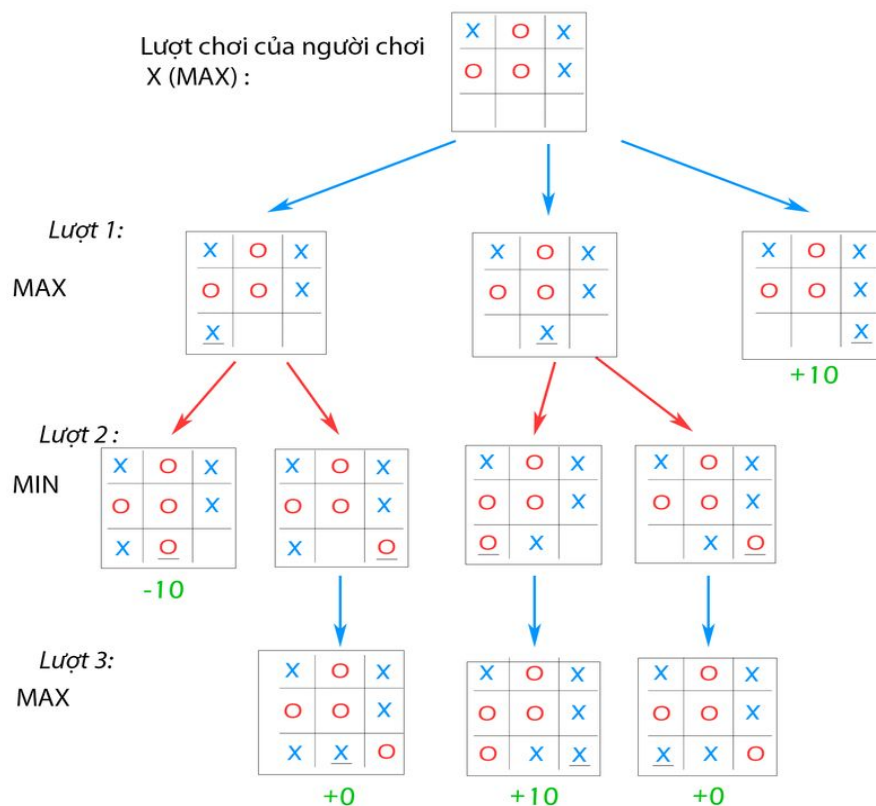
4. Công nghệ sử dụng

4.1 Thuật toán Minimax

- Các khái niệm:

- Cây trò chơi (Game tree) - là một sơ đồ hình cây thể hiện từng trạng thái, từng trường hợp của trò chơi theo từng nước đi.
- Mỗi node biểu diễn 1 trạng thái của trò chơi hiện tại trên cây trò chơi.
- Node được gọi nút lá là tại đó trò chơi kết thúc (trạng thái trò chơi lúc đó có thể thắng, thua hoặc hòa).

- Giải thuật Minimax Hai người chơi trong game được đại diện là MAX và MIN. MAX đại diện cho người chơi luôn muốn chiến thắng và cố gắng tối ưu hóa ưu thế của mình còn MIN đại diện cho người chơi cố gắng cho người MAX giành số điểm càng thấp càng tốt. Giải thuật Minimax thể hiện bằng cách định trị các Node trên cây trò chơi: Node thuộc lớp MAX thì gán cho nó giá trị lớn nhất của con Node đó. Node thuộc lớp MIN thì gán cho nó giá trị nhỏ nhất của con Node đó. Từ các giá trị này người chơi sẽ lựa chọn cho mình nước đi tiếp theo hợp lý nhất.



Như hình trên ta thấy là trạng thái hiện tại của game đang đến lượt đánh của người chơi X đại diện cho MAX. Ta tạm quy định giá trị MAX lúc game thắng cho $X = +10$ và MIN lúc game thua cho $X = -10$ và lúc game hòa $= 0$. Lúc này ở lượt 1, MAX có thể đi được 1 trong 3 nước như hình. Vậy làm sao để chọn 1 trong 3 nước đó nước nào là tốt nhất để đi. Chúng ta dựa vào giá trị của từng nước để chọn nước tốt nhất, như ở đây 3 node đó thuộc lớp MAX nên chọn giá trị lớn nhất. Chúng ta bắt đầu tìm giá trị của từng node đó. Ở lớp MAX trong lượt 1, thì ta có node 1,2,3 được đánh số từ trái sang phải như hình. Node 3 chúng ta đã là node lá (X win game) và có giá trị là +10. Còn 2 node 1,2 thì chưa biết giá trị của nó tại lượt 1 nên chúng ta dựa vào giá trị của các node con để định giá trị và bằng giá trị bé nhất của các node con ở lớp MIN tại lượt 2. Cứ tiếp tục tương tự như vậy đến lúc gặp node lá thì từ node lá đó ta suy ngược lại và ta tính được node 1 có giá trị là -10 và node 2 là 0. Vậy nước đi tốt nhất ở đây là như node 3 có giá trị lớn nhất là +10.

4.2 Kết nối TCP phía client:

```
//Step 1: Construct socket
if((sock = socket(AF_INET, SOCK_STREAM, 0)) == -1){
    strcpy(error,"Error Socket!!!");
    return -1;
}

//Step 2: Specify server address
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(PORT);
server_addr.sin_addr.s_addr = inet_addr(serverAddress);

// set timeout
struct timeval timeout;
timeout.tv_sec = 20; // after 20 seconds connect will timeout
timeout.tv_usec = 0;
if (setsockopt (sock, SOL_SOCKET, SO_RCVTIMEO, (char *)&timeout,
sizeof(timeout)) < 0){
    return -1;
}
else if (setsockopt (sock, SOL_SOCKET, SO_SNDTIMEO, (char
*)&timeout, sizeof(timeout)) < 0){
    return -1;
}

//Step 3: Request to connect server
if( connect(sock, (struct sockaddr*)&server_addr, sizeof(struct
sockaddr)) == -1){
    errorConnect = errno;
```

```

        sprintf(error,"Error! Can not connect to server!
%s",strerror(errorConnect));
        return -1;
    }
    //Step 4: Communicate with server
    send(sock, send_msg, strlen(send_msg), 0);
    recieved = recv(sock, recv_msg, BUFF_SIZE, 0);
    recv_msg[recieved] = '\0';
    strcpy(send_msg, SIGNAL_CLOSE);
    send(sock, send_msg, strlen(send_msg), 0);
    close(sock);
    if(recieved == -1){
        printf("\nError: Timeout!!!\n");
        return -1;
    }
    isCommunicating = 0; // ngat ket noi
    return 0;

```

4.3 Kết nối TCP phía server

```

// Step 1: Construct a TCP socket to listen connection request
if((sock = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
    perror("Socket error\n");
    exit(-1);
}

    if(setsockopt(sock,SOL_SOCKET,SO_REUSEADDR,&true,sizeof(int)) == -1) {
        perror("Setsockopt error\n");
        exit(-2);
    }

//Step 2: Bind address to socket
server_addr.sin_family = AF_INET;
server_addr.sin_port = htons(PORT);
server_addr.sin_addr.s_addr = INADDR_ANY;
bzero(&(server_addr.sin_zero),8);

if (bind(sock, (struct sockaddr *)&server_addr,
    sizeof(struct sockaddr)) == -1) {
    perror("Unable to bind\n");
    exit(-3);
}

//Step 3: Listen request from client
if (listen(sock, 5) == -1) {
    perror("Listen error\n");
    exit(-4);
}
printf("FUNGAME waiting for client on port %d\n", PORT);

```



```

fflush(stdout);

FD_SET(sock, &master);
fdmax = sock;

// Set timeout
struct timeval timeout;
timeout.tv_sec = 1000; // after 1000 seconds will timeout
timeout.tv_usec = 0;
//Step 4: Communicate with clients
while(1){
    read_fds = master;
    rc = select(fdmax + 1, &read_fds, NULL, NULL, &timeout);
    if( rc == -1){
        perror("select() error!\n");
        exit(-6);
    }
    else if (rc == 0){
        printf(" select() timed out. End program.\n");
        exit(-5);
    }
    for(i = 0; i <= fdmax; i++){
        if(FD_ISSET(i, &read_fds)){
            if(i == sock){
                sin_size = sizeof(struct sockaddr_in);
                connected = accept(sock, (struct
sockaddr*)&client_addr, &sin_size);
                if(connected == -1){
                    perror("accept error!\n");
                    exit(-7);
                }
            }
            else{
                FD_SET(connected, &master);
                if(connected > fdmax)
                    fdmax = connected;
                printf("Got a connection from (%s , %d) with fd =
%d\n",
inet_ntoa(client_addr.sin_addr), ntohs(client_addr.sin_por
t), connected);
                handleDataFromClient(connected);
            }
        }
        else{
            handleDataFromClient(i);
        }
    }
}
}

```

```
close(sock);  
return 0;
```

Ngoài ra, để client và server có thể trao đổi dữ liệu với nhau một cách dễ dàng nhất, nhóm em đã định nghĩa ra hàng loạt các tín hiệu để 2 bên cùng hiểu được đối phương đang muốn gì.

```
// fungame  
#define STATUS_START_MENU 0  
#define STATUS_FUNGAME_MENU 1  
#define STATUS_HANDLE_GAME 2  
#define SIGNAL_CHECKLOGIN "SIGNAL_CHECKLOGIN"  
#define SIGNAL_CREATEUSER "SIGNAL_CREATEUSER"  
#define SIGNAL_OK "SIGNAL_OK"  
#define SIGNAL_ERROR "SIGNAL_ERROR"  
#define SIGNAL_CLOSE "SIGNAL_CLOSE"  
  
// caro game  
#define SIGNAL_CARO_NEWGAME "SIGNAL_CARO_NEWGAME"  
#define SIGNAL_CARO_ABORTGAME "SIGNAL_CARO_ABORTGAME"  
#define SIGNAL_CARO_TURN "SIGNAL_CARO_TURN"  
#define SIGNAL_CARO_WIN "SIGNAL_CARO_WIN"  
#define SIGNAL_CARO_LOST "SIGNAL_CARO_LOST"  
#define SIGNAL_CARO_VIEWLOG "SIGNAL_CARO_VIEWLOG"  
#define SIGNAL_LOGLINE "SIGNAL_LOGLINE"  
  
// caro ranking  
#define SIGNAL_CARO_RANKING "SIGNAL_CARO_RANKING"  
  
// tictactoe game  
#define SIGNAL_TICTACTOE "SIGNAL_TICTACTOE"  
#define SIGNAL_TTT_RESULT "SIGNAL_TTT_RESULT"  
// #define SIGNAL_TICTACTOE_AI "SIGNAL_TICTACTOE_AI"  
  
// tictactoe ranking  
#define SIGNAL_TTT_RANKING "SIGNAL_TTT_RANKING"
```

4.4 Các công nghệ khác

- Sử dụng thư viện **termios.h** để ẩn thông tin mật khẩu khi nhập.
- Sử dụng linh hoạt thư viện màu sắc để tùy chỉnh giao diện chơi game.

Trước khi chơi game, người chơi cần đăng nhập thông tin tài khoản của mình. Khi đó client mở kết nối tới server, client gửi tín hiệu muốn kết nối đến server, server chấp nhận kết nối và hai bên tiến hành trao đổi dữ liệu.

5. Đánh giá

Trong quá trình kiểm thử, chúng em nhận thấy khi client kết nối với server, vấn đề xảy ra:

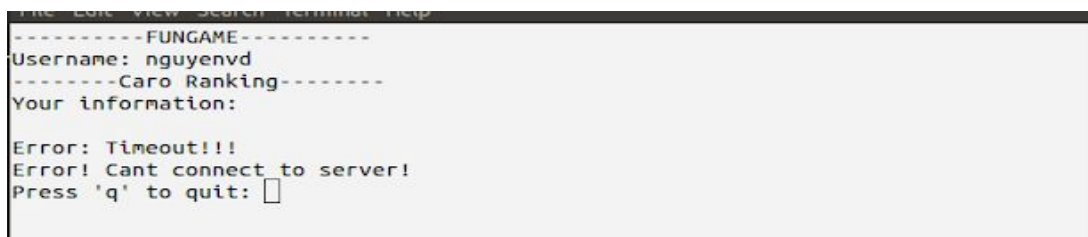
- Khi server bị một lỗi nào đó, không thể gửi trả dữ liệu cho client thì client sẽ đợi mãi mãi.
- Không có client nào kết nối với server nhưng server vẫn chờ, dẫn đến tình trạng lãng phí tài nguyên, server không biết khi nào nên thoát.

Chúng em đã giải quyết vấn đề này bằng kỹ thuật đặt giá trị timeout cho cả client và server.

Phía client

```
// set timeout
struct timeval timeout;
timeout.tv_sec = 20; // after 20 seconds will
timeout
timeout.tv_usec = 0;
if (setsockopt (sock, SOL_SOCKET, SO_RCVTIMEO, (char
*)&timeout, sizeof(timeout)) < 0){
    return -1;
}
else if (setsockopt (sock, SOL_SOCKET, SO_SNDTIMEO,
(char *)&timeout, sizeof(timeout)) < 0){
    return -1;
}
```

Kết quả

A screenshot of a terminal window with a dark background. The text is white and shows the following: a title bar at the top, followed by a separator line, then the text 'Username: nguyenvd', another separator line, 'Caro Ranking', and 'Your information:'. Below this, there is an 'Error: Timeout!!!' message, followed by 'Error! Cant connect to server!', and finally 'Press \'q\' to quit: ' followed by a cursor. The terminal window has a standard Linux-style title bar with 'File Edit View Search Terminal Help'.

Hình 16: Client báo timeout khi quá thời gian timeout

Phía server

```
// Set timeout
struct timeval timeout;
timeout.tv_sec = 1000; // after 1000 seconds will
timeout
timeout.tv_usec = 0;
//Step 4: Communicate with clients
read_fds = master;
rc = select(fdmax + 1, &read_fds, NULL, NULL,
&timeout);
if( rc == -1){
    perror("select() error!\n");
    exit(-6);
}
else if (rc == 0){
    printf(" select() timed out. End program.\n");
    exit(-5);
}
```

Kết quả



```
admina@vdm-27:~/Documents/Lập Trình Mạng/vidu/prjv2$ make
gcc server.c serverHelper.c caroai.c tic-tac-toe.c checkinput.c linklist.c ticta
ctoeRanking.c caroRanking.c -o server
admina@vdm-27:~/Documents/Lập Trình Mạng/vidu/prjv2$ ./server 5500
FUNGAME waiting for client on port 5500
select() timed out. End program.
admina@vdm-27:~/Documents/Lập Trình Mạng/vidu/prjv2$
```

Hình 13: Server dừng khi quá timeout, không có client nào tương tác

6. Phân công công việc

- Phân chia nhiệm vụ

STT	Nhiệm vụ	Thành viên đảm nhiệm
1	Code phần caro game + caro ranking	Vũ Đức Nguyễn + Nguyễn Việt Hưng
2	Code phần tictactoe game + tictactoe ranking	Vũ Đức Nguyễn + Phùng Thị Trang
3	Code phần signin/signout	Nguyễn Việt Hưng + Phùng Thị Trang
4	Báo cáo + slide + ý tưởng không hiển thị mật khẩu khi gõ	Cả 3 cùng lên ý tưởng và thực hiện
5	Xử lý và code phần "server, client bị đơ mãi mãi" bằng kỹ thuật đặt timeout	Vũ Đức Nguyễn

- Đánh giá mức độ đóng góp

STT	Thành viên	Mức độ hoàn thành	Phần trăm đóng góp
1	Vũ Đức Nguyễn	100%	34%
2	Nguyễn Việt Hưng	100%	33%
3	Phùng Thị Trang	100%	33%

7. Kết luận

Trên đây là toàn bộ nội dung về dự án của chúng em, được trình bày từ bước phân tích bài toán đến chương trình.

Qua dự án này, chúng em nắm chắc hơn kiến thức về mạng máy tính, ngôn ngữ lập trình C, và nhiều kỹ thuật mới khác nữa...

Vì còn đang trong quá trình học tập, kiến thức còn nhiều hạn chế nên sẽ có nhiều thiếu sót. Chúng em rất mong nhận được sự đóng góp ý kiến của thầy để chương trình tốt hơn nữa.

8. Tài liệu tham khảo

1. Slide bài giảng của thầy Đặng Tuấn Linh.
2. Sách "Unix-network-programming" của tác giả W. Richard Stevens, Bill Fenner, Andrew M. Rudoff.
3. Tài liệu về thư viện terminos:
https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.1.0/com.ibm.zos.v2r1.bpxbd00/rttcga.htm
4. Thuật toán Minimax:
<https://viblo.asia/p/thuat-toan-minimax-ai-trong-game-APqzeaVVzVe>