

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

BÁO CÁO BÀI TẬP LỚN

MÔN: THỰC HÀNH LẬP TRÌNH MẠNG

ĐỀ TÀI: Tetris Online - Xây dựng ứng dụng chơi game online có yếu tố tương tác "thời gian thực"



GVHD : TS. Trần Hải Anh

Lớp : 151908

Danh sách sinh viên: Nguyễn Anh Trà - 20215151

Đinh Huy Dương - 20215020

Hà Nội, Tháng 1 năm 2025

MỤC LỤC

MỤC LỤC.....	2
LỜI NÓI ĐẦU.....	3
PHÂN CÔNG THÀNH VIÊN TRONG NHÓM.....	4
CHƯƠNG 1. MÔ TẢ CÔNG VIỆC PROJECT.....	6
1.1. Giới thiệu về project.....	6
1.2. Các chức năng của chương trình.....	6
1.3. Kết quả cuối cùng.....	6
CHƯƠNG 2. THIẾT KẾ CHƯƠNG TRÌNH.....	7
2.1. Thiết kế tổng quan.....	7
2.1.1. Biểu đồ usecase.....	7
2.1.2. Thiết kế giao thức ứng dụng.....	7
2.1.3. Thiết kế cấu trúc thông điệp.....	7
2.2. Thiết kế cơ sở dữ liệu / cấu trúc dữ liệu.....	7
CHƯƠNG 3. CHƯƠNG TRÌNH MINH HOẠ.....	8
3.1. Xây dựng chương trình.....	8
3.2. Kiểm thử.....	8
3.3. Kết quả chương trình.....	8
CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	9
TÀI LIỆU THAM KHẢO.....	10
PHỤ LỤC.....	11

LỜI NÓI ĐẦU

Cùng với sự phát triển của máy tính, việc phát triển của các trò chơi điện tử - game, cũng là một xu hướng tất yếu của ngành giải trí hiện nay. Thuở đầu của ngành công nghiệp phát triển game bắt đầu với những game được phát triển một cách độc lập và chỉ có duy nhất một người chơi. Tuy nhiên, cùng với sự trưởng thành mạnh mẽ của ngành mạng máy tính, các game đã được phát triển để có thể tạo ra những trải nghiệm mang tính trực tuyến, nơi mà có thể có nhiều người chơi kết nối và đồng bộ với nhau. Điều này không chỉ tăng cường tính giải trí mà còn thúc đẩy sự kết nối và giao lưu giữa người chơi toàn thế giới.

Cụ thể, game xếp hình (mang tên Tetris) - là một tựa game cổ điển, được nhiều người chơi và trải qua nhiều thế hệ người dùng yêu mến, luôn có những tiềm năng để đưa từ game đơn người dùng thành một tựa game có thể kết nối các trận đấu xếp hình giữa các người chơi kỳ cựu lâu năm của tựa game nổi tiếng này lại với nhau. Với mục tiêu khám phá và ứng dụng những kỹ thuật lập trình mạng, chúng em đã quyết định chọn đề tài *Tetris Online - Xây dựng ứng dụng chơi game online có yếu tố tương tác "thời gian thực"*. Đề tài này không chỉ giúp nhóm em tiếp cận và tìm hiểu sâu hơn về các giao thức truyền thông mạng mà còn rèn luyện khả năng thiết kế hệ thống, xử lý đồng bộ và tối ưu hiệu năng trong môi trường nhiều người dùng.

Trong quá trình thực hiện, nhóm chúng em đã nghiên cứu các kiến thức nền tảng về lập trình socket, xây dựng kiến trúc server-client, và tích hợp giao diện chơi game Tetris quen thuộc. Đồng thời, việc đảm bảo tính ổn định, mượt mà cho trò chơi trong môi trường trực tuyến đã giúp nhóm em hiểu rõ hơn về những thách thức và giải pháp trong lĩnh vực phát triển ứng dụng tương tác thời gian thực.

Để hoàn thành được bài tập lớn này, nhóm chúng em xin được gửi lời cảm ơn chân thành đến thầy Trần Hải Anh hướng dẫn đề tài, Giảng viên Trường Công nghệ Thông tin và Truyền thông, Đại Học Bách khoa Hà Nội - đã hết lòng giúp đỡ, hướng dẫn, chỉ dạy tận tình để nhóm em hoàn thành được đề tài này.

Hà Nội, tháng 1 năm 2025

Lớp Thực hành lập trình mạng - 151908

PHÂN CÔNG THÀNH VIÊN TRONG NHÓM

STT	Họ tên	MSSV	Công việc đóng góp	Mức độ hoàn thành
1	Đinh Huy Dương	20215020	<ul style="list-style-type: none"> - Nghiên cứu và thiết kế game offline. Trong đó có lập trình các khối hình trong game, điểm số, danh sách người chơi tạo bảng xếp hạng và kết nối thuật toán với giao diện - Thiết kế giao diện Client sử dụng SDL2, bao gồm Menu, chuyển đổi màn hình, xử lý sự kiện, và giao diện Game - Lập trình xử lý thông điệp cho Client để gửi cho Server, cũng như xử lý phản hồi - Vá lỗi bên Server và Client, bao gồm cả lỗi liên quan đến đóng socket và lỗi segmentation của giao diện SDL2 - Thiết kế các biểu đồ và làm slide, báo cáo 	-Đóng góp 50% vào đề tài -Hoàn thành:100%
2	Nguyễn Anh Trà	20215151	<ul style="list-style-type: none"> - Đóng góp 50% vào đề tài - Tạo code base, bao gồm lập trình các hàm tiện ích như log, hàm khởi tạo sẵn socket cho cả client và server - Thiết kế giao thức của server với client. Cụ thể là lập trình các struct Message, - Thiết kế Database - Thiết kế Server bao gồm thiết kế hàm tạo mã phiên (session), tạo câu lệnh truy vấn cho Database và gửi qua server, kiểm soát các người chơi Online, kiểm soát phòng,... - Lập trình xử lý thông điệp của Client và gửi phản hồi, bao gồm tạo, vào phòng, quảng bá thông điệp cho phòng, bắt đầu game,... - Thiết kế giao diện client trên terminal để test các thông điệp - Tạo các file Cmake, build_project.sh để build project đơn giản hơn (hiện tại không sử dụng những file này nữa) 	-Đóng góp 50% vào đề tài -Hoàn thành:100%

CHƯƠNG 1. MÔ TẢ CÔNG VIỆC PROJECT

1.1. Giới thiệu về project

Tetris Online là một ứng dụng trò chơi trực tuyến được xây dựng nhằm tái hiện lại trò chơi Tetris kinh điển với các yếu tố tương tác "thời gian thực". Dự án không chỉ đơn thuần là một game giải trí mà còn là cơ hội để nhóm phát triển áp dụng và nâng cao kiến thức về lập trình mạng, xử lý đồng bộ dữ liệu và thiết kế hệ thống ứng dụng.

Project này là một trò chơi Tetris đa người chơi được phát triển bằng ngôn ngữ lập trình C. Trò chơi cho phép nhiều người chơi kết nối với nhau thông qua một server trung tâm, nơi họ có thể tạo phòng chơi, tham gia phòng chơi, và thi đấu với nhau.

Các thành phần chính của Project:

- **Server:** Server chịu trách nhiệm quản lý các kết nối từ client, xử lý các yêu cầu từ người chơi, quản lý các phòng chơi, và duy trì trạng thái của trò chơi. Server sử dụng **PostgreSQL** để lưu trữ thông tin về người chơi, phòng chơi, và các trận đấu.
- **Client:** Client là ứng dụng mà người chơi sử dụng để kết nối đến server, tạo phòng chơi, tham gia phòng chơi, và chơi trò chơi Tetris. Client sử dụng thư viện **SDL2** để hiển thị giao diện đồ họa và xử lý các sự kiện từ người chơi.

1.2. Các chức năng của chương trình

Dự án tập trung vào việc xây dựng hệ thống server-client hoạt động ổn định, cho phép nhiều người chơi cùng tham gia vào một phòng chơi, đồng bộ trạng thái trò chơi giữa các client trong thời gian thực. Các tính năng bao gồm:

- **Đăng ký và đăng nhập:** Người chơi có thể tạo 1 tài khoản và sử dụng tài khoản này làm định danh để thi đấu với người chơi khác.
- **Tạo phòng chơi:** Người chơi có thể tạo phòng chơi mới và trở thành chủ phòng. Khi tạo phòng, người chơi có thể thiết lập các thông số như giới hạn thời gian, giới hạn số lượng gạch, và số lượng người chơi tối đa.
- **Tham gia phòng chơi:** Người chơi có thể tham gia vào các phòng chơi đã được tạo bởi người chơi khác. Nếu phòng chơi đã đầy hoặc không tồn tại, người chơi sẽ nhận được thông báo lỗi. Việc tham gia này sử dụng tên phòng từ danh sách các phòng đã được thông báo cho người ngay từ lúc vào game, hoặc vào phòng ngẫu nhiên mà không cần phải nhập tên phòng.
- **Bắt đầu trò chơi:** Chủ phòng có thể bắt đầu trò chơi. Trò chơi sẽ bắt đầu và tất cả người chơi trong phòng sẽ tham gia vào cùng một trận đấu Tetris. Hệ thống server đảm bảo các gạch (shape) được gửi đồng bộ đến tất cả người chơi. Luật game là phòng sẽ có cố định 1 danh sách gạch giống nhau và được gửi khi trò chơi bắt đầu.

- **Chơi Tetris:** Người chơi có thể điều khiển các khối gạch rơi xuống bằng cách di chuyển sang trái, phải, xoay và tăng tốc độ rơi. Trò chơi sẽ kết thúc khi khối gạch chạm đến đỉnh của màn hình hoặc khi hết thời gian hoặc hết gạch trong danh sách gạch của Server.
- **Bảng xếp hạng điểm số:** Người chơi sau khi phá (clear) hết 1 hàng ngang trên màn hình game, sẽ nhận được điểm và Server sẽ thông báo cho mọi người chơi khác trong phòng. Luật game là khi clear được 1 hàng, người chơi sẽ được 100 điểm.

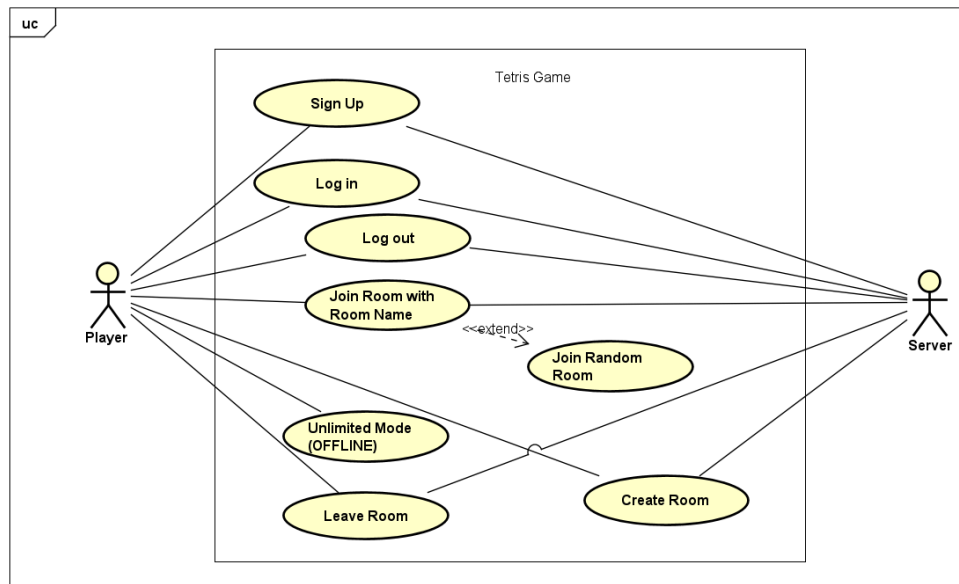
1.3. Kết quả cuối cùng:

Kết quả đầu ra cuối cùng của project là một trò chơi Tetris đa người chơi hoàn chỉnh, với các chức năng đăng ký, đăng nhập, tạo phòng chơi, tham gia phòng chơi, bắt đầu trò chơi, cập nhật điểm số, và ngắt kết nối. Trò chơi cung cấp giao diện đồ họa dễ sử dụng và trải nghiệm chơi game mượt mà cho người chơi. Server quản lý kết nối từ client, xử lý các yêu cầu từ người chơi, và duy trì trạng thái của trò chơi, đảm bảo rằng trò chơi diễn ra một cách công bằng và ổn định. Cụ thể chúng ta sẽ được xem ở phần minh họa chương trình.

CHƯƠNG 2. THIẾT KẾ CHƯƠNG TRÌNH

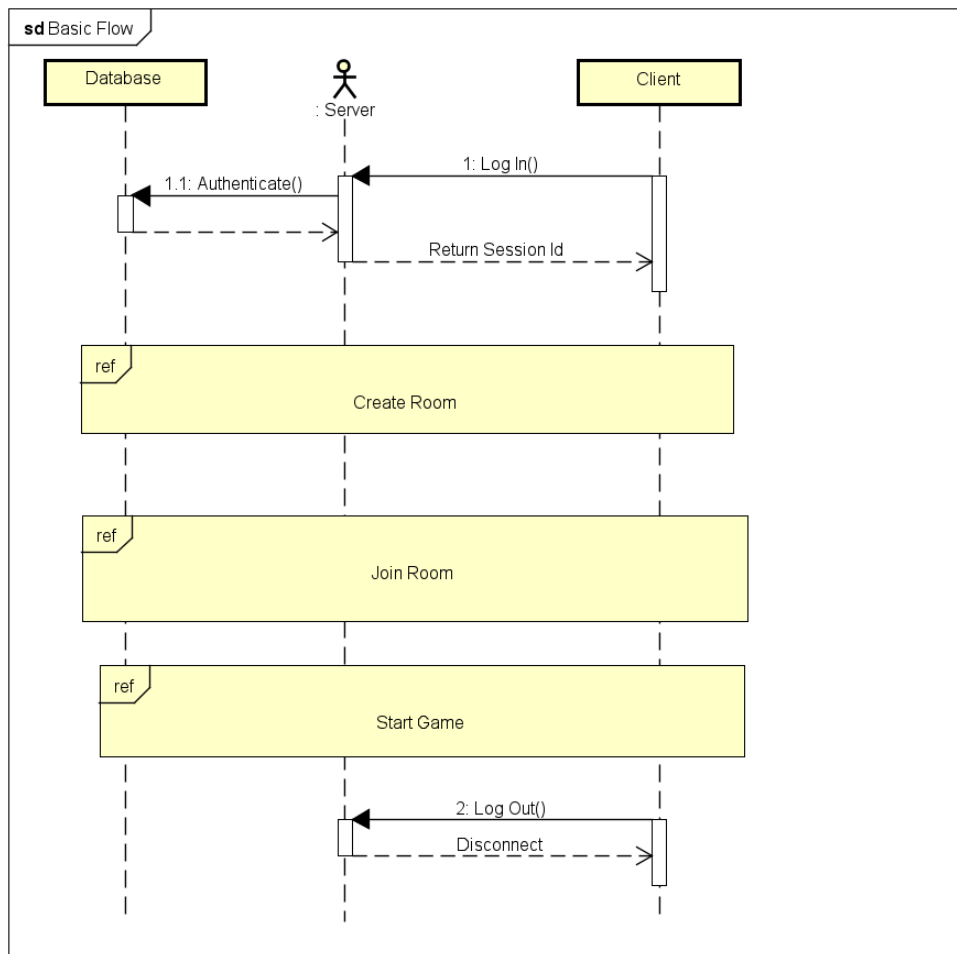
2.1. Thiết kế tổng quan

2.1.1. Biểu đồ usecase

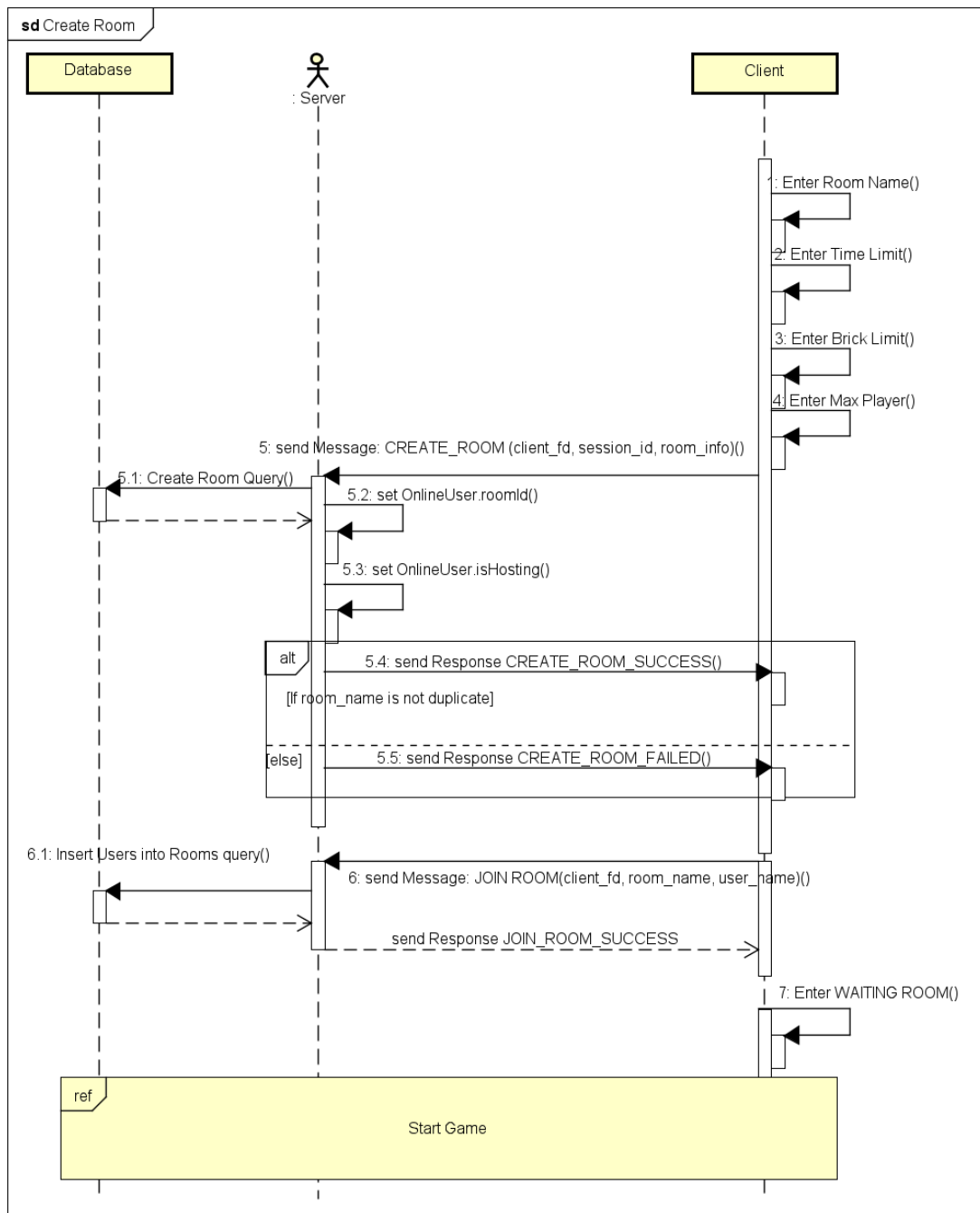


- **Tác nhân:** Người chơi (Player/ Client) và Server
- **Các Use Case:**
 - **Sign Up:** Đăng ký. Người chơi nhập tên đăng nhập và mật khẩu. Server kiểm tra trong Database xem tên đã tồn tại hay chưa, nếu có lưu thông tin người chơi vào trong Database và thông báo đăng ký thành công. Nếu chưa báo lỗi và yêu cầu người chơi nhập lại.
 - **Log In:** Đăng nhập. Người chơi cung cấp tên và mật khẩu để đăng nhập vào hệ thống. Server kiểm tra hệ thống, nếu thông tin đăng nhập không chính xác, thông báo cho người chơi; nếu chính xác thì người chơi được chuyển đến màn hình chính của game.
 - **Create Room:** Người chơi có thể tạo một phòng chơi mới với các thông số như tên phòng, giới hạn thời gian, giới hạn số lượng khối, và số lượng người chơi tối đa. Server kiểm tra xem tên phòng đã tồn tại chưa, nếu chưa tạo phòng mới và cập nhật bảng trong Database. Nếu đã tồn tại phòng này, thông báo lỗi.
 - **Join Room with Room Name:** Người chơi tham gia vào 1 phòng hiện có bằng cách cung cấp tên phòng. Server kiểm tra phòng này có tồn tại không, hoặc kiểm tra xem phòng có đủ người chưa, cho phép người dùng tham gia vào phòng và cập nhật vào Database. Nếu không tồn tại hoặc đã đầy, thông báo cho người dùng.
 - **Join Random Room:** Người chơi tham gia vào 1 phòng hiện có nhưng không cần cung cấp tên phòng

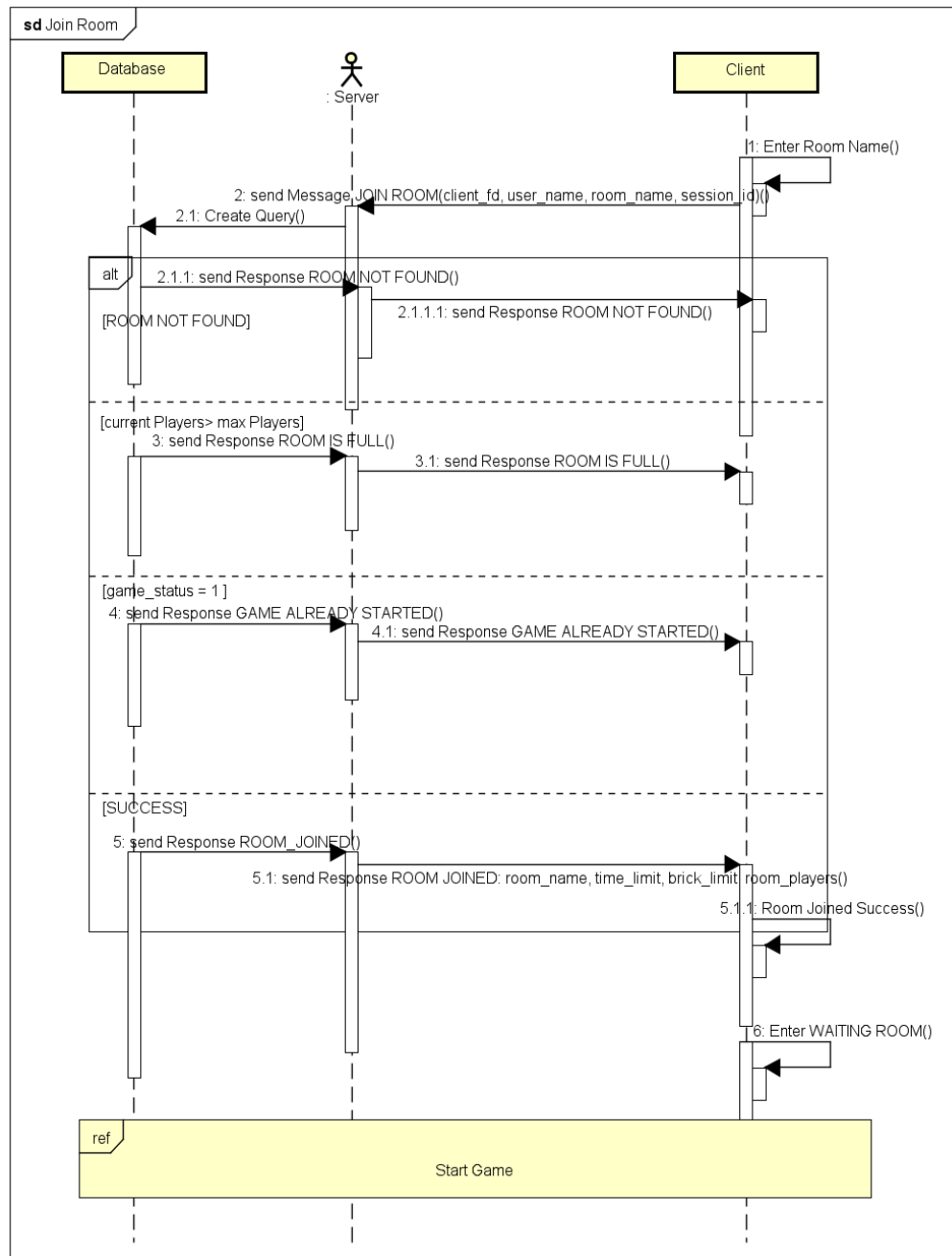
2.1.2. Thiết kế giao thức ứng dụng



Luồng cơ bản: Client (Người chơi) sau khi đăng nhập thành công bởi Server xác thực với Database, sẽ được trả về mã phiên (Session Id). Lúc này Client sẽ được chuyển tới Menu chính. Client từ đây có thể Create Room để tạo phòng, Join Room để tham gia vào 1 phòng. Và sau khi vào phòng Client có thể Start Game để bắt đầu trò chơi (nếu là Host của phòng). Sau khi xong một game, người chơi có thể quay về Menu chính để tiếp tục lại. Client có thể đăng xuất (Log out) bất cứ lúc nào.

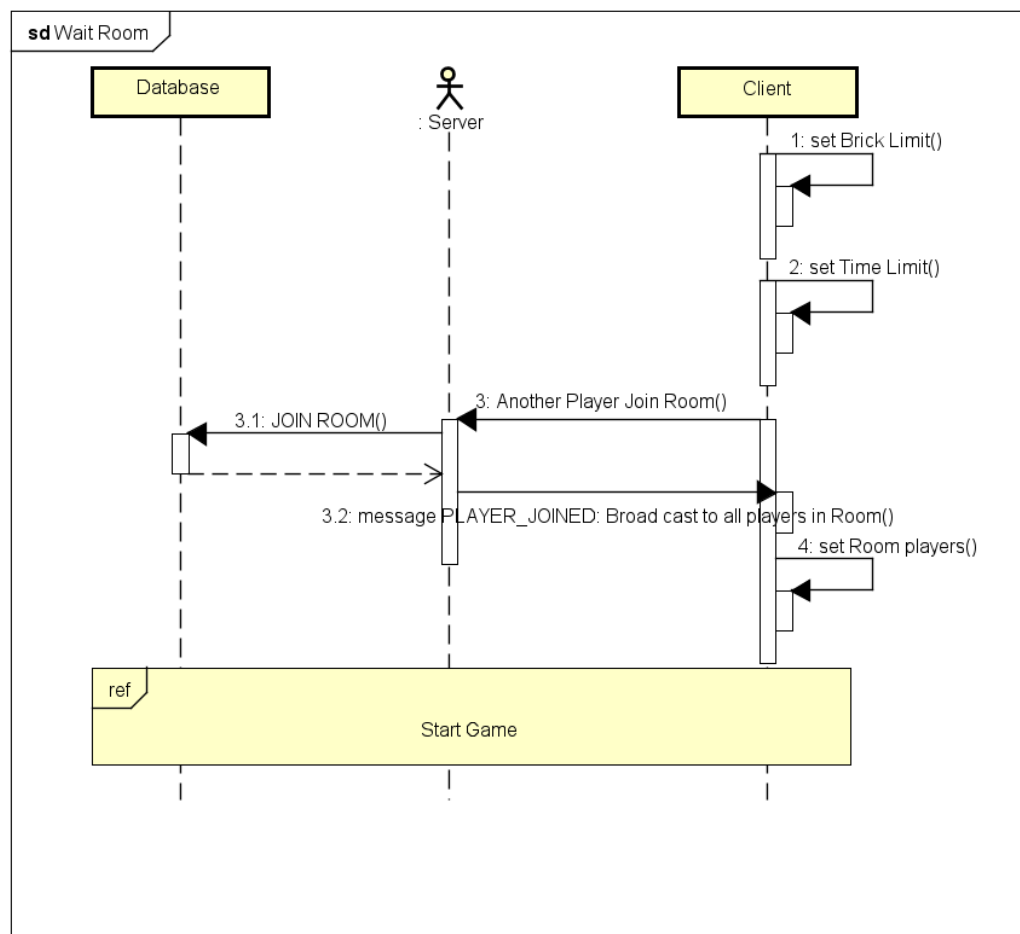


Tạo phòng (Create Room): Client cung cấp tên phòng, thời gian giới hạn của 1 Game (Time Limit), số lượng gạch tối đa (Brick Limit), và số lượng người chơi tối đa (Max Player) tạo thông điệp và gửi cho Server. Server nhận được thông điệp sẽ tạo câu lệnh INSERT phòng mới vào trong bảng Rooms của Database. Nếu tên phòng đã tồn tại trả về tạo phòng không thành công. Ngược lại thông báo tạo phòng thành công cho Client, bên Server sẽ đặt Online User hiện tại (Client hiện tại) là Host của phòng hiện tại. Online User là danh sách các người chơi đang online bao gồm các socket của các Client để Server quản lý. Cuối cùng, cho Client Host vào trong phòng này với thông điệp JOIN ROOM, để Server tạo INSERT Client vào bảng RoomPlayers của Database. Người chơi lúc này sẽ tiến vào phòng chờ (WAITING ROOM) để có thể Start Game sau này.

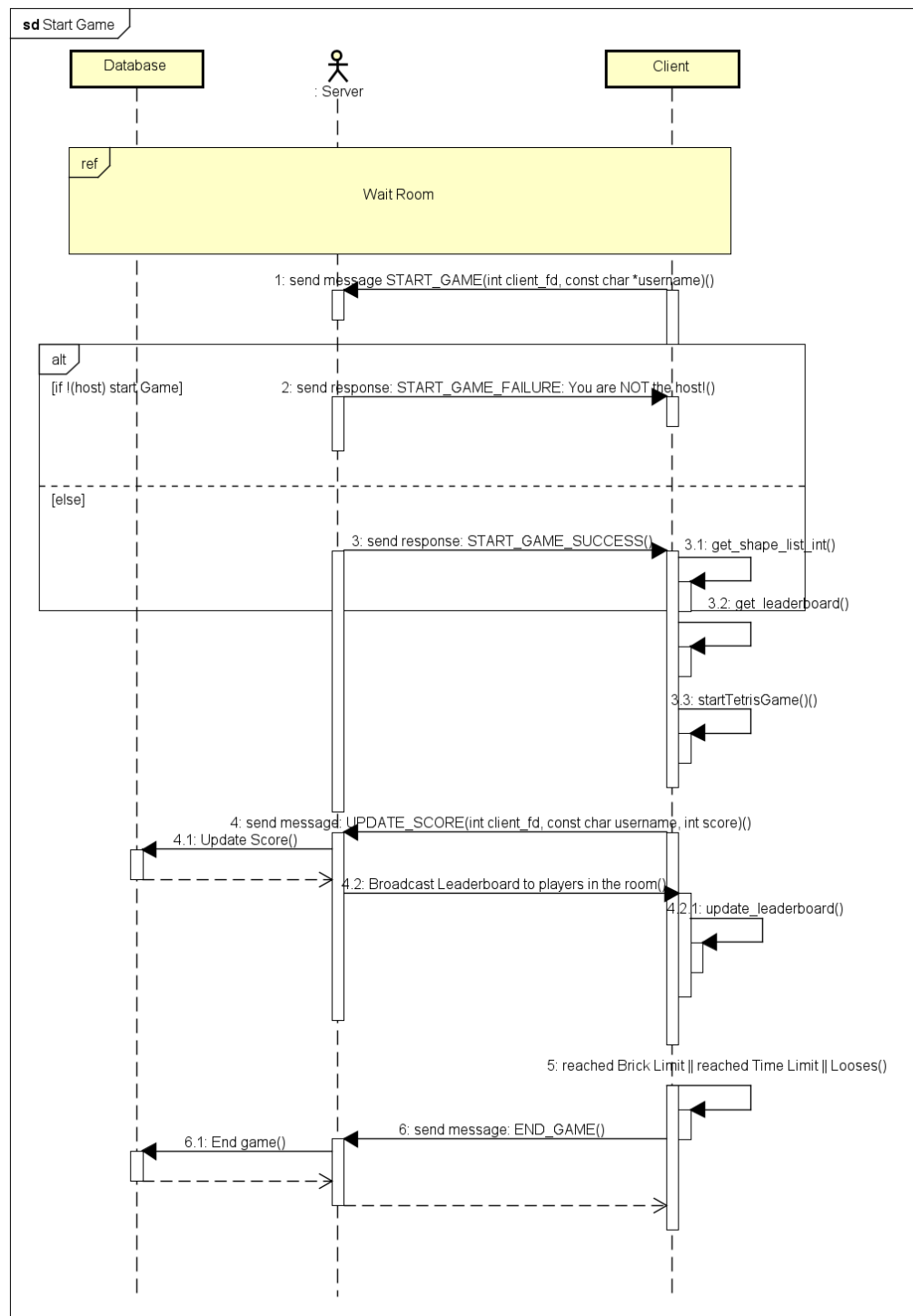


Tham gia phòng (Join Room): Client cung cấp tên phòng và tạo thông điệp và gửi cho Server. Server tạo INSERT Client vào bảng RoomPlayers của Database. Nếu Server gửi về tên phòng không tồn tại, hoặc phòng đã đầy, hoặc game đã bắt đầu (*Tính năng chưa hoàn thiện*) thì Client sẽ nhận được các thông điệp không thể tham gia vào phòng. Ngược lại nhận được thông điệp ROOM JOINED SUCCESS và tiến vào phòng chờ để có thể được host Start Game và bắt đầu trò chơi. Rõ ràng Client dùng chức năng này sẽ không phải là Host của phòng.

Tham gia phòng ngẫu nhiên (Join Random Room): Tương tự như trên, nhưng Client không phải cung cấp tên phòng. Điểm khác biệt nữa là Server sẽ thay đổi lệnh truy vấn ở trên để trong đó tìm được phòng ngẫu nhiên và thêm người chơi vào phòng đó.



Phòng chờ: Tại phòng chờ, nếu người dùng ấn Enter, Server sẽ kiểm tra xem người dùng đó có phải host của phòng không. Nếu sai thì nhận lại được thông báo “Chỉ Host mới có thể bắt đầu Game”, ngược lại bắt đầu Game!



Bắt đầu game (Start Game): Từ phòng chờ, sau khi Host của phòng nhấn Enter để bắt đầu game, các người chơi trong phòng sẽ được nhận về thông điệp START GAME SUCCESS, trong đó bao gồm danh sách các gạch mà Server tạo ra (Danh sách này là danh sách số, mà lúc Client nhận được sẽ dịch ngược lại sang các gạch với thứ tự tương ứng). Trò chơi bắt đầu, cứ mỗi viên gạch được đặt ở cuối màn hình, người chơi sẽ gửi kiểm tra mình được bao nhiêu điểm và gửi điểm này cho Server. Server sẽ quảng bá cho tất cả người chơi trong phòng điểm số hiện tại của người chơi này và các người chơi khác sẽ cập nhật bảng xếp hạng tương ứng. Sau đó, tùy vào những điều kiện sau: Hết thời gian giới hạn, hết gạch, gạch chạm đến đỉnh màn hình, trò chơi sẽ

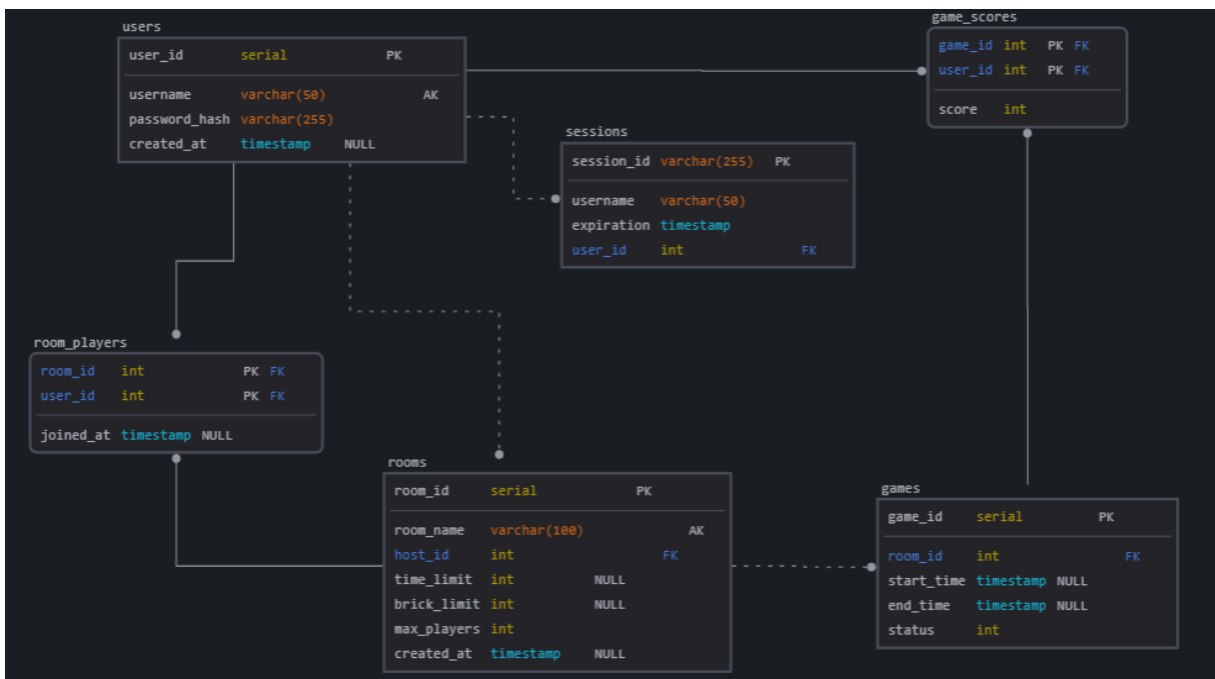
kết thúc. Người chơi vẫn có thể nhận được cập nhật điểm của người chơi khác nếu chưa rời ra khỏi phòng. Client ấn Enter để thoát ra khỏi phòng và quay về Menu chính, hoặc ấn tắt ứng dụng để Disconnect và thoát Game. Sau khi Disconnect, server đóng Socket của Client, và thông báo cho các người chơi khác trong phòng là người chơi đã thoát, nếu như họ vẫn ở trong phòng.

2.1.3. Thiết kế cấu trúc thông điệp

```
// Message types
typedef enum {
    REGISTER, REGISTER_SUCCESS, REGISTER_FAILURE,
    LOGIN, LOGIN_SUCCESS, LOGIN_FAILURE,
    CREATE_ROOM, CREATE_ROOM_SUCCESS, CREATE_ROOM_FAILURE,
    JOIN_ROOM, JOIN_ROOM_SUCCESS, JOIN_ROOM_FAILURE, ROOM_NOT_FOUND, ROOM_FULL,
    GAME_ALREADY_STARTED, PLAYER_JOINED, JOIN_RANDOM, START_GAME, START_GAME_FAILURE, START_GAME_SUCCESS,
    UPDATE_SCORE,
    ROOM_LIST,
    GOT_ROOM_LIST,
    ROOM_JOINED,
    GAME_START,
    GAME_STATUS,
    END_GAME,
    DISCONNECT,
    LOG_OUT
} MessageType;
// Message structure
typedef struct {
    MessageType type;
    char username[MAX_USERNAME];
    char room_name[MAX_ROOM_NAME];
    char data[BUFFER_SIZE];
} Message;
```

Ảnh trên nói về thông điệp được sử dụng trong Server và Client. Trong 1 Message, có mang tên của người gửi - **username** và phòng của người chơi (nếu có) - **room_name**. Bên cạnh đó là trường dữ liệu gửi - **data**, tùy thuộc vào kiểu thông điệp-**MessageType**

2.2. Thiết kế cơ sở dữ liệu / cấu trúc dữ liệu



- **Bảng users:** Lưu trữ thông tin người dùng.
 - user_id: Khóa chính, tự động tăng.
 - username: Tên người dùng, duy nhất và không được để trống.
 - password_hash: Mật khẩu đã được mã hóa, không được để trống.
 - created_at: Thời gian tạo, mặc định là thời gian hiện tại.
- **Bảng sessions:** Lưu trữ thông tin phiên đăng nhập của người dùng.
 - session_id: Khóa chính.
 - username: Tên người dùng, tham chiếu đến bảng users, không được để trống, xóa theo chuỗi.
 - expiration: Thời gian hết hạn, không được để trống.
- **Bảng rooms:** Lưu trữ thông tin các phòng chơi.
 - room_id: Khóa chính, tự động tăng.
 - room_name: Tên phòng, duy nhất và không được để trống.
 - host_id: ID của người chủ phòng, tham chiếu đến bảng users, không được để trống, xóa theo chuỗi.
 - time_limit: Giới hạn thời gian, phải lớn hơn 0.
 - brick_limit: Giới hạn số gạch, phải lớn hơn 0.
 - max_players: Số người chơi tối đa, phải nằm trong khoảng từ 2 đến 8.
 - created_at: Thời gian tạo, mặc định là thời gian hiện tại.
- **Bảng room_players:** Lưu trữ thông tin người chơi trong các phòng.
 - room_id: ID của phòng, tham chiếu đến bảng rooms, không được để trống, xóa theo chuỗi.
 - user_id: ID của người dùng, tham chiếu đến bảng users, không được để trống, xóa theo chuỗi.
 - joined_at: Thời gian tham gia, mặc định là thời gian hiện tại.
 - Khóa chính kết hợp của room_id và user_id.
- **Bảng games:** Lưu trữ thông tin các trò chơi.
 - game_id: Khóa chính, tự động tăng.
 - room_id: ID của phòng, tham chiếu đến bảng rooms, không được để trống, xóa theo chuỗi.
 - start_time: Thời gian bắt đầu, mặc định là thời gian hiện tại.
 - end_time: Thời gian kết thúc.
 - status: Trạng thái, mặc định là 1, chỉ nhận giá trị 0 hoặc 1.
- **Bảng game_scores:** Lưu trữ điểm số của người chơi trong các trò chơi.
 - game_id: ID của trò chơi, tham chiếu đến bảng games, không được để trống, xóa theo chuỗi.
 - user_id: ID của người dùng, tham chiếu đến bảng users, không được để trống, xóa theo chuỗi.
 - score: Điểm số, mặc định là 0.
 - Khóa chính kết hợp của game_id và user_id.

Chú thích: Bảng game_scores sử dụng cho mục đích tương lai để lưu lại điểm với tính năng “**Lịch sử game**” nếu muốn làm. Thực tế trong 1 game chỉ cập nhật điểm tức thời chưa được lưu lại vào đâu.

CHƯƠNG 3. CHƯƠNG TRÌNH MINH HOẠ

3.1. Xây dựng chương trình

Cấu trúc thư mục Project bao gồm:

- Thư mục src:
 - client: Chứa các file mã nguồn liên quan đến client được test ở terminal, bao gồm các hàm tiện ích (client_utils.c) và các định nghĩa (client_utils.h).
 - server: Chứa các file mã nguồn liên quan đến server, bao gồm các hàm xử lý cơ sở dữ liệu (database.c), các đối tượng (object.c), và các hàm chính của server (server.c).
 - protocol: Chứa các file định nghĩa giao tiếp mạng giữa client và server (protocol.h).
 - tetris_game.c và tetris_game.h: Chứa các hàm và định nghĩa liên quan đến trò chơi Tetris, bao gồm việc khởi tạo trò chơi, xử lý các sự kiện, và cập nhật trạng thái của trò chơi.
 - client_utils.c và client_utils.h: Chứa các hàm xử lý gửi các thông điệp cho Server
 - utils.c: Chứa các hàm tiện ích viết log được sử dụng trong project.
 - main.c: Hàm chạy chính của Client, trong đó bao gồm 1 thread (sử dụng pthread) để xử lý nhận các thông điệp từ bên Server
- Thư mục config: Chứa các file cấu hình cho client (client_config.h) và server (server_config.h). **LƯU Ý:** Folder này là cho Legacy, chỉ phục vụ cho giai đoạn đầu, hiện tại không sử dụng
- Thư mục build:
 - Chứa các file và thư mục liên quan đến quá trình build project, bao gồm các file CMake và các file biên dịch.
 - **LƯU Ý:** Folder này là cho Legacy, chỉ phục vụ cho giai đoạn đầu, hiện tại không sử dụng
- Thư mục assets:
 - Chứa các tài nguyên như font chữ (font.ttf) và các thư mục con chứa các tài nguyên khác (fonts).
- Thư mục design:
 - Chứa các file thiết kế liên quan đến project, như các hình ảnh mô tả trình tự tạo phòng chơi (CreateRoomSequence.png.bak) và trình tự PvP (PvP Sequence.asta).
- Thư mục test:
 - Chứa các file mã nguồn và tài nguyên liên quan đến việc kiểm thử project.
- Các file khác:
 - CMakeLists.txt: File cấu hình CMake cho project.
 - Makefile: File Makefile để build project.
 - **README.md:** Chứa hướng dẫn build chương trình và thông tin về project.
 - build_project.sh: Script để build project.

- **LƯU Ý:** Các file Makefile, CMakeList, build_project.sh chỉ sử dụng ở giai đoạn đầu, hiện tại không sử dụng nữa. Chi tiết build chương trình ở README.md

4. Compile the Client

1. Navigate to the client source directory:

```
cd src
```

2. Run the following command to compile the client:

- Linux

```
gcc main.c tetris_game.c client_utils.c utils.c -o  
game.o -lSDL2 -lSDL2_ttf -lpthread
```

- macOS

```
gcc main.c tetris_game.c client_utils.c utils.c -o  
game.o \  
-I/opt/homebrew/include/SDL2 -L/opt/homebrew/lib -  
lSDL2 -lSDL2_ttf -lpthread
```

- **Output:** This will generate an executable file named `game.o` in the `src` directory.
- **NOTE:** AGAIN, please check the path to the SDL2 and SDL2_ttf library for the compiler to link the library accurately

3.2. Kiểm thử

- Chưa tích hợp được các công cụ kiểm thử tự động, kiểm thử đơn vị,..
- Tuy nhiên cần phải kiểm thử luồng chương trình để kiểm thử lỗi bộ nhớ cấp phát động, cũng như gửi thông điệp, quản lý socket người như nào. Sử dụng các công cụ như valgrind, và những hàm log tiện ích nói trên ở utils.c


```

==1058== Process terminating with default action of signal 11 (SIGSEGV)
==1058== Access not within mapped region at address 0x0
==1058== at 0x10CE18: renderLeaderboard (in /mnt/c/Users/phuon/OneDrive/Documents/Network Programming/tetris-websocket/src/game.o)
==1058== by 0x10D124: renderGame (in /mnt/c/Users/phuon/OneDrive/Documents/Network Programming/tetris-websocket/src/game.o)
==1058== by 0x10B061: startTetrisGame (in /mnt/c/Users/phuon/OneDrive/Documents/Network Programming/tetris-websocket/src/game.o)
==1058== by 0x10B995: main (in /mnt/c/Users/phuon/OneDrive/Documents/Network Programming/tetris-websocket/src/game.o)
==1058== If you believe this happened as a result of a stack
==1058== overflow in your program's main thread (unlikely but
==1058== possible), you can try to increase the size of the
==1058== main thread stack using the --main-stacksize= flag.
==1058== The main thread stack size used in this run was 8388608.
==1058== LEAK SUMMARY:
==1058== definitely lost: 102,557 bytes in 64 blocks
==1058== indirectly lost: 48,352 bytes in 91 blocks
==1058== possibly lost: 7,777,306 bytes in 47,338 blocks
==1058== still reachable: 8,254,383 bytes in 15,323 blocks
==1058== of which reachable via heuristic:
==1058== newarray : 418,080 bytes in 5 blocks
==1058== multipleinheritance: 3,080 bytes in 5 blocks
==1058== suppressed: 0 bytes in 0 blocks
==1058==
==1058== For lists of detected and suppressed errors, rerun with: -s
==1058== ERROR SUMMARY: 32744 errors from 31234 contexts (suppressed: 10 from 4)

```

Sử dụng valgrind.log để kiểm tra lỗi Segmentation fault

ID	Username	Session ID	SocketFD	IP Address	Port	Last Activity	Authenticated
1	player1	cfc9065-7798-4935-ae3b-265b49e7b397	4	127.0.0.1	37638	2024-12-28 15:10:55	

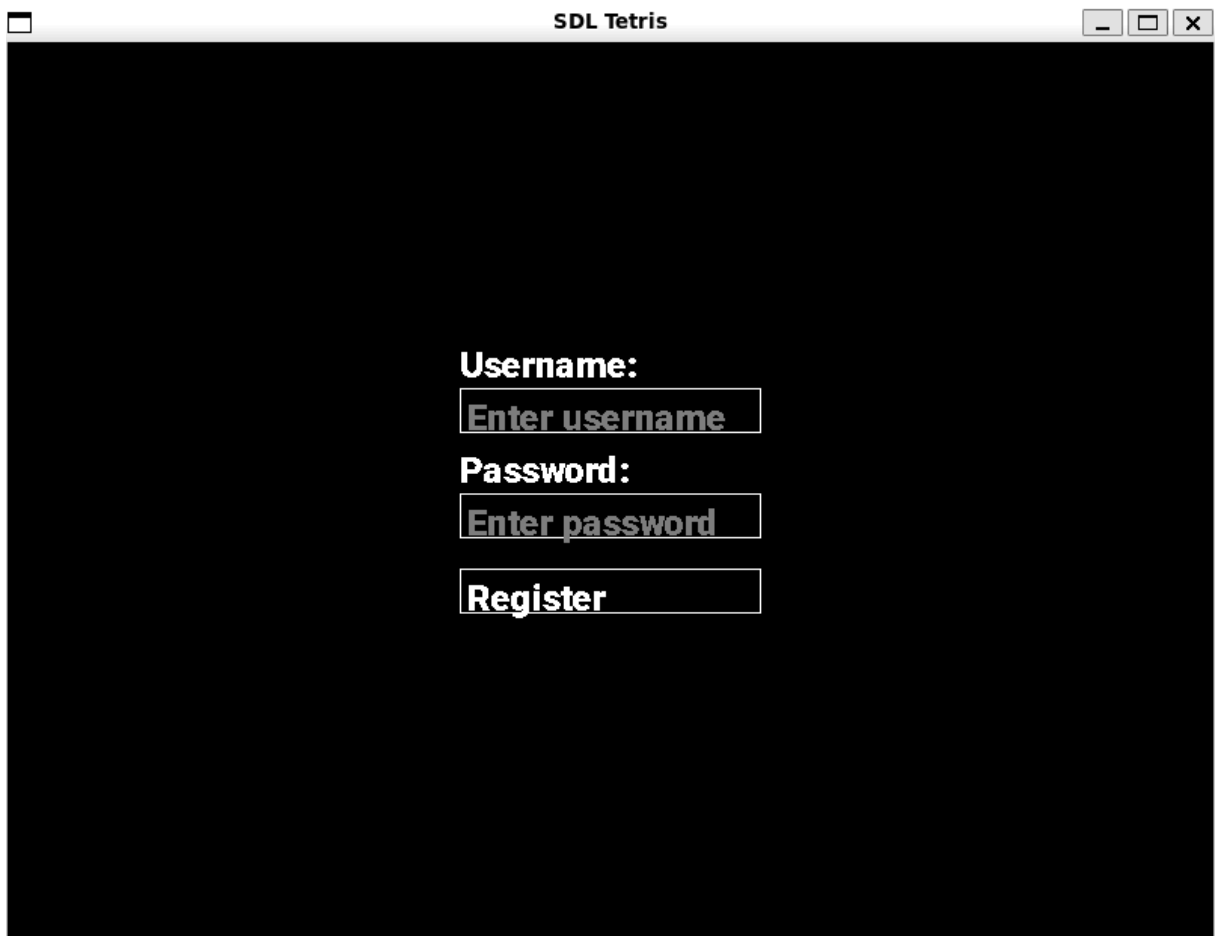
```

[Sat Dec 28 15:11:14 2024] =====
[Sat Dec 28 15:11:20 2024] Server received message of type 6 from user player1 (IP: 127.0.0.1, SocketFD: 4)
[Sat Dec 28 15:11:20 2024] handle_create_room: Starting function execution
[Sat Dec 28 15:11:20 2024] handle_create_room: Received message
[Sat Dec 28 15:11:20 2024] t3
[Sat Dec 28 15:11:20 2024] cfc9065-7798-4935-ae3b-265b49e7b397|30000|20|8
[Sat Dec 28 15:11:20 2024] Parsed session_id:
[Sat Dec 28 15:11:20 2024] cfc9065-7798-4935-ae3b-265b49e7b397
[Sat Dec 28 15:11:20 2024] Parsed time_limit:
[Sat Dec 28 15:11:20 2024] 30000
[Sat Dec 28 15:11:20 2024] Parsed brick_limit:
[Sat Dec 28 15:11:20 2024] 20
[Sat Dec 28 15:11:20 2024] Parsed max_player:
[Sat Dec 28 15:11:20 2024] 8
[Sat Dec 28 15:11:20 2024] handle_create_room: Converted parameters to strings:
[Sat Dec 28 15:11:20 2024] 30000
[Sat Dec 28 15:11:20 2024] 20
[Sat Dec 28 15:11:20 2024] 8
[Sat Dec 28 15:11:20 2024] handle_create_room: Preparing to execute query
[Sat Dec 28 15:11:20 2024] handle_create_room: Query executed, status: PGRES_FATAL_ERROR
[Sat Dec 28 15:11:20 2024] handle_create_room: PostgreSQL query execution error
[Sat Dec 28 15:11:20 2024] ERROR: duplicate key value violates unique constraint "rooms_room_name_key"
DETAIL: Key (room_name)=(t3) already exists.
[Sat Dec 28 15:11:20 2024] handle_create_room: Room creation failed
[Sat Dec 28 15:11:20 2024] handle_create_room: Returning response

```

Sử dụng hàm log tiện ích để ghi lại hoạt động các socket ở bên Server

3.3. Kết quả chương trình



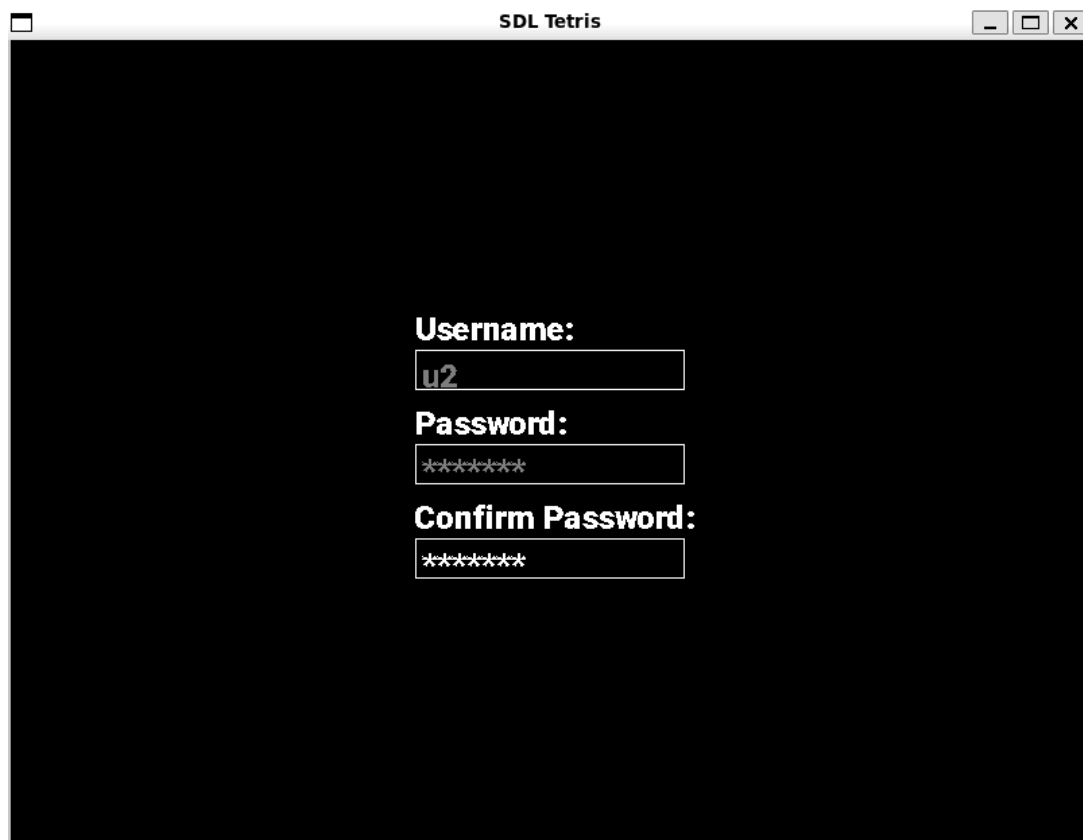
SDL Tetris

Username:
Enter username

Password:
Enter password

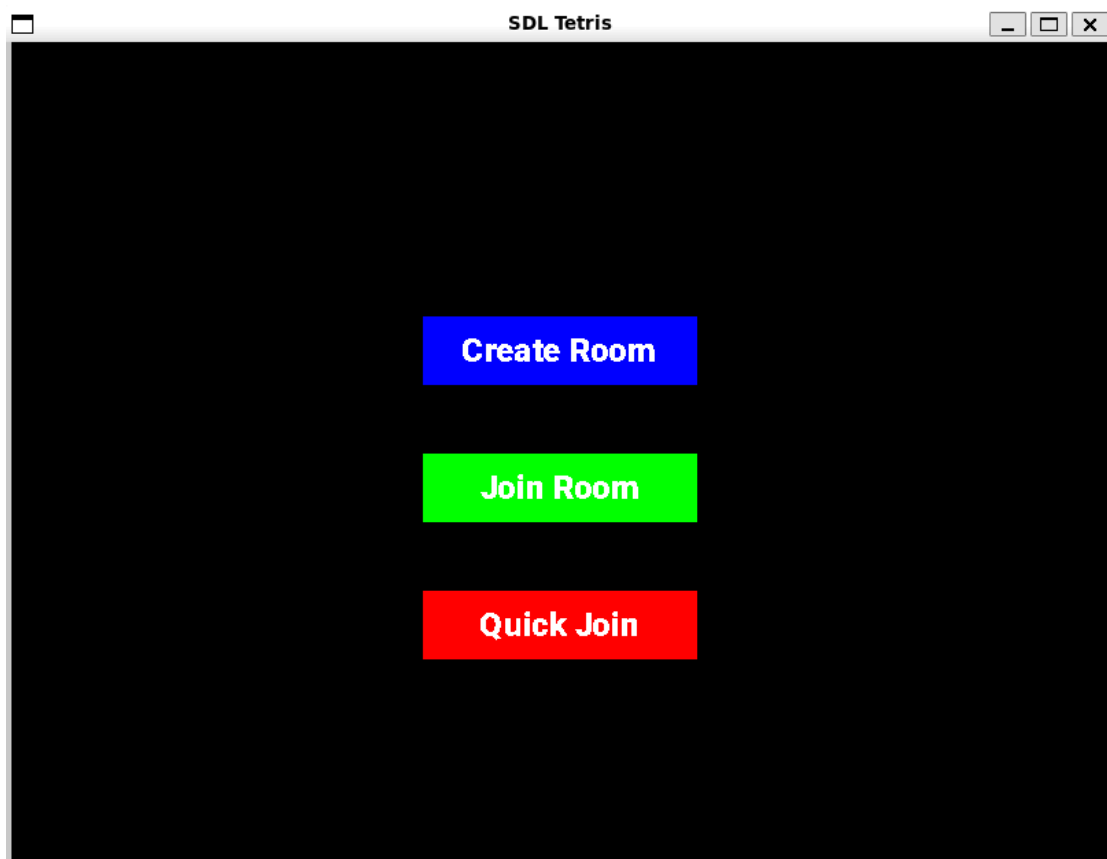
Register

Màn hình đăng nhập



A screenshot of a web browser window titled "SDL Tetris". The background is black. In the center, there are three white text labels: "Username:", "Password:", and "Confirm Password:". Below each label is a white rectangular input field. The "Username" field contains the text "u2". The "Password" and "Confirm Password" fields contain seven asterisks "*****".

Màn hình đăng ký



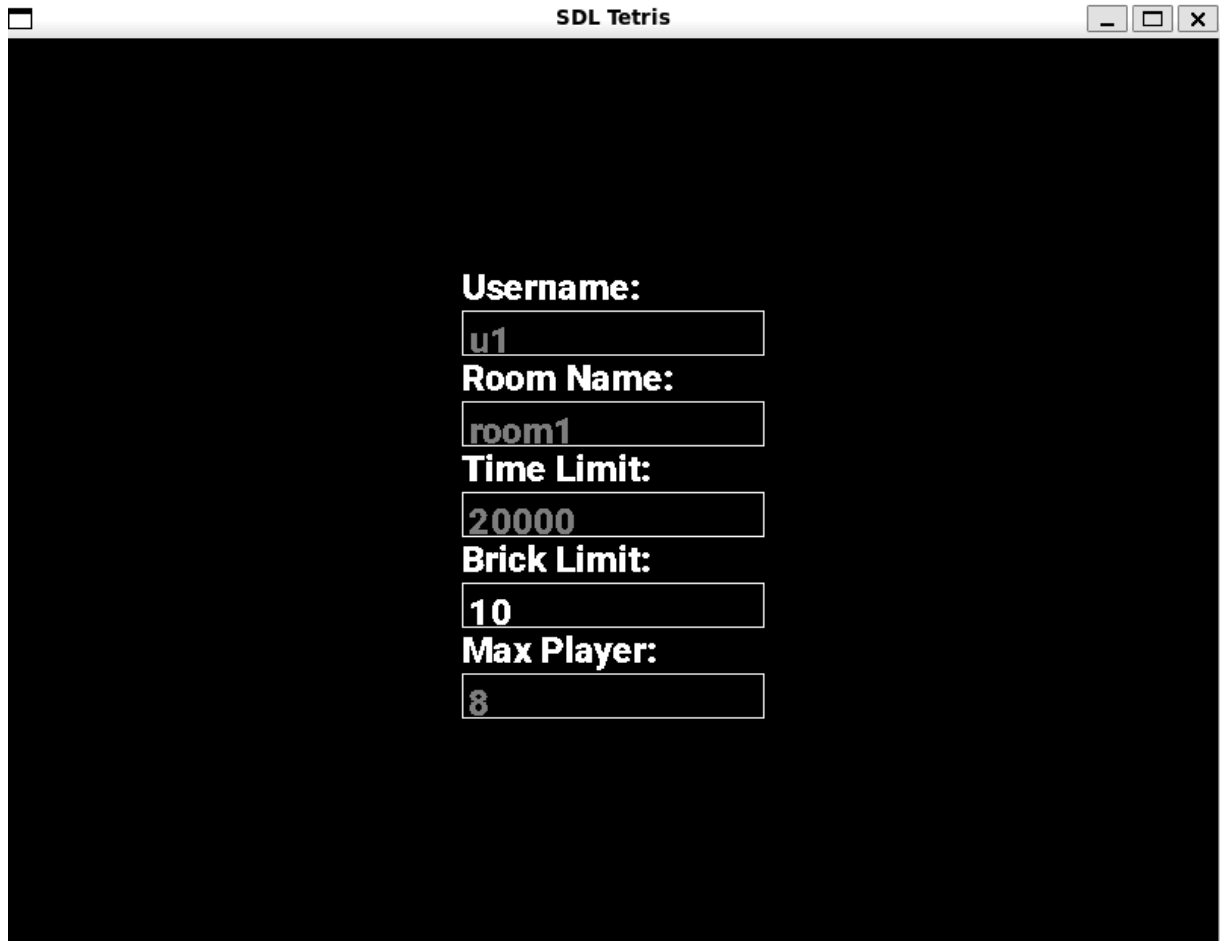
A screenshot of a web browser window titled "SDL Tetris". The background is black. In the center, there are three colored rectangular buttons stacked vertically. The top button is blue with the text "Create Room" in white. The middle button is green with the text "Join Room" in white. The bottom button is red with the text "Quick Join" in white.

Màn hình menu chính

Tetris Online

```
Login successful! Welcome, u2.  
Login success.  
Server: Number of rooms: 3  
Room ID: 270, Room Name: r1, Host ID: 1, Time Limit: 20000, Brick Limit: 20, Max Players: 8  
Room ID: 271, Room Name: r2, Host ID: 1, Time Limit: 20000, Brick Limit: 20, Max Players: 8  
Room ID: 272, Room Name: room1, Host ID: 21, Time Limit: 20000, Brick Limit: 10, Max Players: 8
```

Danh sách các phòng (Ở terminal) sau khi đăng nhập



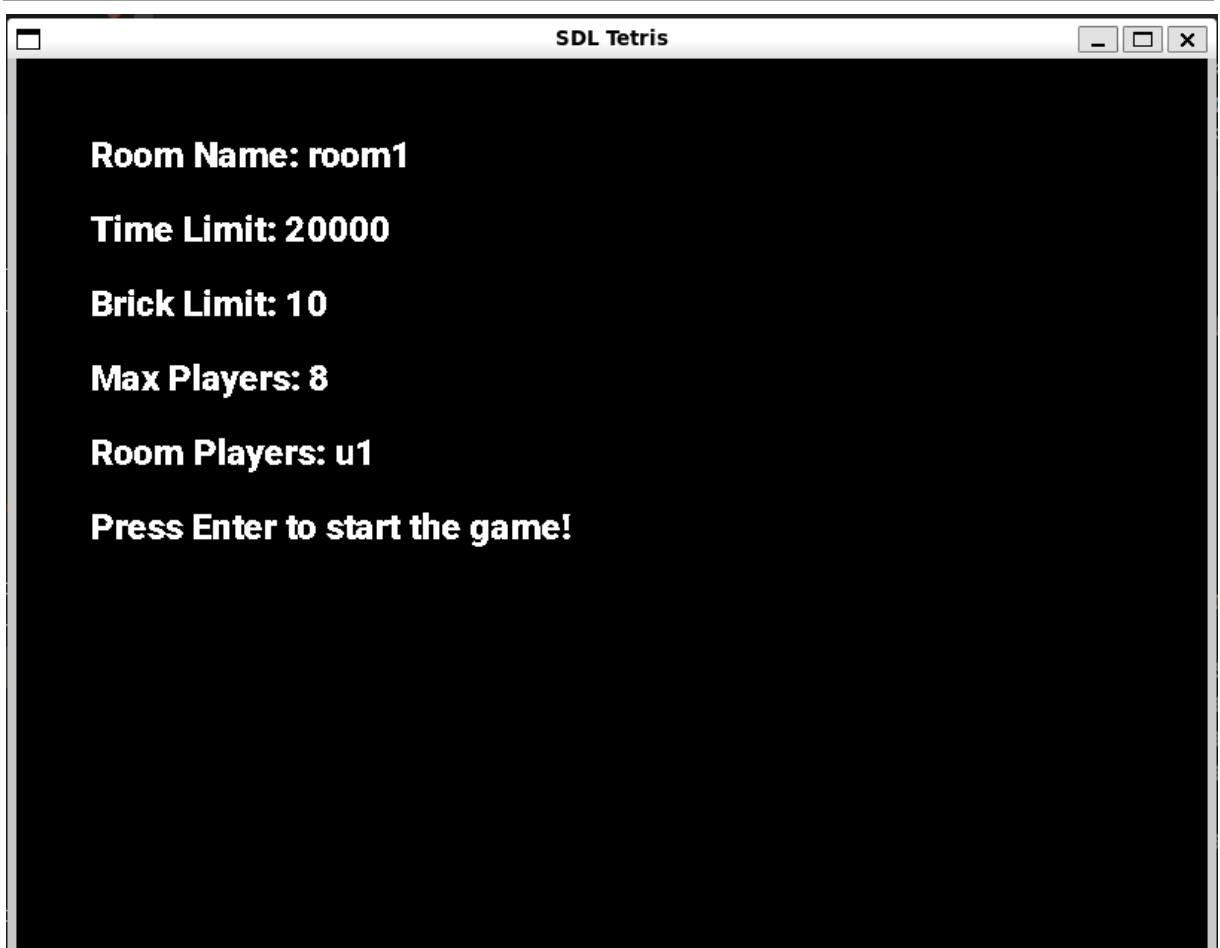
The screenshot shows a window titled "SDL Tetris" with a black background. In the center, there is a form with white text and input fields. The form contains the following labels and values:

- Username:** u1
- Room Name:** room1
- Time Limit:** 20000
- Brick Limit:** 10
- Max Player:** 8

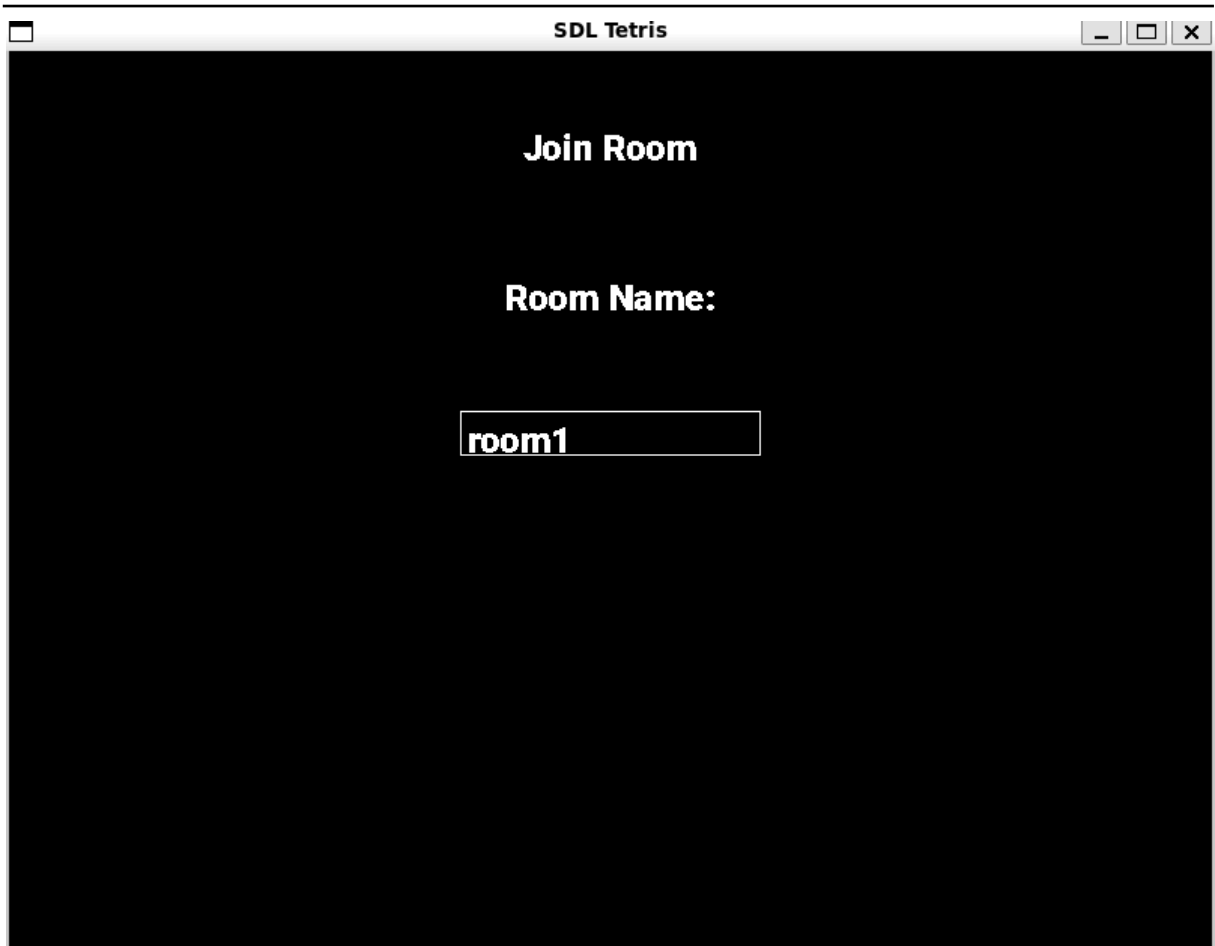
Màn hình Tạo phòng (Create Room)

```
Create Room button clicked  
Room created successfully: Room created successfully!  
Another player joined: Player 'u1' has joined the room.  
Successfully joined room: room1  
Server message: room1|20000|10|8|u1
```

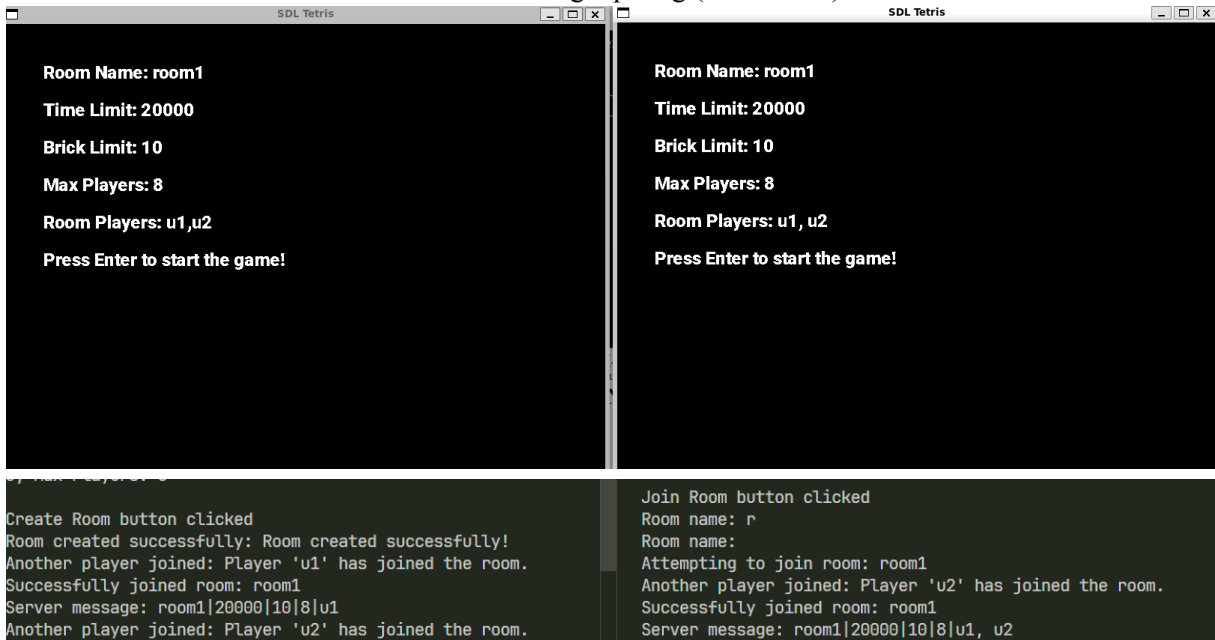
Thông điệp của Server (Ở terminal) sau khi tạo phòng



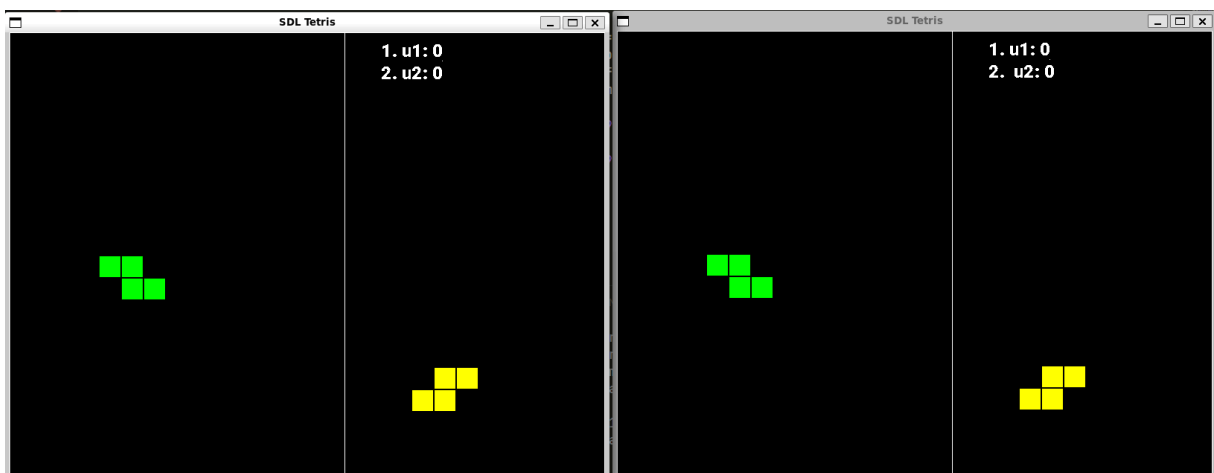
Màn hình phòng chờ (Waiting Room)



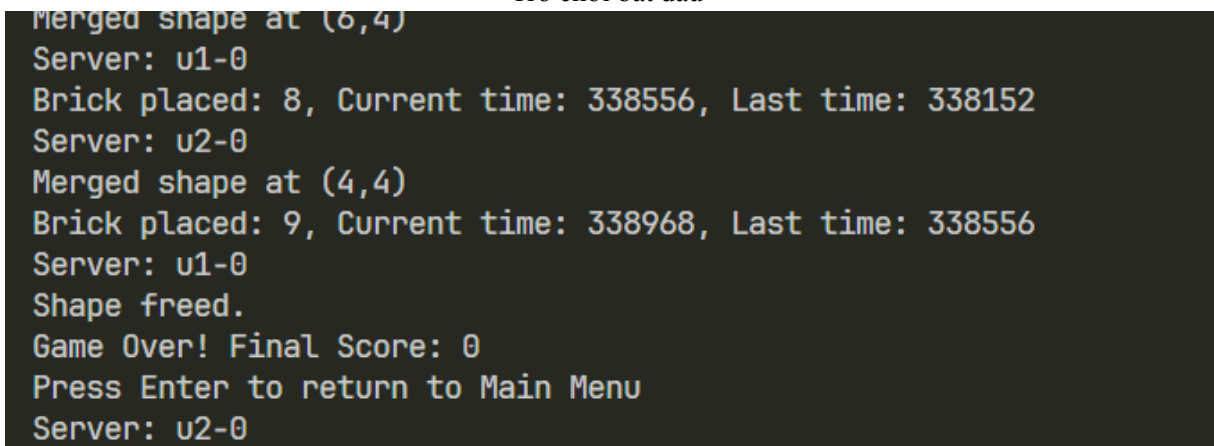
Màn hình Tham gia phòng (Join Room)



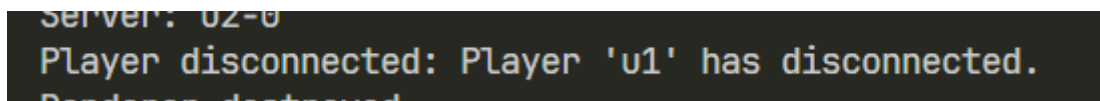
Màn hình chờ của 2 người được cập nhật khi người khác tham gia vào và thông điệp của Server (hiện trên Terminal)



Trò chơi bắt đầu



Server nhận thông điệp cập nhật điểm từ Client và quảng bá cho các người chơi trong phòng



Nhận thông báo người chơi khác trong phòng đã disconnected

CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Nhóm đã hoàn thiện được đại khái các chức năng cơ bản của dự án với việc hoạt động trao đổi điểm, quản lý tham gia phòng chơi và tạo tài khoản cho người chơi. Nhìn chung đã thỏa mãn được một ứng dụng áp dụng được những kiến thức về lập trình socket đa luồng để có thể trao đổi thông tin giữa các người dùng theo thời gian thực.

Tuy nhiên dự án của nhóm chúng em vẫn chưa hoàn thiện ở đôi chỗ. Lỗi đầu tiên là có thể hoàn toàn dựa trên SDL, khi mà đồng bộ các trang màn hình khác nhau chưa đồng đều, khiến cho đôi lúc, sau lần chơi đầu sau đăng nhập, Host sẽ không thể hiện ra giao diện màn nhưng người chơi khác lại có thể vào trong game được. Giải pháp của chúng em ở đây có thể sử dụng các thư viện giao diện khác, hoặc sử dụng ngôn ngữ khác làm giao diện, như HTML, Javascript, Java,... Một số bảng cũng chưa được sử dụng hợp lý, như đã đề cập ở phần thiết kế cơ sở dữ liệu, bảng `player_scores` vẫn chưa được sử dụng. Tuy nhiên ở phần này bảng này sẽ để dành cho tính năng trong tương lai, đó là lưu lại lịch sử của các ván game (Game History). Tính năng này là một tính năng thú vị, tuy nhiên trong khuôn khổ của dự án chúng em vẫn chưa dành ra thời gian để làm được. Hơn nữa danh sách các phòng hiện có vẫn đang ở trên giao diện terminal, chúng em chưa tạo được một popup/ màn hình riêng để hiển thị danh sách các phòng, đây cũng là hạn chế của SDL2, khi thư viện đồ họa đòi hỏi phải vẽ (render) từng các thành phần của giao diện bằng các hàm khá cơ bản, chưa tạo thành 1 trang giao diện lên một cách dễ dàng được. Cách khắc phục ở đây đơn giản nhất là sử dụng thư viện khác, tuy nhiên nếu trực tiếp nhúng tay lại vào SDL2, chúng em sẽ có thể sửa lại các hàm render màn hình thành 1 hàm chung nếu chung các thành phần giao diện, có thể sử dụng các Union của các struct thành phần giao diện để có thể linh hoạt trong các thành phần của một màn hình. Cùng với đó là khó khăn trong việc học hỏi và áp dụng thư viện đồ họa SDL2 một cách hiệu quả, khi việc debug trong giao diện cực kỳ mệt mỏi, công cụ debug cơ bản như gdb lại không hiệu quả bởi SDL2 là thư viện mà chấp nhận nhiều rò rỉ bộ nhớ, khiến cho việc tìm ra segmentation fault gặp cực nhiều khó khăn. Tại điều này, chúng em xin cố gắng việc những đoạn code thật “clean” để tránh làm khó bản thân, cũng như debug đơn giản hơn.

Mặt khác việc tổ chức quản lý nhóm cũng là một trở ngại. Nhóm có 3 thành viên, nhưng chỉ có 2 người thực hiện công việc, và với một dự án có khối lượng lớn như này, đây là một thách thức lớn. Hơn thế nữa những trắc trở về việc học công nghệ mới, quản lý thời gian cũng làm cho nhóm có những khoảng khắc mệt mỏi. Tuy nhiên, vượt qua những rào cản này, chúng em - Trà và Dương, vẫn cố gắng hoàn thành công việc để đưa ra sản phẩm mượt mà nhất để demo. Có thể dự án vẫn còn nhiều lỗ hổng và cần sửa chữa, nhưng trong tầm khả năng của mình, chúng em xin nhận trách nhiệm và sẽ cố gắng hoàn thiện, trau dồi kỹ năng cá nhân để có thể đưa ra những sản phẩm hoàn thiện hơn trong tương lai. Em xin chân thành cảm ơn mọi người đã giúp đỡ em trong thời gian đã qua và thời gian sắp tới.

TÀI LIỆU THAM KHẢO

- [1] UNIX Network Programming Volume 1, Third Edition: The Sockets Networking API, tác giả Addison Wesley
- [2] Trang Web Tetr.io
- [3] Github SDL-tetris của olzhasar

PHỤ LỤC

Hướng dẫn cài đặt chi tiết đã được đặt trong phần README.md của thư mục project. Lưu ý phần cài đặt các thư viện có sự khác biệt giữa Linux và macOS, với các trình cài đặt khác nhau (apt và brew), nhìn lưu ý này trong file README