TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI TP. HCM

**KHOA CÔNG NGHỆ THÔNG TIN**

A colorful text on a black background

Description automatically generated

**BÁO CÁO MÔN HỌC**

**LẬP TRÌNH MẠNG**

**TÀI LIỆU ĐẶC TẢ PHẦN MỀM**

**GAME STUDOKU**

GVHD: Bùi Dương Thế

SVTH:

052205008668\_Phan Dương Khang

MSSV\_Ngô Viết Tài

MSSV\_Võ Phi Tịnh

MSSV\_Trương Công Lý

Mục Lục

[Danh sách hình 2](#_Toc198945334)

[I.Giới thiệu chung 2](#_Toc198945335)

[1. Studoku là gì? 2](#_Toc198945336)

[2. Lý do chọn đề tài 2](#_Toc198945337)

[3. Mô tả ngắn về sản phẩm 2](#_Toc198945338)

[II. Mô tả tổng quát 2](#_Toc198945339)

[II.1 Chức năng 2](#_Toc198945340)

[II.2 Tổng quan về Game Studoku 4](#_Toc198945341)

[II.2.1 Luật chơi chi tiết 4](#_Toc198945342)

[III.3. Phân loại người dùng 8](#_Toc198945343)

[IV. Kiến trúc hệ thống 9](#_Toc198945344)

[IV.1. kiến trúc tổng thể của hệ thống 9](#_Toc198945345)

[IV.2. Chi tiết các thành phần 10](#_Toc198945346)

[V. Kết quả và đánh giá 15](#_Toc198945347)

[5.1 Kinh nghiệm rút ra 15](#_Toc198945348)

[5.2 Kết quả đạt được 15](#_Toc198945349)

[5.3 Kết quả chưa đạt được 16](#_Toc198945350)

[5.4 Hướng phát triển 16](#_Toc198945351)

[VI. Tài liệu tham khảo 16](#_Toc198945352)

# Danh sách hình

[Hình 1: : Component diagram 10](#_Toc198945976)

[Hình 2: Class diagram 11](#_Toc198945977)

[Hình 3: Use case 12](#_Toc198945978)

[Hình 4: Activity diagram Multiplayer Server 13](#_Toc198945979)

[Hình 5: Activity diagram hệ thống Sudoku 14](#_Toc198945980)

[Hình 6: Activity diagram Client 15](#_Toc198945981)

[Hình 7: Sequence Diagram: Server Process Make Move Request 15](#_Toc198945982)

[Hình 8: Sequence Diagram: Start New Classic Game 16](#_Toc198945983)

[Hình 9: Sequence Diagram: Input Number into Cell (Classic Mode) 17](#_Toc198945984)

[Hình 10: Sequence Diagram: Connect to Multiplayer Server 17](#_Toc198945985)

# I.Giới thiệu chung

1. Studoku là gì?

Sudoku là một trò chơi trí tuệ có nguồn gốc từ Nhật Bản, được thiết kế dưới dạng một bảng gồm 9 hàng và 9 cột, chia thành 9 vùng 3x3. Mục tiêu của người chơi là điền các chữ số từ 1 đến 9 vào các ô trống sao cho mỗi hàng, mỗi cột và mỗi vùng 3x3 đều chứa đầy đủ các số từ 1 đến 9, không được trùng lặp.

1. Lý do chọn đề tài

Đề tài Game Sudoku được chọn nhằm tạo ra một sản phẩm vừa mang tính giải trí vừa rèn luyện tư duy logic. Qua quá trình phát triển game, người thực hiện có thể áp dụng và củng cố kiến thức lập trình, thuật toán và thiết kế giao diện. Sudoku là trò chơi phổ biến, dễ tiếp cận, có giá trị giáo dục và tiềm năng mở rộng cao, phù hợp để phát triển thành một ứng dụng hữu ích và thực tiễn.

1. Mô tả ngắn về sản phẩm

Game Sudoku của em được thiết kế với giao diện cơ bản, dễ sử dụng, tập trung vào việc giải đố. Màn hình chính sẽ là lưới 9x9 quen thuộc, nơi người chơi có thể chọn ô để điền số. Các chức năng chính dự kiến bao gồm:

1. Tạo bàn chơi mới: Sinh ra một ván Sudoku mới (có thể có tùy chọn độ khó).

2. Nhập/Xóa số: Cho phép người chơi điền số vào các ô trống hoặc xóa số đã điền.

3. Kiểm tra lỗi: Hệ thống sẽ cảnh báo nếu người chơi điền số vi phạm luật Sudoku.

4. Gợi ý: Cung cấp một nước đi đúng nếu người chơi gặp khó khăn."

# II. Mô tả tổng quát

1. Chức năng

1.1 Chức năng khởi tạo và hiển thị game Studoku (Cung cấp các công cụ để khởi tạo một ván Sudoku mới và hiển thị giao diện bàn chơi.)

| **TT (Mục chính)** | **STT (Mục con)** | **Tên Chức Năng** | **Mô Tả** |
| --- | --- | --- | --- |
| **2.1. Khởi tạo và Hiển thị Game** | 2.1 | Tạo Bàn Chơi Mới (New Game / Generate Puzzle) | Người chơi có thể yêu cầu hệ thống tạo một ván Sudoku mới. Hệ thống sẽ sinh ra một bảng Sudoku hợp lệ với một số ô đã được điền sẵn (ô gợi ý). |
|  | 2.2 | Lựa Chọn Độ Khó (Difficulty Level) | Trước khi bắt đầu, người chơi có thể chọn mức độ khó của bàn chơi (Dễ, Trung Bình, Khó). Mức độ khó ảnh hưởng đến số lượng ô gợi ý ban đầu. |
|  | 2.3 | Hiển Thị Bảng Sudoku (Display Board) | Giao diện chính hiển thị lưới Sudoku 9x9 (hoặc kích thước tùy chỉnh). Các ô gợi ý được làm nổi bật và không thể chỉnh sửa. Các ô còn lại là ô trống để người chơi điền số. |

1.2 Chức năng tương tác của người chơi (Hỗ trợ người chơi thao tác với bàn Sudoku trong quá trình giải đố.)

| **STT** | **Tên Chức Năng** | **Mô Tả** |
| --- | --- | --- |
| 2.2.1 | Chọn Ô (Select Cell) | Cho phép người chơi chọn ô bằng chuột hoặc phím điều hướng. Ô được chọn sẽ nổi bật. |
| 2.2.2 | Nhập Số (Input Number) | Người chơi có thể nhập số (1–9) vào ô trống bằng bàn phím hoặc bảng số ảo. |
| 2.2.3 | Xóa Số (Delete/Clear Number) | Cho phép xóa số đã nhập trong ô trống. Không áp dụng cho ô gợi ý. |
| 2.2.4 | Ghi Chú / Nháp (Pencil Marks / Notes) | Cho phép ghi chú các số khả dĩ trong ô trống (dạng số nhỏ trong ô). |

1.3 Chức năng hỗ trợ và kiểm soát ván chơi (Cung cấp công cụ hỗ trợ người chơi trong quá trình chơi, kiểm tra sai sót, gợi ý, hoặc quản lý thời gian và trạng thái bàn chơi).

| **STT** | **Tên Chức Năng** | **Mô Tả** |
| --- | --- | --- |
| 2.3.1 | Kiểm Tra Lỗi (Validate / Check for Errors) | Kiểm tra lỗi vi phạm luật Sudoku và thông báo cho người chơi. |
| 2.3.2 | Gợi Ý (Hint) | Cung cấp nước đi đúng cho một ô trống. Có thể giới hạn số lần dùng. |
| 2.3.3 | Đặt Lại Bàn Chơi (Reset Board) | Xóa toàn bộ dữ liệu người chơi nhập và khôi phục về trạng thái ban đầu. |
| 2.3.4 | Giải Đố (Solve Puzzle) | Hiển thị toàn bộ lời giải của bàn Sudoku hiện tại. |
| 2.3.5 | Hiển Thị Thời Gian (Timer) | Đồng hồ đếm thời gian chơi, hỗ trợ theo dõi và tăng tính thử thách. |

2. Tổng quan về Game Studoku

2.1 Luật chơi chi tiết

* Mỗi hàng ngang phải chứa đủ các số từ 1 đến 9, không trùng số và không cần phải đúng thứ tự.
* Mỗi hàng dọc cũng phải đảm bảo có đủ các số từ 1 đến 9, không trùng số và không cần phải theo thứ tự.
* Mỗi vùng 3×3 cũng phải chứa đủ các số từ 1 đến 9 và không được trùng số nào trong cùng một vùng.

2.1.2 Các thuật toán liên quan

2.1.2.1 Thuật toán Backtracking

- Thuật toán Backtracking kết hợp với việc xóa ô ngẫu nhiên để sinh bàn cờ.  
+ Bước 1: Tạo một bàn cờ Sudoku hoàn chỉnh:  
Khởi tạo một bảng 9x9 rỗng.  
Sử dụng thuật toán Backtracking (sẽ được mô tả chi tiết hơn ở mục Thuật toán Giải Sudoku) để điền đầy đủ các số vào bảng sao cho bảng đó là một lời giải Sudoku hợp lệ. Để tăng tính ngẫu nhiên, thứ tự thử các số (1-9) cho mỗi ô có thể được xáo trộn, hoặc sau khi có một bảng hoàn chỉnh, có thể thực hiện các phép biến đổi (hoán vị hàng/cột trong cùng một khối 3x3, hoán vị các khối hàng/cột) để tạo ra nhiều bảng khác nhau.  
+ Bước 2: Xóa các ô để tạo thành câu đố:  
Sau khi có một bàn cờ đã giải hoàn chỉnh, tiến hành xóa ngẫu nhiên các ô.  
Tạo một danh sách các tọa độ của tất cả 81 ô, sau đó xáo trộn (shuffle) danh sách này.  
Lần lượt duyệt qua danh sách các ô đã xáo trộn. Với mỗi ô, thử xóa giá trị trong ô đó.  
Kiểm tra tính duy nhất của nghiệm: Sau khi xóa một ô, sử dụng thuật toán giải Sudoku (đã được sửa đổi để đếm số lượng nghiệm) để kiểm tra xem bàn cờ hiện tại còn có nghiệm duy nhất hay không.  
- Nếu bàn cờ vẫn còn nghiệm duy nhất sau khi xóa ô, thì việc xóa ô đó được chấp nhận. Ngược lại, nếu bàn cờ không còn nghiệm hoặc có nhiều hơn một nghiệm, thì giá trị của ô đó được khôi phục lại.  
- Quá trình này lặp lại cho đến khi đạt được số lượng ô trống mong muốn (tương ứng với độ khó) hoặc không thể xóa thêm ô nào mà vẫn đảm bảo nghiệm duy nhất.  
- Cách đảm bảo bàn cờ có nghiệm duy nhất:  
+ Như đã mô tả ở trên, sau mỗi lần thử xóa một ô khỏi bàn cờ hoàn chỉnh, thuật toán giải Sudoku (có khả năng đếm số nghiệm) sẽ được thực thi trên bàn cờ mới. Chỉ khi thuật toán này xác nhận rằng bàn cờ có chính xác một nghiệm thì việc xóa ô đó mới được giữ lại. Nếu số nghiệm là 0 (không giải được) hoặc lớn hơn 1 (nhiều nghiệm), ô đó sẽ được phục hồi giá trị ban đầu.  
- Cách tạo độ khó khác nhau:  
+ Dựa vào số lượng ô đã cho (givens) / ô trống: Đây là phương pháp chính em sử dụng.  
+ Dễ: Số lượng ô trống ít hơn (ví dụ: 30-40 ô trống, tức là 41-51 ô đã cho).  
+ Trung bình: Số lượng ô trống ở mức vừa phải (ví dụ: 45-50 ô trống).  
+ Khó: Số lượng ô trống nhiều hơn (ví dụ: 50-55 ô trống, hoặc thậm chí hơn).

2.1.2.2 Thuật toán giải Sudoku (Solving Algorithm)

- Thuật toán này không chỉ dùng để giải một bàn Sudoku bất kỳ mà còn là công cụ quan trọng trong việc sinh bàn cờ và cung cấp chức năng gợi ý.  
- Mô tả thuật toán sử dụng:  
+ Đã triển khai thuật toán Backtracking (Quay lui) để giải Sudoku. Đây là một thuật toán tìm kiếm theo chiều sâu (Depth-First Search).  
- Nguyên lý hoạt động:  
+ Tìm một ô trống tiếp theo trên bàn cờ (thường duyệt từ trái sang phải, từ trên xuống dưới).   
+ Nếu không còn ô trống nào, nghĩa là bàn cờ đã được giải thành công. Trả về true.  
+ Với ô trống hiện tại, thử lần lượt các số từ 1 đến 9:  
a. Nếu một số k là hợp lệ để điền vào ô đó (kiểm tra theo luật Sudoku: không trùng hàng, không trùng cột, không trùng trong khối 3x3 chứa ô đó).  
b. Điền số k vào ô trống.  
c. Gọi đệ quy hàm giải Sudoku cho trạng thái bàn cờ mới.  
d. Nếu lời gọi đệ quy trả về true (nghĩa là tìm được lời giải từ bước này), thì trả về true.  
e. Nếu lời gọi đệ quy trả về false (nghĩa là việc điền số k không dẫn đến lời giải), xóa số k khỏi ô đó (quay lui – backtrack) và thử số tiếp theo.  
- Nếu đã thử tất cả các số từ 1 đến 9 mà không số nào dẫn đến lời giải, trả về false (báo hiệu nhánh tìm kiếm này thất bại).  
- Ứng dụng của thuật toán:  
+ Kiểm tra tính duy nhất của nghiệm: Để kiểm tra tính duy nhất, thuật toán Backtracking được sửa đổi một chút. Thay vì dừng lại và trả về true ngay khi tìm thấy nghiệm đầu tiên, nó sẽ tiếp tục tìm kiếm để đếm tổng số nghiệm có thể có. Một biến đếm solutionCount được sử dụng. Mỗi khi tìm thấy một nghiệm hoàn chỉnh, solutionCount tăng lên và thuật toán tiếp tục quay lui để tìm các nghiệm khác. Sau khi duyệt hết không gian tìm kiếm, nếu solutionCount == 1, bàn cờ có nghiệm duy nhất.  
+ Chức năng "Gợi ý" (Hint): Khi người dùng yêu cầu gợi ý, thuật toán giải Sudoku có thể được chạy trên bàn cờ hiện tại. Số đầu tiên mà thuật toán điền vào một ô trống hợp lệ có thể được hiển thị cho người dùng như một gợi ý. Hoặc, giải hoàn toàn bàn cờ (một cách bí mật) rồi tiết lộ giá trị của một ô trống bất kỳ mà người dùng chưa điền.

2.1.2.3: Thuật toán kiểm tra tính hợp lệ (Validation Algorithm)

- Thuật toán này dùng để xác minh xem một nước đi của người chơi có tuân thủ luật Sudoku hay không, hoặc một bàn cờ đã hoàn thành có đúng không.  
- Cách kiểm tra một nước đi (khi người chơi nhập số val vào ô (row, col)):  
- Kiểm tra hàng: Duyệt qua tất cả các ô trong hàng row.

- Nếu có ô (row, c) (với c != col) mà giá trị của nó bằng val, thì nước đi không hợp lệ.  
- Kiểm tra cột: Duyệt qua tất cả các ô trong cột col. Nếu có ô (r, col) (với r != row) mà giá trị của nó bằng val, thì nước đi không hợp lệ.  
- Kiểm tra khối 3x3:  
+ Xác định tọa độ góc trên bên trái của khối 3x3 chứa ô (row, col).

+ Duyệt qua tất cả 9 ô trong khối 3x3 này (từ startRow đến startRow + 2, và từ startCol đến startCol + 2). Nếu có ô (r, c) (khác ô (row, col) đang xét) mà giá trị của nó bằng val, thì nước đi không hợp lệ.  
+ Nếu cả ba kiểm tra trên đều không phát hiện vi phạm, nước đi là hợp lệ.  
- Cách kiểm tra một bàn cờ đã hoàn thành:  
+ Đầu tiên, kiểm tra xem có ô nào còn trống không. Nếu có, bàn cờ chưa hoàn thành.  
+ Sau đó, với mỗi ô đã điền số trên bàn cờ, áp dụng lại thuật toán kiểm tra tính hợp lệ như trên (đảm bảo số đó không trùng trong hàng, cột, và khối 3x3 của nó). Nếu tất cả các ô đều hợp lệ, bàn cờ hoàn thành đúng.  
+ Cách khác hiệu quả hơn: Với mỗi hàng, mỗi cột, và mỗi khối 3x3, kiểm tra xem nó có chứa đủ các số từ 1 đến 9 và không có số nào lặp lại hay không (ví dụ, bằng cách dùng một set hoặc mảng boolean để đánh dấu).

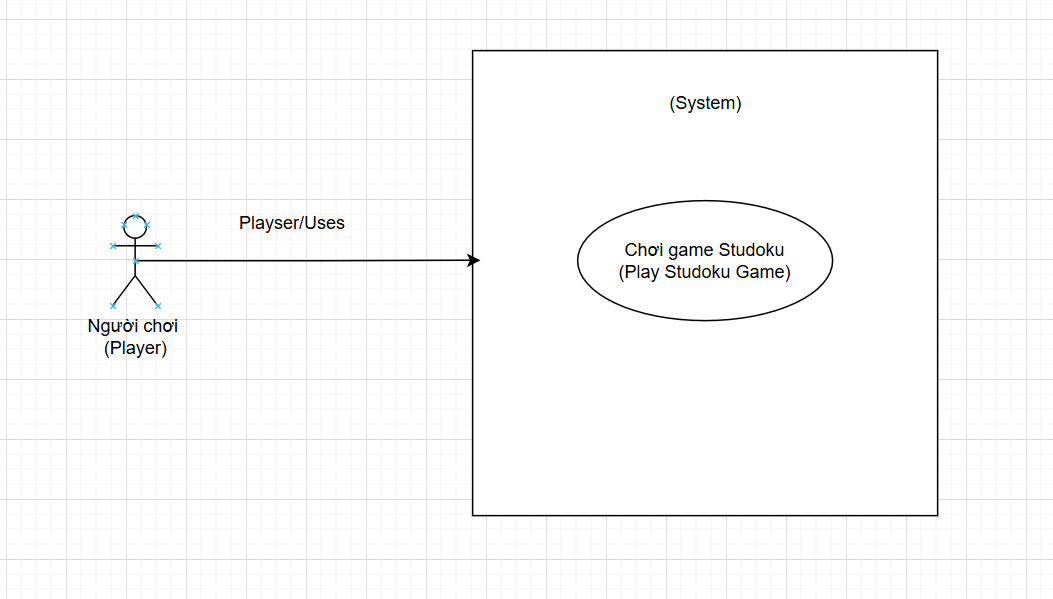
2.1.2.4: Môi trường thiết kế và xây dựng.

- Python (version 3)

- Đảm bảo hoạt động tốt trên các trình duyệt phổ biến như: Chrome, Firefox, Safari, Internet Explorer.

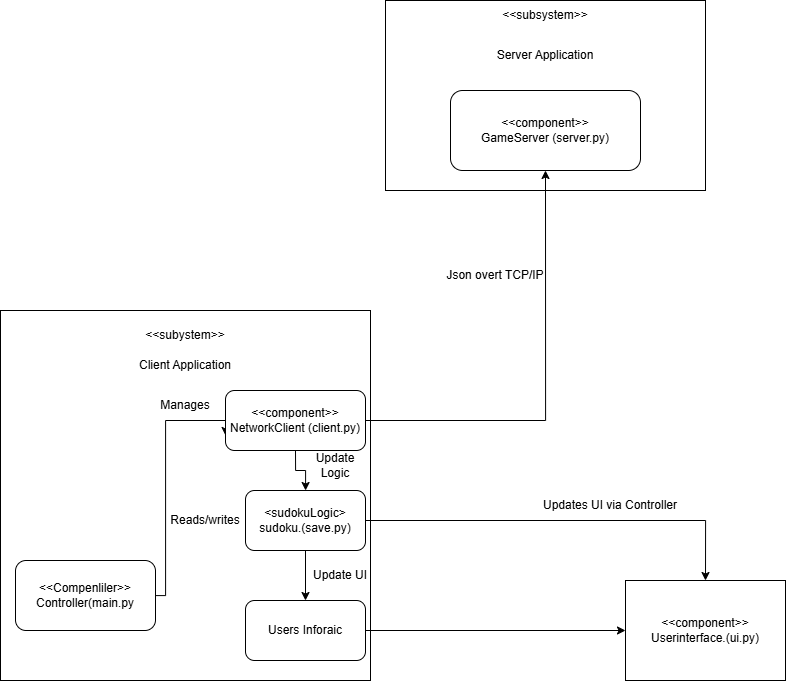
- Đảm bảo hoạt động tốt trên nhiều loại thiết bị khác nhau: Computer, Smartphone, IPAD

# III. Phân loại người dùng

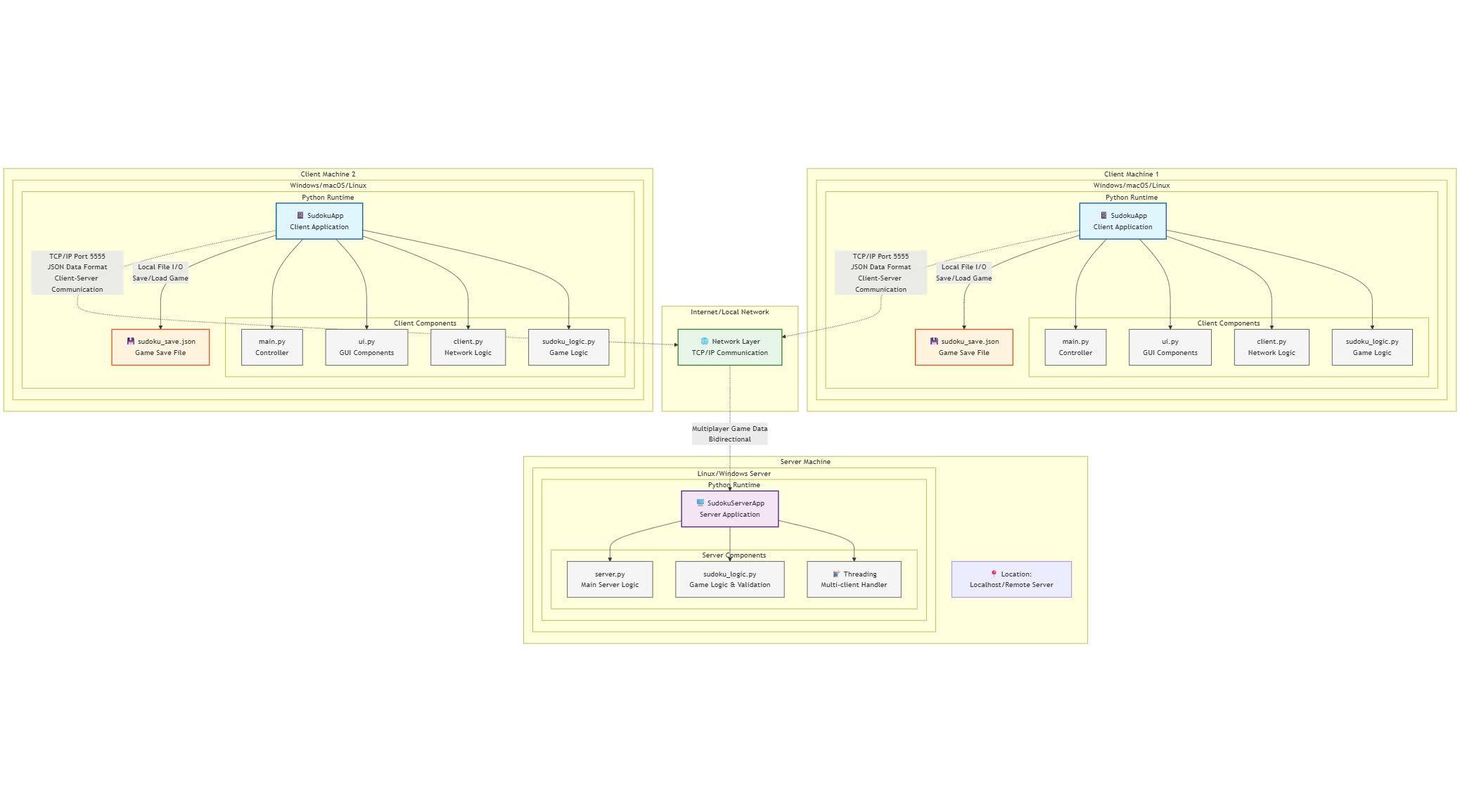
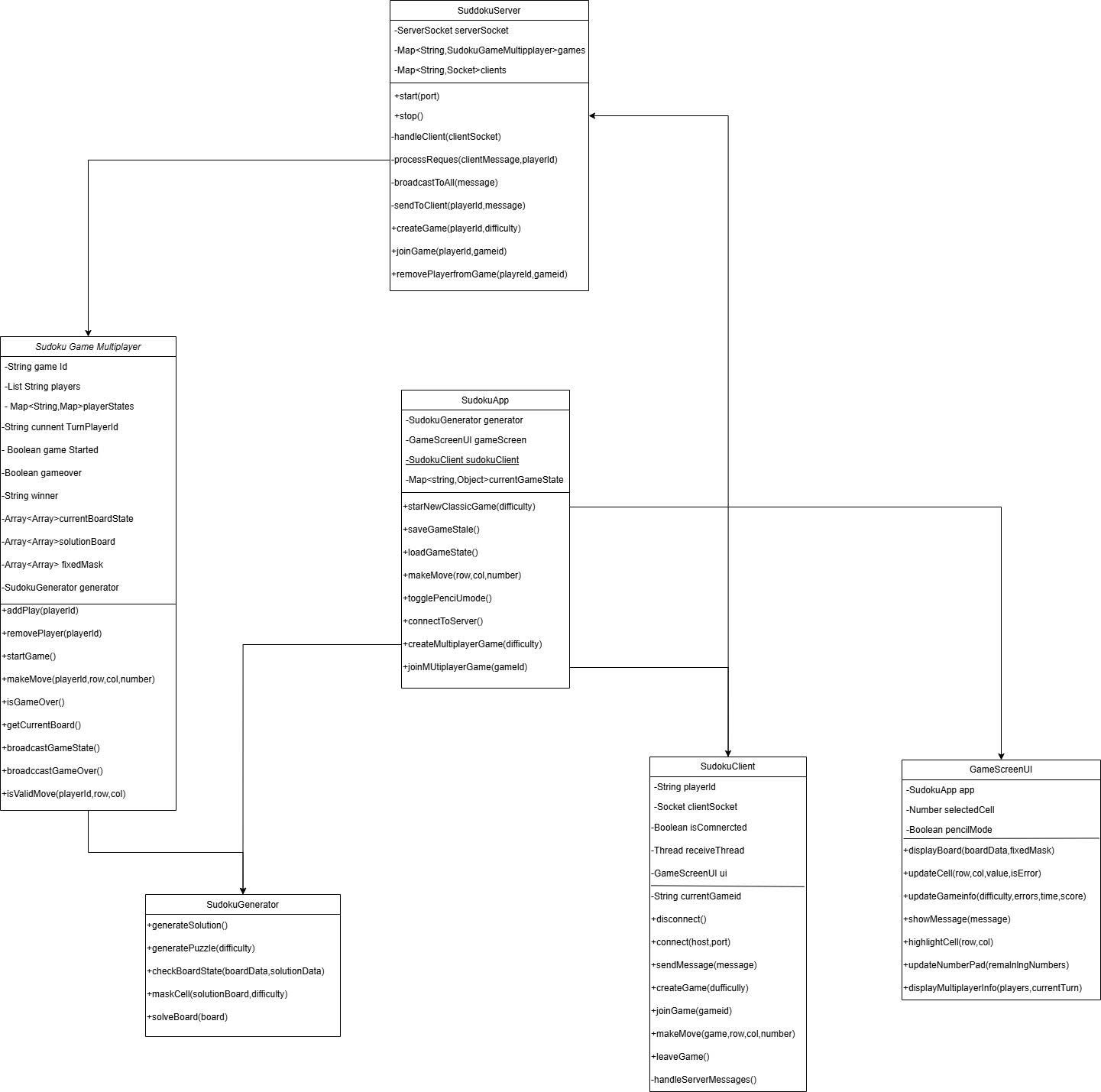
* Các đối tượng tương tác với hệ thống gồm: User

# IV. Kiến trúc hệ thống

## 4.1. kiến trúc tổng thể của hệ thống



Hình 1: : Component diagram

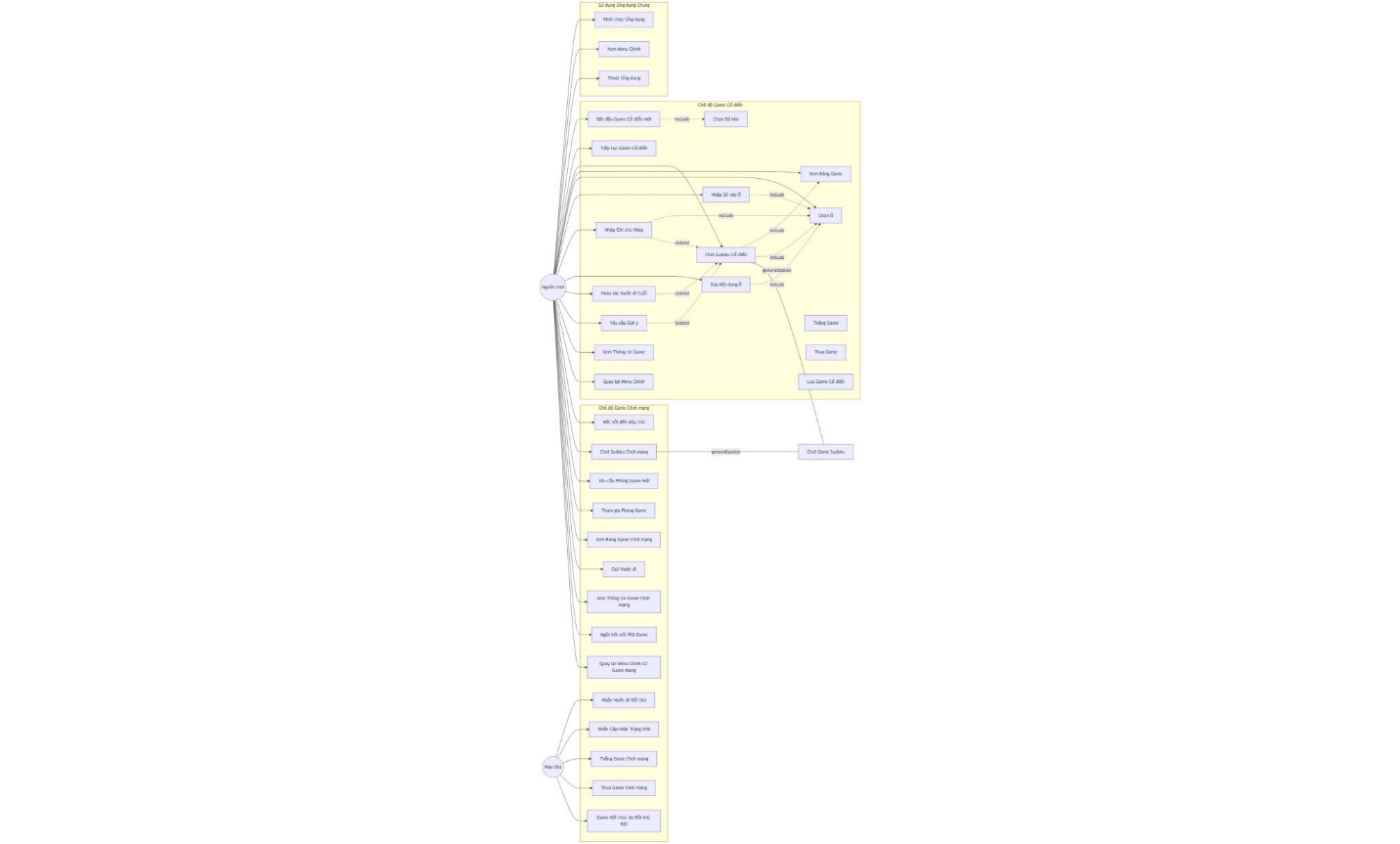


Hình2: Deployment diagram

Hình 2: class diagram

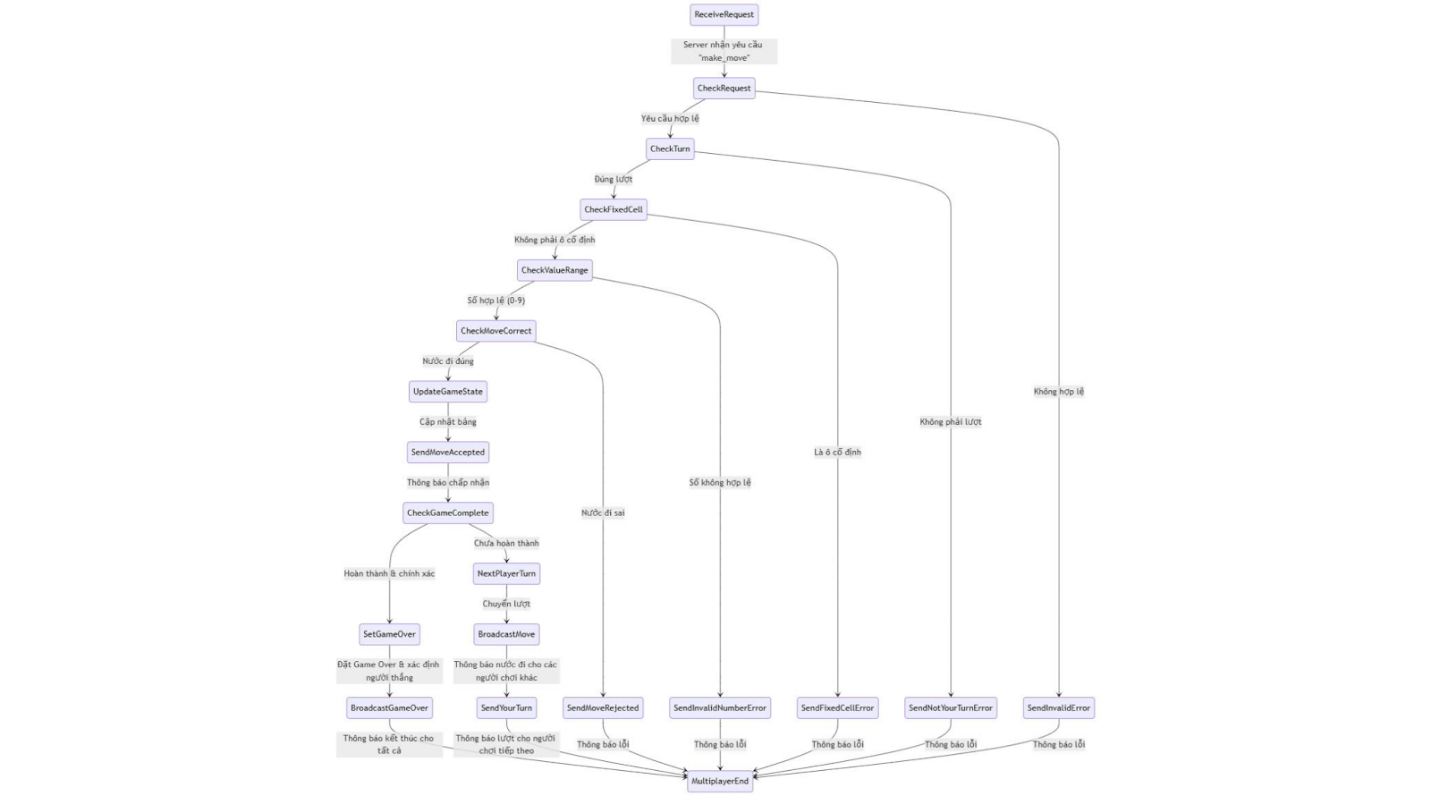
## 4.2. Chi tiết các thành phần

- Biểu đồ use case tổng quát của hệ thống:



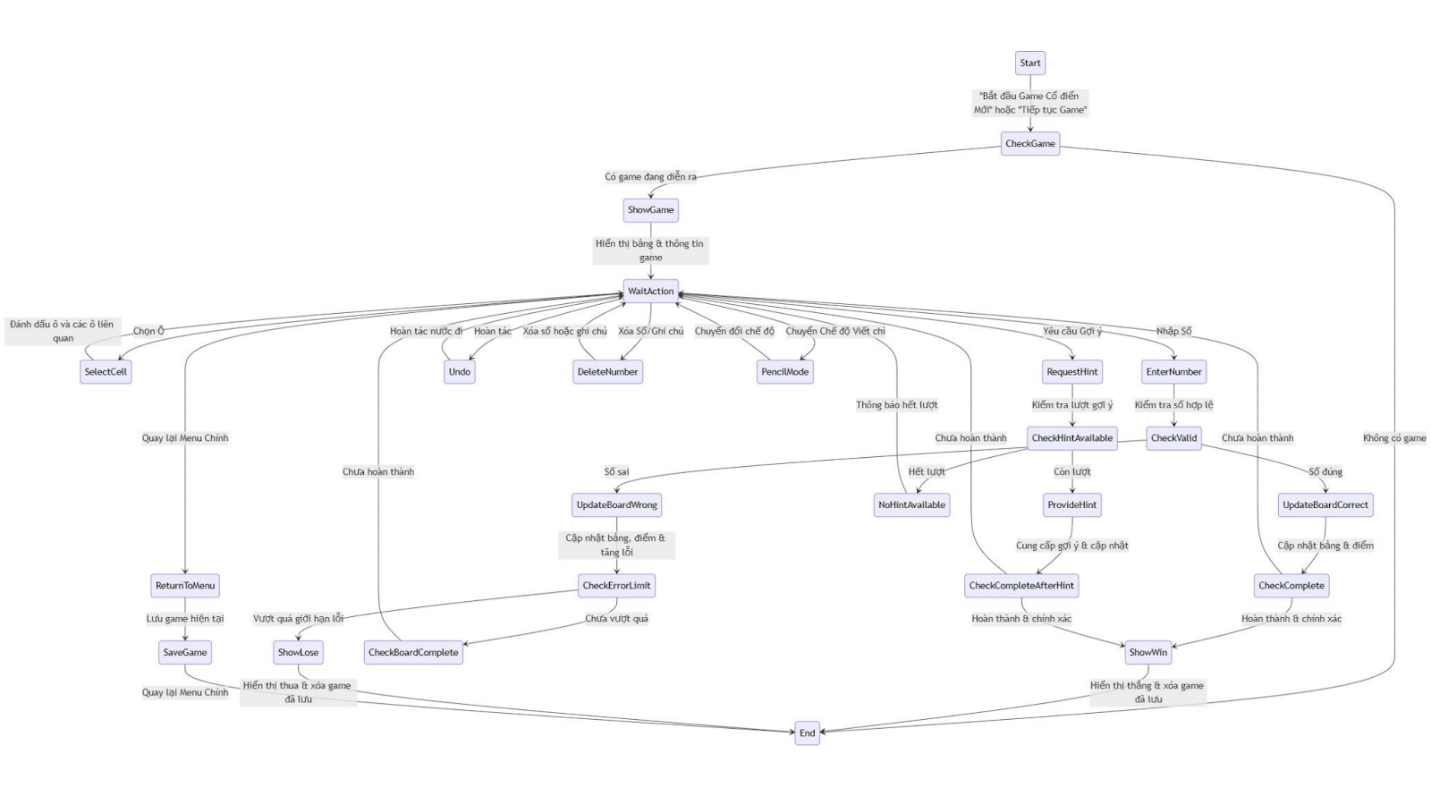
Hình 3: Use case

4.2.1 Sơ đồ Activity diagram xử lý nước đi trong game Multiplayer Server



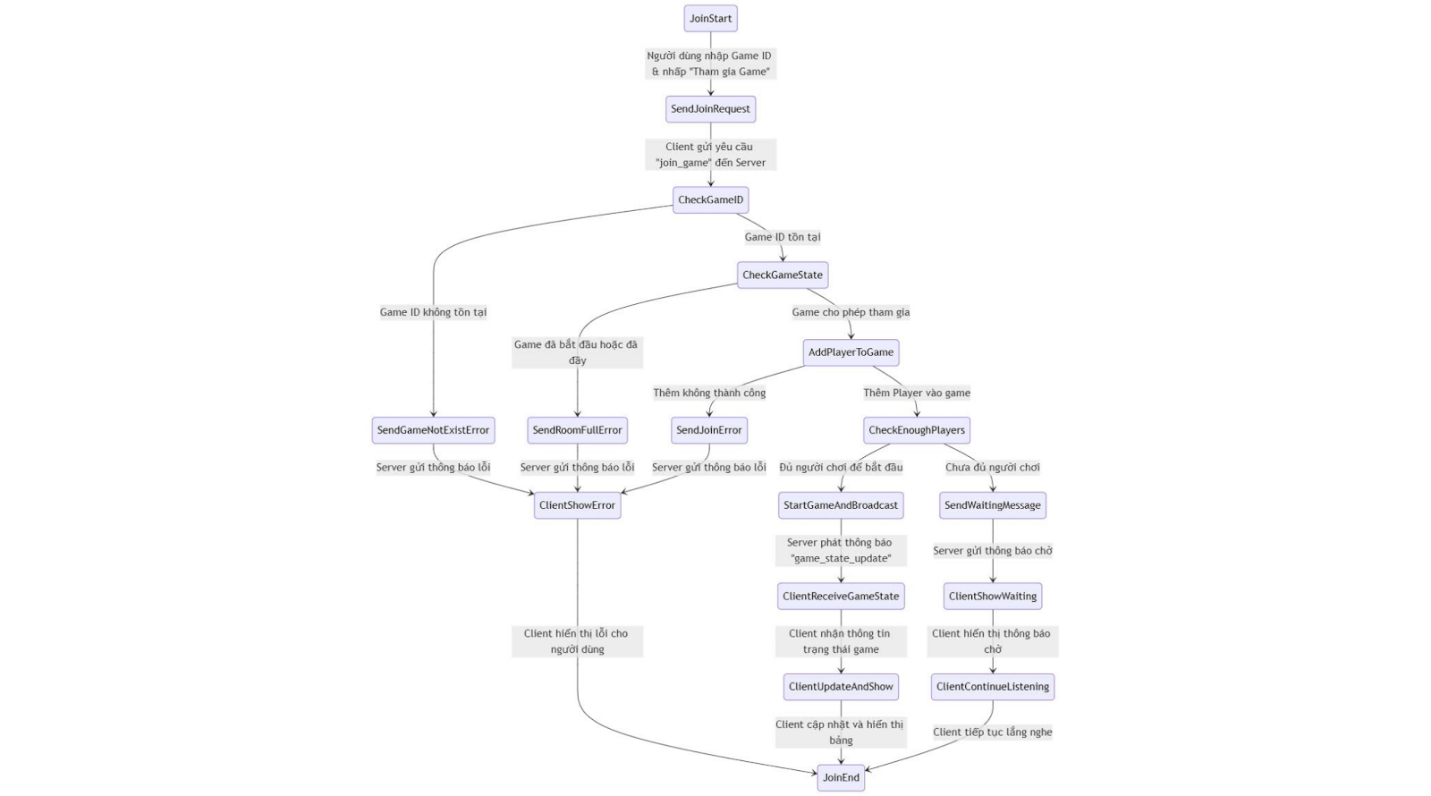
Hình 4: Activity diagram Multiplayer Server

4.2.2 Sơ đồ Activity diagram hoạt động hệ thống Sudoku



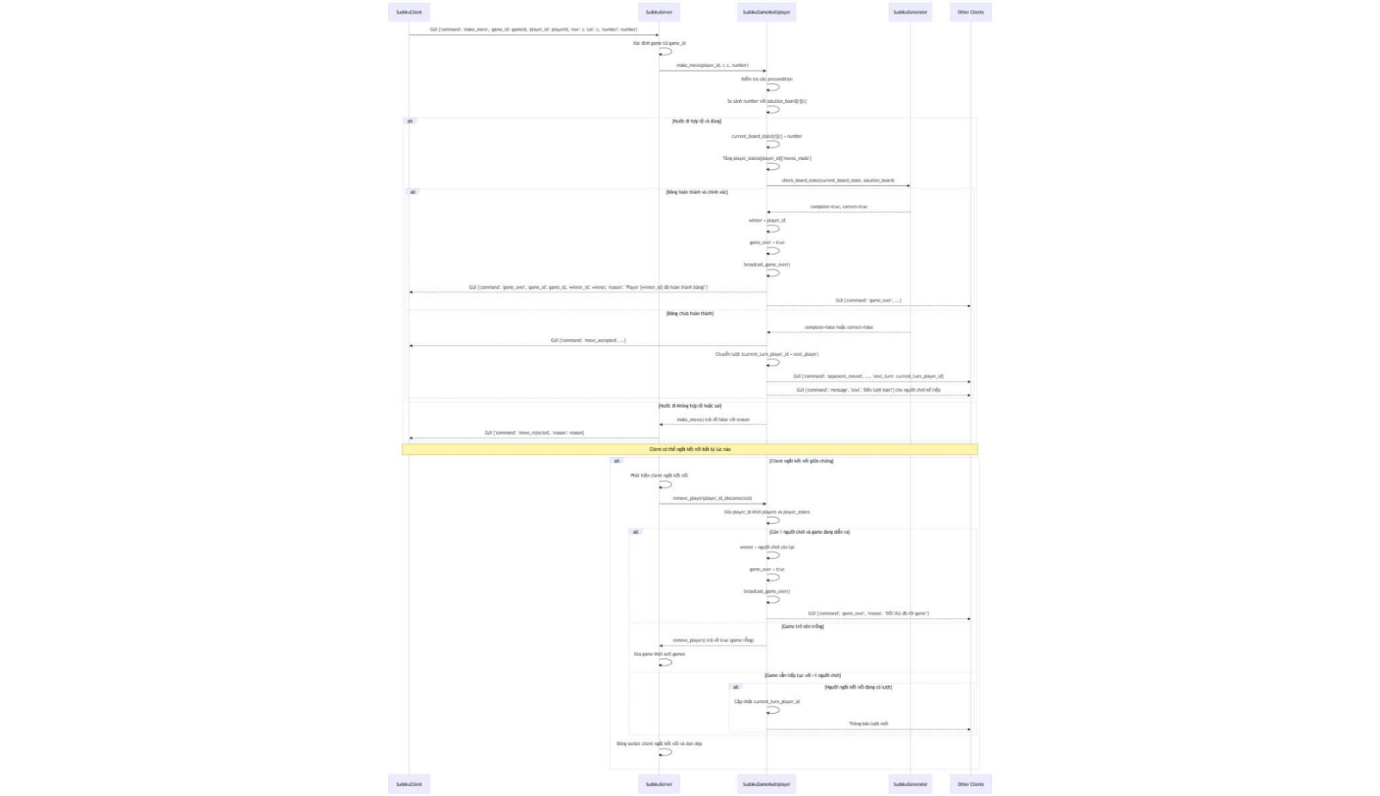
Hình 5: Activity diagram hệ thống Sudoku

4.2.3. Sơ đồ Activitydiagram Client tham gia Game nhiều người chơi



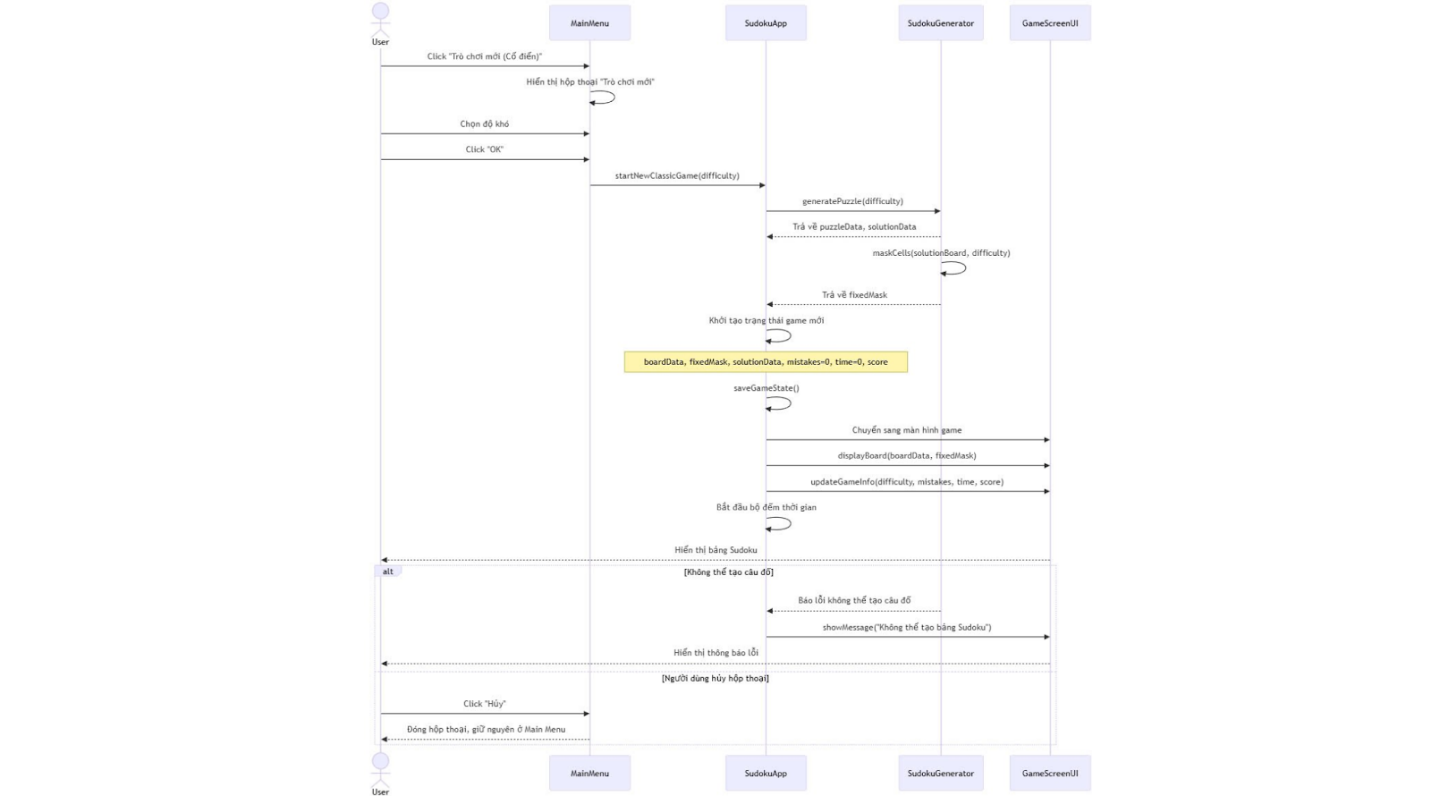
Hình 6: Activity diagram Client

4.2.4 Biểu đồ tuần tự (sequence) Sequence Diagram: Server Process Make Move Request



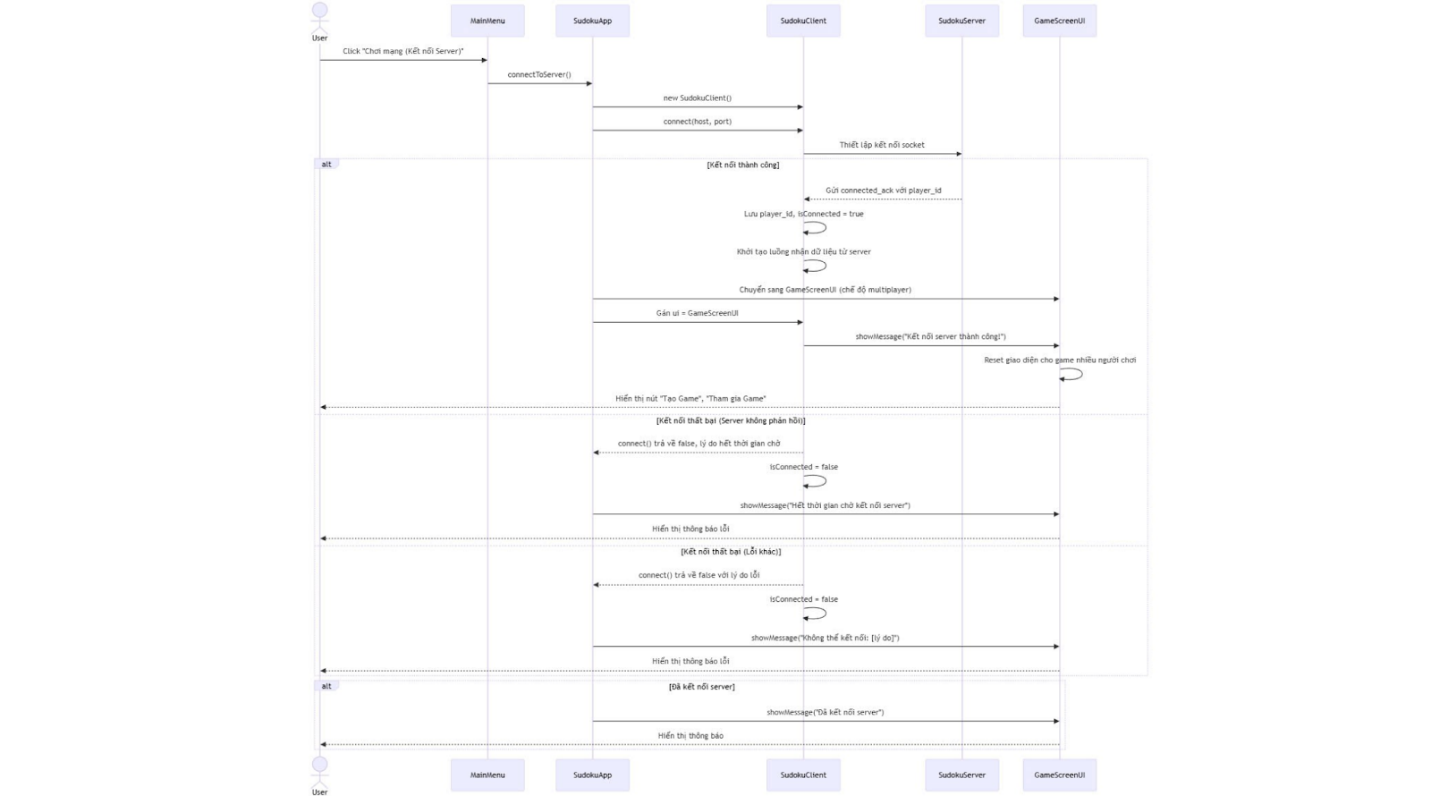
Hình 7: Sequence Diagram: Server Process Make Move Request

4.2.5 Sequence Diagram: Start New Classic Game



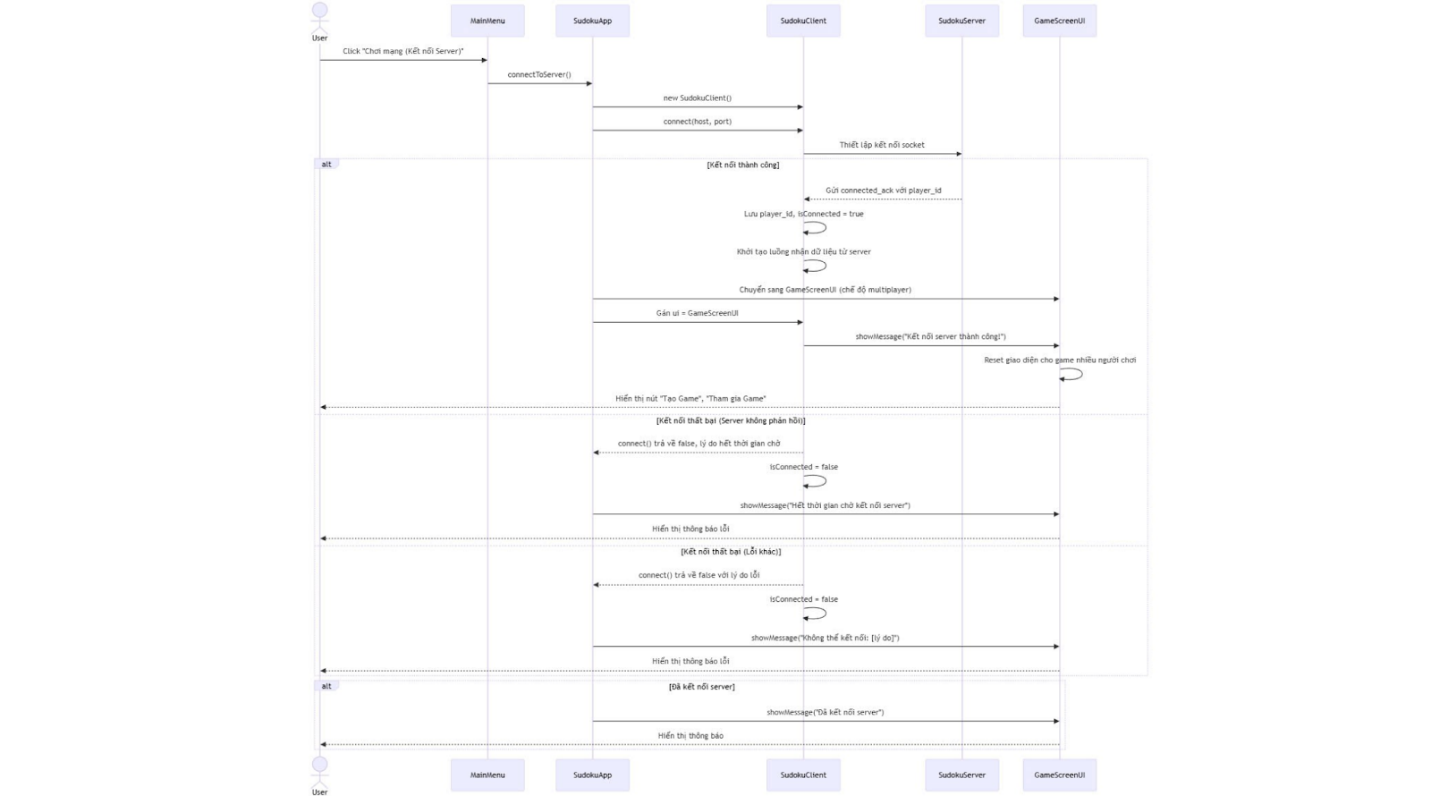
Hình 8: Sequence Diagram: Start New Classic Game

4.2.6 Sequence Diagram: Input Number into Cell (Classic Mode)



Hình 9: Sequence Diagram: Input Number into Cell (Classic Mode)

4.2.7 Sequence Diagram: Connect to Multiplayer Server



Hình 10: Sequence Diagram: Connect to Multiplayer Server

# V. Thực hành và thực nghiệm

# 5.1. Chế độ Chơi Đơn (Classic Mode)

# 5.1.1. Tạo game mới và chọn độ khó

# Người dùng chọn "Trò chơi mới (Cổ điển)" trên MainMenuScreen.

# SudokuApp.show\_new\_classic\_game\_dialog() hiển thị dialog chọn độ khó.

# Người dùng chọn độ khó (Rất Dễ, Dễ, Bình Thường, Khó, Chuyên Gia) và nhấn OK.

# SudokuApp.start\_new\_classic\_game(difficulty) tạo bảng mới, khởi tạo trạng thái game và chuyển sang GameScreenUI.

# 5.1.2. Lưu và tiếp tục game

# Lưu game: Trạng thái game (classic\_game\_state) được lưu vào file sudoku\_save.json sau mỗi thay đổi quan trọng.

# Tiếp tục game: Tải dữ liệu từ file khi khởi động hoặc nhấn "Tiếp tục" trên menu chính.

# 5.1.3. Các tính năng: Nhập số, Viết chì, Xóa, Hoàn tác, Gợi ý

# Nhập số/Viết chì: Hỗ trợ nhập số chính hoặc ghi chú (pencil marks) vào ô được chọn.

# Xóa: Xóa số hoặc ghi chú trong ô.

# Hoàn tác: Khôi phục trạng thái trước đó từ lịch sử (history).

# Gợi ý: Cung cấp số đúng cho một ô trống/sai, giảm điểm và số lượt gợi ý.

# 5.1.4. Tính điểm, thời gian, lỗi

# Điểm: Khởi tạo 1500, tăng/giảm dựa trên hành động người chơi (nhập đúng, sai, dùng gợi ý, thời gian).

# Thời gian: Đếm ngược mỗi giây, giảm điểm sau mỗi 10 giây.

# Lỗi: Giới hạn lỗi dựa trên độ khó, tăng khi nhập sai số.

# 5.1.5. Kiểm tra thắng/thua

# Thắng: Bảng hoàn chỉnh và đúng với lời giải.

# Thua: Vượt quá số lỗi tối đa cho phép.

# 5.2. Chế độ Chơi Mạng (Multiplayer Mode)

# 5.2.1. Kết nối Client - Server

# Server: Chạy server.py, lắng nghe kết nối, tạo luồng riêng cho mỗi client.

# Client: Kết nối qua SudokuClient.connect(), khởi tạo luồng nhận dữ liệu.

# 5.2.2. Tạo phòng chơi và tham gia phòng chơi

# Người chơi tạo phòng mới hoặc tham gia phòng bằng cách nhập Game ID.

# 5.2.3. Đồng bộ trạng thái game

# Server gửi trạng thái game (game\_state\_update) cho client sau mỗi thay đổi.

# Client cập nhật giao diện dựa trên dữ liệu từ server.

# 5.2.4. Xử lý nước đi và lượt chơi

# Client gửi nước đi, server kiểm tra tính hợp lệ, cập nhật bảng và chuyển lượt.

# 5.2.5. Thông báo thắng/thua/lỗi

# Server thông báo kết quả game hoặc lỗi qua JSON.

# 5.3. Giao diện người dùng

# Giao diện gồm:

# MainMenuScreen: Nút chức năng lớn, rõ ràng.

# GameScreenUI: Lưới Sudoku, bảng số, nút chức năng, thông tin game, khu vực multiplayer.

# 

# VI. Kết quả và đánh giá

## 6.1 Kinh nghiệm rút ra

- Về lập trình: Kỹ năng tổ chức mã nguồn Python theo hướng đối tượng đã được cải thiện, ví dụ: như chia game thành các lớp Board, Cell, GameLogic."

- Về thuật toán: Hiểu sâu hơn và tự tay triển khai được thuật toán Backtracking cho cả việc giải Sudoku và hỗ trợ sinh bàn cờ. Nhận thấy được tầm quan trọng của việc kiểm tra điều kiện ràng buộc (constraints) trong Backtracking.

## 6.2 Kết quả đạt được

- Hoàn thiện game Sudoku với chế độ chơi đơn đầy đủ và chế độ chơi mạng cơ bản.

- Giao diện thân thiện, mã nguồn tổ chức rõ ràng theo mô-đun.

6.3 Kết quả chưa đạt được

- Những hạn chế của game: Giao diện người dùng còn đơn giản, chưa có nhiều tùy biến về màu sắc, font chữ hay chủ đề.

- Thuật toán sinh bàn cờ hiện tại chủ yếu dựa vào số lượng ô trống để xác định độ khó, chưa phân tích sâu về các kỹ thuật logic cần thiết để giải, do đó độ khó có thể chưa thực sự chuẩn xác và đa dạng.

## 6.4 Hướng phát triển

- Thêm nhiều chế độ chơi

- Cải thiện trải nghiệm người dung

- Cải thiện thuật toán Sudoku, thêm sảnh chờ, chat, bảng xếp hạng.

# VII. Tài liệu tham khảo

Python Documentation: https://docs.python.org/3/

Sudoku Algorithms: https://en.wikipedia.org/wiki/Sudoku\_solving\_algorithms

Tkinter Tutorials: https://www.pythontutorial.net/tkinter/

Socket Programming: https://realpython.com/python-sockets/