TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CUỐI KỲ MÔN

CHUYÊN ĐỀ CÔNG NGHỆ PHẦN MỀM

GAME ONLINE DÙNG HTML5 KẾT HỢP ASP.NET

Giảng viên hướng dẫn: **THẦY VŨ ĐÌNH HỒNG**

Sinh viên thực hiện: **TÔ VĨNH KHANG - 51800408**

BÙI QUANG KHẢI - 51800785

DU THUẬN LONG - 51800429

Lớp: **18050203**

Khóa: **22**

Nhóm Lớp: **04**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2022

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CUỐI KỲ MÔN

CHUYÊN ĐỀ CÔNG NGHỆ PHẦN MỀM

GAME ONLINE DÙNG HTML5 KẾT HỢP ASP.NET

Giảng viên hướng dẫn: **THẦY VŨ ĐÌNH HỒNG**

Sinh viên thực hiện: **TÔ VĨNH KHANG - 51800408**

BÙI QUANG KHẢI - 51800785

DU THUẬN LONG - 51800429

Lớp: **18050203**

Khóa: **22**

Nhóm Lớp: **04**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2022

LỜI CẢM ƠN

Chuyên Đề Công Nghệ Phần Mềm là một môn học rất hay và bổ ích. Chúng em chân thành cảm ơn Khoa Công nghệ thông tin và Trường Đại học Tôn Đức Thắng đã cho chúng em được học tập trong suốt thời gian qua. Chân thành cảm ơn Thầy Vũ Đình Hồng đã hướng dẫn và tạo dựng những điều kiện tốt nhất để chúng em hoàn thành đồ án cuối kỳ với đề tài "Game online dùng HTML5 kết hợp ASP.NET". Giúp chúng em có thêm kiến thức về những giai đoạn từ lên ý tưởng đến hiện thực game, hiểu biết khái quát hơn về cách sử dụng nền tảng ASP.NET để xây dựng một game 2D thực tế và những vấn đề, khó khăn gặp phải trong quá trình phát triển.

Trong quá trình thực hiện, nhóm vẫn khó tránh khỏi những sai sót không mong muốn, kính mong thầy có thể góp ý và giúp đỡ chúng em. Nhóm xin chân thành cảm ơn thầy.

ĐỒ ÁN CUỐI KỲ NÀY ĐƯỢC HOÀN THÀNH TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Chúng tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của thầy Vũ Đình Hồng. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án này còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào chúng tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình. Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 09 tháng 01 năm 2022

Tác giả

(ký tên và ghi rõ họ tên)

Tô Vĩnh Khang

(Trưởng nhóm)

Bùi Quang Khải

Du Thuận Long

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

_____

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm

(ký và ghi họ tên)

Phần đánh giá của GV chấm bài

_____

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm

(ký và ghi họ tên)

TÓM TẮT

Trong xã hội ngày càng phát triển với sự bùng phát mạnh mẽ từ các dịch vụ giải trí, nhu cầu về một game thú vị mới lạ, mang lại cảm giác thử thách người chơi với lối chơi vui nhộn, lôi cuốn ngày càng được chú trọng. Và từ đó các game 2D ra đời. Để đáp ứng được nhu cầu trong bối cảnh thị trường hiện nay, nhóm đã triển khai thực hiện dự án Web Game online với góc nhìn từ trên xuống mang tên Journey - Hành trình Vui nhộn. Game được triển khai với mục tiêu giúp người chơi xả stress sau những ngày làm việc căng thẳng, tạo cảm giác muốn chinh phục trò chơi, thử thách bản thân trong việc thực hiện các nhiệm vụ ở từng căn phòng khác biệt.

Journey lấy một phần nhỏ ý tưởng từ một tựa game khá nổi tiếng hiện nay tên là Among Us. Ý tưởng nhân vật làm từng task trong game Among Us sẽ được chuyển thể thành từng game mini (ý tưởng game trong game). Game cho người dùng chơi trên trình duyệt mà không cần cài thêm plugin. Các người chơi sẽ gặp nhau ở Sảnh khi vừa vào game, họ sẽ lần lượt vào từng căn phòng bí mật để thực hiện các game mini khác nhau (Cosmic Snake, Galaxy 2048, Spaceship Flappy và Robotic Dinosaur). Thể loại game trong game này khá vui nhộn và gần gũi nhưng cũng tiêu tốn thời gian người chơi để thực hiện nó. Khi người chơi kết thúc một game mini, họ sẽ được thưởng kinh nghiệm và đồng xu. Đối với kinh nghiệm, khi thanh kinh nghiệm đầy thì người chơi sẽ được thăng lên một cấp. Đối với đồng xu, người chơi có thể dùng số xu này để mua thể lực. Số thể lực sẽ bị trừ đi một khoảng nhất định khi người chơi bắt đầu một game mini. Việc này nhằm giúp cho người chơi bảo vệ sức khỏe vì lối chơi này sẽ khiến người chơi bị nghiện và chơi liên tục để chinh phục. Người chơi có thể xem thông tin của bản thân cũng như bảng xếp hạng các người chơi khác theo cấp bậc. Hệ thống sẽ thống kê tất cả người chơi, cho họ biết họ đang xếp ở vị trí thứ mấy trong bảng xếp hạng theo cấp bậc. Trong quá trình chơi, các người chơi có thể giao tiếp với nhau qua khung chat. Các thiết lập của hệ thống như điều chỉnh âm thanh và độ sáng của trò chơi cũng được hỗ trợ.

Về công nghệ, nhóm sử dụng HTML, CSS, Javascript kết hợp nền tảng ASP.NET. Thư viện SignalR được sử dụng để phục vụ cho việc truyền tải dữ liệu không đồng bộ, giúp cho việc giao tiếp theo thời gian thực giữa client và server. Các

chức năng được viết chủ yếu qua ngôn ngữ C# thông qua ASP.NET Core MVC. Sử dụng Docker và Amazon Elastic Compute Cloud (Amazon EC2) trong việc hosting sản phẩm. Game Journey được triển khai với giao diện vui nhộn, thân thiện với người chơi, dễ dàng trong các thao tác di chuyển và phù hợp với mọi lứa tuổi.

Với đề tài Game online HTML5 kết hợp ASP.NET này, nhóm mong muốn áp dụng những kiến thức đã học trong trường cùng với việc tìm hiểu nghiên cứu, phân tích tâm lý người chơi, những kiến thức bên ngoài xã hội để xây dựng game đạt được hiệu quả cao nhất.

MỤC LỤC

DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT

CSDL	Cơ Sở Dữ Liệu
2D	2-Dimensional
NPC	Non Player Character
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
ASP	Active Server Pages
NET	Network Enabled Technologies
SQL	Structured Query Language
ERD	Entity Relationship Diagram
KKL	Khang Khải Long

DANH MỤC CÁC HÌNH VÀ BẢNG

Danh mục Hình

Danh mục Bảng

CHƯƠNG I: GIỚI THIỆU CHUNG

1.1 Giới thiệu đề tài

Trong xã hội ngày càng phát triển với sự bùng phát mạnh mẽ từ các dịch vụ giải trí, nhu cầu về một game thú vị mới lạ, mang lại cảm giác thử thách người chơi với lối chơi vui nhộn, lôi cuốn ngày càng được chú trọng. Và từ đó các game 2D ra đời. Để đáp ứng được nhu cầu trong bối cảnh thị trường hiện nay, nhóm đã triển khai thực hiện dự án Web Game online với góc nhìn từ trên xuống mang tên Journey - Hành trình Vui nhộn. Game được triển khai với mục tiêu giúp người chơi xả stress sau những ngày làm việc căng thẳng, tạo cảm giác muốn chinh phục trò chơi, thử thách bản thân trong việc chơi các game mini ở từng căn phòng khác biệt.

1.2 Lý do và mục đích chọn đề tài

Chúng em chọn đề tài này là vì muốn có thêm kiến thức về xây dựng game online trên nền tảng web bằng việc sử dụng nền tảng ASP.NET để hiện thực một game 2D. Ngoài ra, loại hình game thiên hướng vui nhộn này đang ngày càng được phát triển và có chỗ đứng trong xã hội. Khi mà nhu cầu đòi hỏi về những game mang tính chinh phục, lôi cuốn khi tranh bảng xếp hạng cùng với nhau đồng thời sử dụng lối thiết kế vui nhộn cho trẻ nhỏ có thể chơi được.

1.3 Công nghệ sử dụng

1.3.1 Front-end

1.3.1.1 HTML5

Hình 1. Hình ảnh về các logo HTML5

HTML5 là một ngôn ngữ cấu trúc và trình bày nội dung cho World Wide Web và sẽ là công nghệ cốt lõi của Internet trong tương lai không xa. Mục tiêu cốt lõi khi thiết kế ngôn ngữ là cải thiện khả năng hỗ trợ cho đa phương tiện mới nhất trong khi vẫn giữ nó dễ dàng đọc được bởi con người và luôn hiểu được bởi các thiết bị và các chương trình máy tính. HTML5 vẫn sẽ giữ lại những đặc điểm cơ bản của HTML4 và bổ sung thêm các đặc tả nổi trội của XHTML, DOM cấp 2 và JavaScript. Nó có thêm nhiều tính năng cú pháp mới với các thẻ mới như <video>, <audio> và <canvas>. Được thiết kế để làm cho nó dễ dàng bao quát, xử lý đa phương tiện và nội dung đồ họa trên web mà không cần phải dùng đến quyền sở hữu bổ sung và APIs. Các yếu tố mới khác, chẳng hạn như <section>, <article>, <header> và <nav>, được thiết kế để làm phong phú thêm nội dung ngữ nghĩa của tài liệu. Thuộc tính mới đã được giới thiệu với mục đích tương tự, trong khi một số yếu tố và các thuộc tính đã được loại bỏ. Một số yếu tố, chẳng hạn như <a>, <cite> và <menu> đã được thay đổi, xác định lại hoặc chuẩn hóa.

1.3.1.2 CSS3

Hình 2. Hình ảnh về các logo CSS3

CSS3 là chuẩn mới nhất cho CSS, nó tương thích với chuẩn CSS cũ, hầu hết các trình duyệt đều hỗ trợ các tính năng mới trong CSS3. Một số tính năng mới cho CSS3 là bo trong các đường viền ở 4 góc của box phần tử, bo tròn các góc của ảnh, hỗ trợ nhiều nền trong phần tử, xây dựng phần tử động, đổ bóng cho box phần tử với box-shadow, tạo ra gradient với hai kiểu linear và radial, biến đổi phần tử bằng cách quay hoặc dịch chuyển và tạo đối tượng động animation.

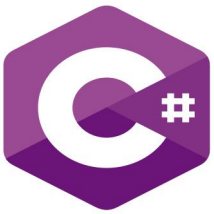
1.3.1.3 JavaScript

Hình 3. Hình ảnh về các logo JavaScript

JavaScript là một ngôn ngữ lập trình thông dịch, dựa trên nguyên mẫu với cú pháp phát triển từ C. Giống như C, JavaScript có khái niệm từ khóa, do đó, JavaScript gần như không thể được mở rộng. Trên trình duyệt, rất nhiều trang web sử dụng JavaScript để thiết kế trang web động và một số hiệu ứng hình ảnh thông qua DOM.

1.3.2 Back-end

1.3.2.1 CSharp - ASP.NET



Hình 4. Hình ảnh về logo CSharp

CSharp (C#) là một ngôn ngữ lập trình hướng đối tượng đa năng và cũng là phần khởi đầu cho kế hoạch .NET.

Hình 5. Hình ảnh về logo ASP.NET

ASP.NET là một nền tảng dành cho phát triển web, cho phép những người lập trình tạo ra những trang web động, những ứng dụng web và những dịch vụ web. Nó được thiết kế để tương thích với giao thức HTTP. Thông thường, ngôn ngữ C# được sử dụng nhiều nhất trong ASP.NET.

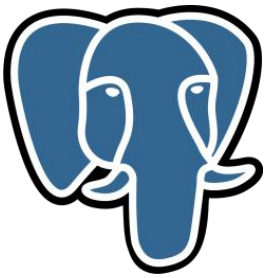
1.3.2.2 SignalR

SignalR là thư viện phần mềm mã nguồn mở và miễn phí dành cho ASP.NET, cho phép mã máy chủ gửi thông báo không đồng bộ đến các ứng dụng web phía máy khách. Nó tận dụng WebSocket, một API HTML5 cho phép giao tiếp hai chiều giữa trình duyệt và máy chủ. SignalR sẽ sử dụng WebSockets dưới vỏ bọc khi có sẵn và quay trở lại các kỹ thuật và công nghệ khác khi không có, trong khi mã ứng dụng vẫn giữ nguyên.

Hình 6. Hình ảnh về logo SignalR

1.3.2.3 PostgreSQL

PostgreSQL là một hệ quản trị cơ sở dữ liệu quan hệ, hỗ trợ mạnh trong việc lưu trữ dữ liệu không gian. Nó mở đường cho nhiều khái niệm quan trọng mà các hệ quản trị dữ liệu thương mại rất lâu sau mới có.



Hình 7. Hình ảnh về logo PostgreSQL

1.3.3 Hosting

1.3.3.1 Docker

Docker là một mã nguồn mở giúp tự động triển khai các ứng dụng vào trong các container ảo hóa. Nó cung cấp một lớp trừu tượng và tự động ảo hóa dựa trên việc sử dụng những tài nguyên cô lập của Linux để cho phép các container chạy độc lập bên trong một thực thể Linux. Các thay đổi được lưu trữ trong các "docker image", các lớp tệp hệ thống được tạo ra và lưu lại dựa theo từng layer, giúp cho "docker image" giảm dung lượng đáng kể so với máy ảo. Docker có thể được tích hợp vào Amazon Web Services, Google Cloud Platform, Microsoft Azure,.. Các ứng dụng muốn chạy bằng Docker phải là ứng dụng chạy được trên Linux. Gần đây, Docker có hỗ trợ thêm việc chạy ứng dụng Windows trong các Windows container.

Hình 8. Hình ảnh về logo Docker

1.3.3.2 Amazon EC2

Amazon Elastic Compute Cloud hay còn gọi là Amazon EC2 là một phần của Amazon Web Services - AWS. Nó cho phép người dùng thuê máy tính ảo trên đó để chạy các ứng dụng máy tính của mình, khuyến khích triển khai các ứng dụng có thể mở rộng bằng cách cung cấp một dịch vụ web mà qua đó người dùng có thể khởi động Amazon Machine Image - AMI để cấu hình một máy ảo và có thể chứa bất kỳ phần mềm nào mong muốn. Người dùng có thể tạo, khởi chạy và kết thúc máy chủ. Ngoài ra, nó còn cung cấp cho người dùng quyền kiểm soát vị trí địa lý của các phiên bản cho phép tối ưu hóa độ trễ.

Hình 9. Hình ảnh về logo Amazon EC2

1.4 Thông tin về game

1.4.1 Tên game

Journey - Hành trình Vui nhộn

1.4.2 Bản quyền

Bản quyền thuộc về KKL Software Team

1.4.3 Phiên bản

Phiên bản 1.0 - Ban hành ngày 09/01/2022

Ngôn ngữ: Tiếng Anh

Tác giả: KKL Software Team

1.4.4 Ý tưởng game

Journey lấy một phần nhỏ ý tưởng từ một tựa game khá nổi tiếng hiện nay tên là Among Us. Game cho người dùng chơi trên trình duyệt mà không cần cài thêm plugin. Các người chơi sẽ gặp nhau ở Sảnh khi vừa vào game, họ sẽ lần lượt vào từng căn phòng bí mật để thực hiện các game mini khác nhau (ý tưởng game trong game). Vui nhộn và cũng tiêu tốn thời gian người chơi để thực hiện nó. Khi người chơi kết thúc một game mini, họ sẽ được thưởng kinh nghiệm và đồng xu. Đối với kinh nghiệm, khi thanh kinh nghiệm đầy thì người chơi sẽ được thăng lên một cấp. Đối với đồng xu, người chơi có thể dùng số xu này để mua thể lực. Số thể lực sẽ bị trừ đi một khoảng nhất định khi người chơi bắt đầu một game mini. Việc này nhằm giúp cho người chơi bảo vệ sức khỏe vì lối chơi này sẽ khiến người chơi bị nghiện và chơi liên tục để chinh phục. Người chơi có thể xem thông tin của bản thân cũng như bảng xếp hạng các

người chơi khác theo cấp bậc. Hệ thống sẽ thống kê tất cả người chơi, cho họ biết họ đang xếp ở vị trí thứ mấy trong bảng xếp hạng theo cấp bậc. Trong quá trình chơi, các người chơi có thể giao tiếp với nhau qua khung chat. Các thiết lập của hệ thống như điều chỉnh âm thanh và độ sáng của trò chơi cũng được hỗ trợ. Server tự trừ thời gian ngay cả khi người chơi không online.

1.4.5 Thể loại

Vui nhộn, Chinh phục, Thử thách.

1.4.6 Đối tượng nhắm đến

Độ tuổi: 6+

1.4.7 Tóm tắt dòng game

Mở đầu game người chơi sẽ thấy màn hình hiển thị menu chính và chọn lựa chức năng "Play Game" để bắt đầu game, phía server lưu đầy đủ thông tin người chơi để người chơi có thể chơi tiếp game của mình. Game có những thao tác tốn thời gian. Người chơi sẽ bắt đầu game tại vị trí Sảnh và di chuyển lần lượt để tìm đến các cổng của từng căn phòng bí mật bằng phím "E" khi bước tới cổng và chơi các game mini trong từng căn phòng đó. Bật game mini cũng bằng phím "E". Khi bắt đầu 1 lượt chơi game mini, người chơi sẽ bị trừ điểm thể lực và khi hoàn tất 1 lượt chơi game mini đó thì người chơi sẽ được thưởng xu và kinh nghiệm. Người chơi vào cửa hàng thể lực, sử dụng xu để mua lại điểm thể lực. Thanh kinh nghiệm hiển thị theo tỉ lệ phần trăm, khi đầy thì người chơi được thăng cấp. Các thông tin luôn được lưu giữ lại liên tục trên CSDL. Phía góc trái màn hình có một khung chat giúp các người chơi giao tiếp với nhau theo thời gian thực. Đối với nút "Exit Game" người chơi sẽ thoát khỏi game. Người chơi có thể nhấn vào nút hình chiếc "bánh răng cưa" hoặc nhấn nút "ESC" trên bàn phím để mở chức năng Settings. Tại mục Settings sẽ có chức năng "Volume" sẽ giúp người chơi thay đổi âm lượng cũng như chức năng "Brightness" sẽ giúp người chơi thay đổi độ sáng cho phù hợp với sở thích, hiển thị theo tỉ lệ phần trăm. Người chơi có thể xem thông tin về của bản thân bằng cách nhấn vào avatar hoặc phím "I",

có thể xem bảng xếp hạng theo cấp bậc các người chơi khác. Về lại Menu game bằng phím “M”.

1.4.8 Góc nhìn

2D - Góc nhìn từ trên xuống

1.4.9 Phạm vi dự án

1.4.9.1 Số địa điểm

Địa điểm	Mô tả
0	Sảnh (Hall)
1	Cổng 1 (Gateway 1 - Red Room)
2	Cổng 2 (Gateway 2 - Blue Room)
3	Cổng 3 (Gateway 3 - Green Room)
4	Cổng 4 (Gateway 4 - Purple Room)

Bảng 1. Số địa điểm trong game

1.4.9.2 Số cấp bậc

Đối với game này, cấp bậc của người chơi là vô hạn, tùy thuộc vào việc người chơi chơi càng nhiều lượt game mini thì sẽ được thưởng càng nhiều kinh nghiệm và đo đó cấp bậc càng tăng và hạng cũng tăng theo cấp bậc trong bảng xếp hạng. Với số kinh nghiệm được tính theo số phần trăm (từ 0 đến 100%).

1.4.9.3 Số đối tượng

Đối tượng	Mô tả
0	Người hướng dẫn (NPC Guide)
1	Người chơi nam
2	Người chơi nữ

Bảng 2. Số đối tượng trong game

1.4.10 Lối chơi và cơ chế

1.4.10.1 Lối chơi

- Tiến trình: Người chơi sẽ trả một khoảng thể lực để được một lượt chơi game mini. Khi hoàn thành thì người chơi được thưởng kinh nghiệm và xu, khi thanh kinh nghiệm đầy thì người chơi được thăng cấp. Xu sẽ dùng để mua lại điểm thể lực.
- Cấu trúc: Bản đồ game của từng căn phòng sẽ có những màu sắc riêng biệt, từng game mini riêng biệt. Các game mini sẽ được bố trí sẵn trong từng phòng. Người chơi khi tiến đến và có sự kiện va chạm thì cửa sổ game mini sẽ hiển lên.
- Mục tiêu: Ở mỗi game mini, người chơi sẽ thực hiện các quy luật của từng game mini, game mini không có thắng mà chỉ có thử thách để đạt điểm cao nhất trong game mini đó. Và khi xong 1 lượt thì được tính là hoàn thành và hệ thống tự động tính điểm và xu tương ứng.
- Môi trường: Được thiết kế để có thể điều hướng người chơi một cách tự nhiên, giúp người chơi có thể dễ dàng hiểu và nắm bắt được lối chơi của game.

1.4.10.2 Cơ chế

- Vật lý: Các chuyển động va chạm tương ứng với các đối tượng trong game.
- Chuyển động: Di chuyển qua trái, đi lên, đi xuống, qua phải.
- Hành động:
    + Điều khiển: tổ hợp phím [←][↑][↓][→] hoặc tổ hợp phím [A][W][S][D]
    + Tùy chọn khác:
        - Khi người chơi truy cập game sẽ hiển thị Menu game gồm Play Game, Settings và Exit Game. Trong mục Thiết lập sẽ có Điều chỉnh Volume (Âm lượng), Bright (Độ sáng) và Exit Game.
        - Khi đang chơi game, người chơi có thể nhấn phím “ESC” trên bàn phím hoặc nút hình “bánh răng cưa” ở góc phải màn hình thì hiển thị popup Settings và đồng thời người chơi không thể di chuyển khi đang trong trạng thái này.

- Khi người chơi đi đến từng cổng (Gateway) và nhấn phím “E” thì sẽ điều hướng đến căn phòng tương ứng với cổng đó.
- Khi người chơi tương tác với NPC Guide (Người hướng dẫn), nhấn phím “E” thì sẽ hiển thị popup Guide;
- Khi người chơi tương tác với ô Game Mini, nhấn phím “E” thì sẽ hiển thị popup Game Mini tương ứng, nhấn đồng ý chi trả thể lực để bắt đầu game mini đó.
- Người chơi cũng thể nhấn phím “M” để quay về lại trang Menu game. Phím “B” để mở cửa hàng thể lực. Phím “I” để xem thông tin người chơi.

+ Phát lại và lưu trữ: Cơ sở dữ liệu PostgreSQL.

1.4.11 Thế giới game

1.4.11.1 Chi tiết về địa điểm

- Địa điểm 0:
  + Đặc tả chung: Hall là một sảnh nơi tập trung tất cả các người chơi khi họ bắt đầu vào game.
  + Đặc trưng vật lý: Nhân vật đi xuyên vào cánh cổng và nhấn phím “E” để dịch chuyển đến địa điểm đó.
  + Liên kết đến các địa điểm khác: Khi vào bên trong Hall, người chơi có thể đi đến 1 trong 4 cánh cổng để bước qua các phòng khác
  + Đối tượng bên trong: Từ Gateway 1 đến Gateway 4.
- Địa điểm 1:
  + Đặc tả chung: Gateway 1 - Red Room là một cánh cổng không gian đưa người chơi đến căn phòng với tông màu đỏ.
  + Đặc trưng vật lý: Nhân vật khi đang ở trong phòng này có thể nhấn phím “E” tại cổng Hall để quay trở về Hall, hoặc bước tiếp để tương tác với NPC Guide cũng như ô Game Mini trong căn phòng đó.
  + Liên kết đến các địa điểm khác: Khi vào bên trong Red Room, người chơi có thể quay về lại Hall hoặc đi tiếp để qua căn phòng khác.

+ Đối tượng bên trong: NPC Guide, ô Game Mini “Galaxy 2048”.

- Địa điểm 2:
  + Đặc tả chung: Gateway 2 - Blue Room là một cánh cổng không gian đưa người chơi đến căn phòng với tông màu đỏ.
  + Đặc trưng vật lý: Nhân vật khi đang ở trong phòng này có thể nhấn phím “E” tại cổng Hall để quay trở về Hall, hoặc bước tiếp để tương tác với NPC Guide cũng như ô Game Mini trong căn phòng đó.
  + Liên kết đến các địa điểm khác: Khi vào bên trong Blue Room, người chơi có thể quay về lại Hall hoặc đi tiếp để qua căn phòng khác.
  + Đối tượng bên trong: NPC Guide, ô Game Mini “Cosmic Snake”.
- Địa điểm 3:
  + Đặc tả chung: Gateway 2 - Green Room là một cánh cổng không gian đưa người chơi đến căn phòng với tông màu đỏ.
  + Đặc trưng vật lý: Nhân vật khi đang ở trong phòng này có thể nhấn phím “E” tại cổng Hall để quay trở về Hall, hoặc bước tiếp để tương tác với NPC Guide cũng như ô Game Mini trong căn phòng đó.
  + Liên kết đến các địa điểm khác: Khi vào bên trong Green Room, người chơi có thể quay về lại Hall hoặc đi tiếp để qua căn phòng khác.
  + Đối tượng bên trong: NPC Guide, ô Game Mini “Spaceship Flappy”.
- Địa điểm 4:
  + Đặc tả chung: Gateway 4 - Purple Room là một cánh cổng không gian đưa người chơi đến căn phòng với tông màu đỏ.
  + Đặc trưng vật lý: Nhân vật khi đang ở trong phòng này có thể nhấn phím “E” tại cổng Hall để quay trở về Hall, hoặc bước tiếp để tương tác với NPC Guide cũng như ô Game Mini trong căn phòng đó.
  + Liên kết đến các địa điểm khác: Khi vào bên trong Purple Room, người chơi có thể quay về lại Hall hoặc đi tiếp để qua căn phòng khác.
  + Đối tượng bên trong: NPC Guide, ô Game Mini “Robotic Dinosaur”.

1.4.11.2 Chi tiết về đối tượng

- Đối tượng 0:
    + Chi tiết đối tượng:

        Tên: NPC Guide

        Ngày sinh: ??-??-????

        Giới tính: Không xác định

        Quê quán: Vũ trụ Journey
    + Tính cách nhân vật: NPC Guide là người có ngoại hình và cách ăn mặc khá lập dị nhưng nói về kiến thức thì ngài ấy được xem là một thiên tài, trên thông thiên văn, dưới tường địa lý, ngài có thể chỉ dẫn cho người chơi biết họ cần phải làm gì, game mini thực hiện như thế nào trong từng căn phòng bí mật tương ứng. Không một ai biết ngài sinh ra từ đâu, tương truyền rằng đây là người sống lâu đời nhất trong thế giới vũ trụ này.
- Đối tượng 1:
    + Chi tiết đối tượng:

        Tên: Najone

        Ngày sinh: 01-02-2021

        Giới tính: Nam

        Quê quán: Ấn Độ
    + Tính cách nhân vật: Najone là anh chàng thú vị, thích khám phá những điều mới mẻ. Nhìn anh rất tri thức bởi vì người ta thấy anh thường xuyên nghe nhạc và đọc sách.
- Đối tượng 2:
    + Chi tiết đối tượng:

        Tên: Somin

        Ngày sinh: 09-01-2022

        Giới tính: Nữ

        Quê quán: Hàn Quốc
    + Tính cách nhân vật: Somin là cô nàng ăn chơi khét tiếng, cô thường xuyên chi tiêu vào những món hàng xa xỉ và du lịch khắp nơi.

1.4.11.3 Chi tiết về luật game mini

- Game mini "Cosmic Snake":

  + Mô tả: Người chơi cố gắng ăn các sao bằng cách chạy vào chúng với đầu của con rắn. Mỗi sao ăn được làm cho con rắn dài hơn. Khi đầu rắn va chạm với chính bản thân nó thì người chơi thua cuộc.

  + Di chuyển: Phím [A][W][D][S].

  + Màn hình trạng thái: ready, playing, over; Click chuột để chuyển đổi.

  + Cách tính điểm: Ăn 1 sao thì tăng 1 điểm; Số kinh nghiệm thưởng bằng tổng điểm x 2; Số xu thưởng bằng tổng điểm.

- Game mini "Spaceship Flappy":

  + Mô tả: Điều khiển tàu di chuyển liên tục qua các ống có các khoảng trống kích thước bằng nhau được đặt ở các độ cao ngẫu nhiên. Tàu tự động hạ xuống và chỉ bay lên khi người chơi click chuột. Mỗi lần vượt qua một ống thành công, người chơi sẽ nhận được một điểm. Va chạm với một đường ống hoặc mặt đất, trò chơi kết thúc.

  + Di chuyển: Click chuột

  + Màn hình trạng thái: ready, playing, over; Click chuột để chuyển đổi.

  + Cách tính điểm: Nhảy qua 1 ống thì tăng 1 điểm; Số kinh nghiệm thưởng bằng tổng điểm x 2; Số xu thưởng bằng tổng điểm.

- Game mini "Galaxy 2048":

  + Mô tả: Mỗi lượt, một ô mới ngẫu nhiên xuất hiện ở một vị trí trống trên bảng với giá trị là 2. Các ô trượt càng xa càng tốt theo hướng đã chọn cho đến khi chúng bị chặn lại bởi một ô khác hoặc cạnh của lưới. Nếu hai ô cùng một số va chạm trong khi di chuyển, chúng sẽ hợp nhất thành một ô bằng tổng giá trị của hai ô đã va chạm. Người chơi có thể tiếp tục để đạt được điểm số cao hơn. Khi người chơi không có nước đi hợp pháp (không còn ô trống và không thể di chuyển sang 4 hướng), trò chơi kết thúc.

  + Di chuyển: Phím [A][W][D][S].

  + Màn hình trạng thái: ready, playing, over; Click chuột để chuyển đổi.

  + Cách tính điểm: Số kinh nghiệm thưởng bằng tổng điểm/5; Số xu thưởng bằng số kinh nghiệm/2.

- Game mini “Robotic Dinosaur”:

+ Mô tả: Robot tự động chạy liên tục về bên phải và người chơi cần nhấn phím nhảy để vượt qua các chướng ngại vật. Đi càng xa thì điểm càng cao.

+ Di chuyển: Hệ thống tự động di chuyển. Phím [Space] để nhảy.

+ Màn hình trạng thái: ready, playing, over; Click chuột để chuyển đổi.

+ Cách tính điểm: Số kinh nghiệm thưởng bằng tổng điểm/10; Số xu thưởng bằng số kinh nghiệm/2.

CHƯƠNG II: PHÂN TÍCH YÊU CẦU

2.1 Yêu cầu chức năng

2.1.1 Xác định tác nhân

Sau khi tìm hiểu, thu thập thông tin từ một số loại Game online 2D, chúng em đã đúc kết được tác nhân của game Journey chỉ có một đó là người chơi.

2.1.2 Xác định các use cases

Dựa vào yêu cầu đặc ra, và với các tác nhân tương ứng chúng em xác định use case cho game online này gồm:

	ID	Tên use case	Mô tả
1	SU	Đăng ký tài khoản	Người dùng đăng ký tài khoản game.
2	SI	Đăng nhập	Người dùng đăng nhập tài khoản game.
3	VM	Xem menu game	Người dùng xem menu game.
4	VS	Xem thiết lập hệ thống	Người dùng xem thông tin các thiết lập về âm thanh và độ sáng.
5	MV	Điều chỉnh âm lượng	Người dùng điều chỉnh âm lượng nhạc nền trong game.
6	MB	Điều chỉnh độ sáng	Người dùng điều chỉnh độ sáng trong game.
7	PG	Chơi game	Người chơi vào bên trong game và trạng thái người chơi lúc này là online.
8	SC	Lựa chọn nhân vật	Người chơi lựa chọn đối tượng nhân vật nam hoặc nữ.

9	ER	Vào phòng bí mật	Người chơi vào các cánh cổng để dịch chuyển đến căn phòng bí mật tương ứng.
10	PGM	Chơi game mini	Người chơi chơi các game mini của từng phòng tương ứng để lấy tiền và kinh nghiệm. Người chơi cũng bị mất một khoảng thể lực nhất định để có thể chơi được game mini.
11	CR	Chat trực tuyến	Tất cả người chơi đều có thể chat với nhau theo thời gian thực.
12	BM	Mua thể lực	Người chơi dùng xu để mua thể lực từ cửa hàng thể lực.
13	VI	Xem Thông tin người chơi	Người chơi xem các thông tin điểm cao nhất đạt được của các căn phòng bí mật đã chơi. Xem danh sách các người chơi khác và biết được tình trạng online hoặc offline của họ.
14	VB	Xem bảng xếp hạng	Người chơi xem thông tin về cấp độ và thứ tự xếp hạng của người chơi khác.
15	EG	Thoát game	Người chơi thoát khỏi game.

Bảng 3. Các use cases trong game

2.1.3 Danh sách các phím tắt trong game

	Phím tắt	**Mô tả**
1	ESC	Bật/tắt modal Thiết lập hệ thống

2	M	Điều hướng đến trang Menu
3	E	Tương tác với cánh cổng để điều hướng vào từng phòng tương ứng hoặc Bật modal Game mini.
4	B	Bật/tắt modal Cửa hàng thể lực
5	I	Bật/tắt modal Thông tin người chơi

Bảng 4. Các phím tắt trong game

2.2 Yêu cầu phi chức năng

2.2.1 Yêu cầu về Hiệu suất

Dữ liệu trong quá trình chơi được cập nhật theo thời gian thực, nhất quán toàn vẹn.
Thông tin tiếp nhận được nhanh chóng hiệu quả.
Giao diện thân thiện, dễ sử dụng, tốc độ phản hồi nhanh.
Có khả năng duy trì thời gian truy cập cao và ổn định.

2.2.2 Yêu cầu về Bảo mật

Đảm bảo dữ liệu của các người chơi hiển thị đúng với các tài khoản khác đã truy cập.

2.2.3 Thuộc tính chất lượng phần mềm

Xây dựng trên nền tảng web và tương thích được với nhiều loại trình duyệt.
Dễ dàng bảo trì, nâng cấp, phát triển sau khi đã triển khai.
Khả năng tương tác, khả năng bảo trì, độ tin cậy cao.
Khả năng kiểm soát dữ liệu tuyệt đối, an toàn và bảo mật.

2.3 Đặc tả game

Game online HTML5 kết hợp ASP.NET - Journey cung cấp dịch vụ cho đối tượng người chơi ở mọi lứa tuổi và hướng đến trẻ em là chính. Do đó, bối cảnh game cần có vui nhộn, lôi cuốn, mang thiên hướng học đường, gần gũi. Nhân vật và NPC thể hiện rõ những đặc tính riêng và linh hoạt nhất có thể, nhạc nền sống động. Các chức năng hệ thống phải được tổng hợp tại phần Menu game, các chức năng liên quan đến người chơi được tổng hợp tại phần Player Info. Giao diện game dễ sử dụng, luật chơi đơn giản nhưng không kém phần thử thách. Người chơi sau khi vào web game Journey này sẽ cần tài khoản đăng nhập do đó cũng cần đăng ký tài khoản game trước.

Game cho phép một số chức năng như Xem Menu game về các chức năng của game như Chơi game, Lựa chọn nhân vật, Xem Thiết lập hệ thống (gồm Điều chỉnh âm lượng và độ sáng), Thoát game. Tại thanh thông tin người chơi có cấp bậc, thanh kinh nghiệm (0-100%) và thanh thể lực (0-300%). Trong chức năng Xem thông tin người chơi sẽ biết thêm về điểm số cao nhất đạt được của từng game mini và Bảng xếp hạng. Với bảng xếp hạng là những người đang chơi game hay còn gọi là đang hoạt động (active) và được sắp xếp theo thứ tự cấp độ từ cao xuống thấp.

Đối với chức năng Chơi Game, người chơi sẽ vào trực tiếp game ở vị trí Sảnh trong bản đồ chính và thông tin về trạng thái của người chơi khi đó sẽ được truy xuất từ CSDL và hiển thị cho phù hợp. Đối với chức năng Lựa chọn nhân vật thì người chơi được phép chọn đối tượng nhân vật theo giới tính nam hoặc nữ mà người chơi thích, khi vào game thì hệ thống sẽ sử dụng đối tượng nhân vật được chọn làm mẫu nhân vật. Username sẽ được hiển thị phía trên đầu nhân vật của người chơi và được phân biệt giữa người chơi và các người chơi khác bằng cách hiển thị màu đỏ cho người chơi chính và màu trắng cho các người chơi khác. Khi vào từng căn phòng bí mật và tiến đến các ô Game Mini nhấn phím tắt “E” thì popup game mini tương ứng hiện lên, người chơi chi trả điểm thể lực để đổi lấy lượt tham gia game mini. Hoàn thành một lượt sẽ được thưởng kinh nghiệm và xu tương ứng. Người chơi dùng xu để mua điểm thể lực từ cửa hàng thể lực, với số th Kinh nghiệm có giá trị trong khoảng từ 0-100, khi đầy thì thăng cấp. Xu là vô hạn. Hệ thống sẽ tự động cập nhật số xu, kinh nghiệm, thể lực khi người chơi thực hiện các game mini trong căn phòng bí mật. Cửa hàng thể lực có thể click chuột vào biểu tượng cửa hàng hoặc nhấn phím “B”.

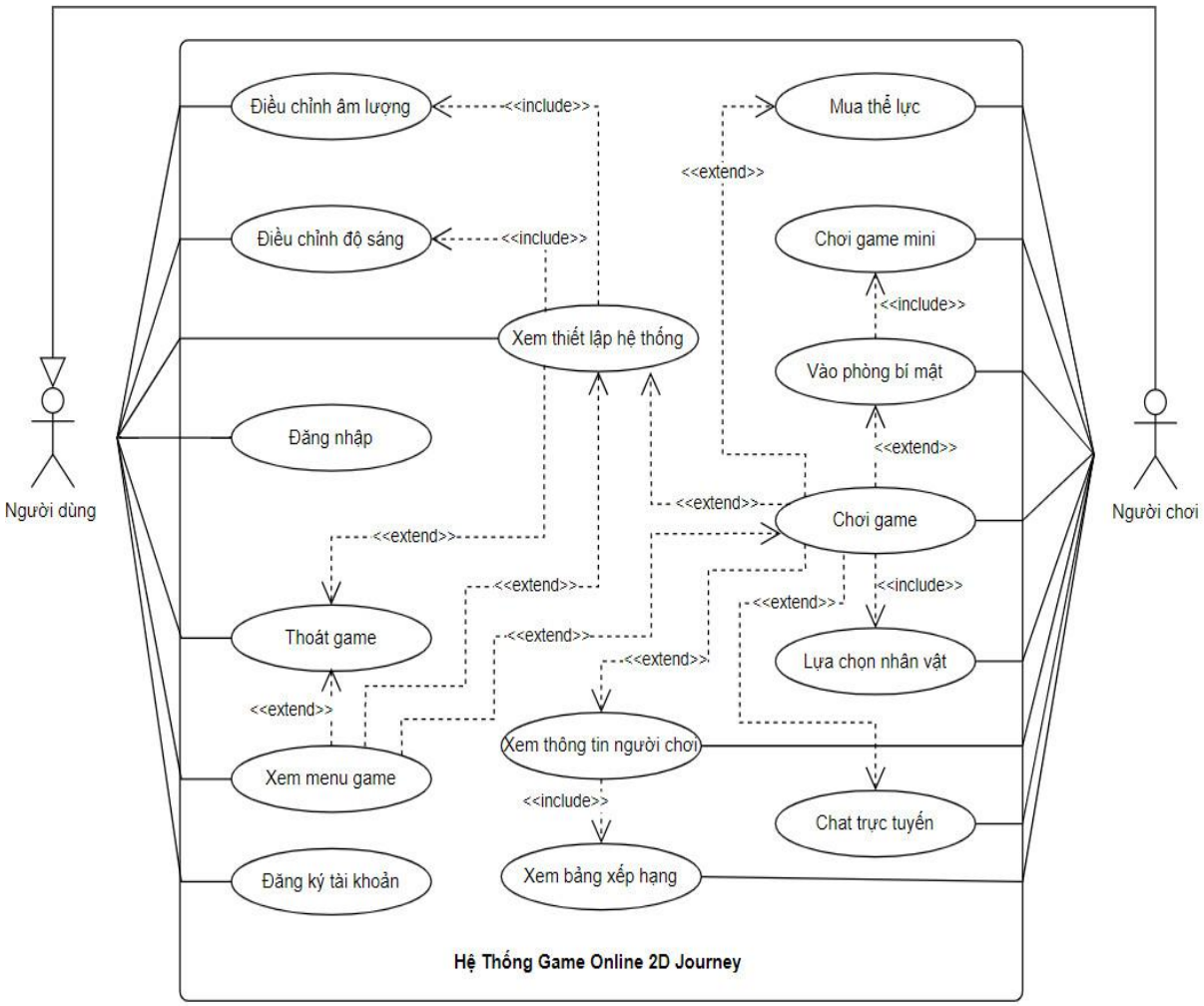
Người chơi giao tiếp với các người chơi khác thông qua khung chat ở góc trái bên dưới màn hình, các đoạn chat được hiển thị theo thời gian thực.

Đối với chức năng Xem thiết lập hệ thống thì khi người chơi chọn vào chức năng này với hình "bánh răng cưa" ở góc phải màn hình hoặc nhấn phím "ESC" sẽ xuất hiện một popup giúp người chơi điều chỉnh độ sáng và âm thanh nền trong game tương đương 0-100% và các giá trị thiết lập này sẽ được lưu trữ bằng Local Storage. Đối với chức năng Thoát (Exit Game), người chơi sẽ bị xóa session và đồng thời nhảy về giao diện trang chủ, các thông tin về tiền xu, thể lực, kinh nghiệm, cấp độ sẽ được lưu lại. Phím "I" được sử dụng để hiển thị popup chứa thông tin người chơi thay cho việc click chuột vào avatar. Nhấn phím "M" để điều hướng về trang Menu game.

Một số sự kiện trong game như Người hướng dẫn (NPC Guide), Cổng game (Gateway) sẽ được bố trí theo từng căn phòng bí mật. Đối với NPC, sẽ giúp người chơi biết mình cần làm gì tiếp theo. Đối với Cổng game sẽ điều hướng qua các căn phòng bí mật tương ứng hoặc quay về lại sảnh Hall thông qua việc nhấn phím "E".

CHƯƠNG III: THIẾT KẾ HỆ THỐNG

3.1 Sơ đồ use case tổng quát



Hình 10. Sơ đồ use case tổng quát

Giải thích:

Hệ thống chỉ có một tác nhân là Người chơi. Do đó, ngầm hiểu rằng tác nhân "Người dùng" chỉ là một đối tượng tạm để tổng hợp hóa cho tác nhân "Người chơi". Việc vẽ sơ đồ có đối tượng tạm này nhằm mục đích giúp sơ đồ gọn gàng hơn, cũng như dễ dàng hình dung về việc Người dùng là người thao tác trên hệ thống nhưng chưa thực sự đang trong game, còn Người chơi là người ở trạng thái đã vào chơi game.

3.2 Đặc tả các use cases chính

3.2.1 Use case Đăng ký tài khoản

Use case	Đăng ký tài khoản	
ID	SU	
Mô tả chi tiết	Người dùng đăng ký tài khoản game.	
Sự kiện kích hoạt	Truy cập vào đường dẫn đăng ký hoặc ấn vào nút “Sign Up” trên navbar.	
Mối liên hệ giữa các usecases	-	
Điều kiện trước	Truy cập được vào trang đăng ký.	
Điều kiện sau	Đăng ký tài khoản game thành công.	
Luồng sự kiện	Người dùng	Hệ thống
	1. Điền thông tin vào form đăng ký tài khoản. 2. Nhấn vào nút “Sign Up”.	1.1 Hiển thị giao diện form đăng ký tài khoản game. 2.1 Kiểm tra tính hợp lệ của các thông tin trong form. Nếu hợp lệ thì gửi yêu cầu thêm tài khoản vào CSDL, nếu không thông báo yêu cầu điền lại form. 2.2 Kiểm tra tính duy nhất của thông tin theo username trên CSDL. Nếu hợp lệ, insert thông tin người dùng đó vào CSDL và thông báo đăng ký tài khoản thành công. Nếu không hợp lệ, yêu cầu điền lại form.
Ngoại lệ (nếu có)	-	

Bảng 5. Đặc tả use case Đăng ký tài khoản

3.2.2 Use case Đăng nhập

<table>
<tr><td>Use case</td><td colspan="2">Đăng nhập</td></tr>
<tr><td>ID</td><td colspan="2">SI</td></tr>
<tr><td>Mô tả chi tiết</td><td colspan="2">Người dùng đăng nhập vào hệ thống để được cấp quyền chơi game.</td></tr>
<tr><td>Sự kiện kích hoạt</td><td colspan="2">Truy cập vào trang đăng nhập của hệ thống hoặc nhấp vào nút “Sign In” trên navbar.</td></tr>
<tr><td>Mối liên hệ giữa các use cases</td><td colspan="2">-</td></tr>
<tr><td>Điều kiện trước</td><td colspan="2">Truy cập được trang đăng nhập.</td></tr>
<tr><td>Điều kiện sau</td><td colspan="2">Đăng nhập thành công.</td></tr>
<tr><td rowspan="2">Luồng sự kiện</td><td>Người dùng</td><td>Hệ thống</td></tr>
<tr><td>1. Điền username và password.
2. Nhấp vào nút “Sign In”.</td><td>1.1 Hiển thị giao diện form đăng nhập.

2.1 Kiểm tra tính hợp lệ của các thông tin trong form. Nếu hợp lệ thì gửi request đăng nhập kèm username và password đã được hash. Nếu không hợp lệ, thông báo yêu cầu điền lại form.
2.2 Kiểm tra username và password có tồn tại trong CSDL hay không. Nếu có tồn tại thì tạo session cho người dùng và điều hướng người dùng về trang chủ, nếu không tồn tại thì thông báo đăng nhập không thành công.</td></tr>
<tr><td>Ngoại lệ (nếu có)</td><td colspan="2">Người dùng chưa đăng ký tài khoản game.</td></tr>
</table>

Bảng 6. Đặc tả use case Đăng nhập

3.2.3 Use case Xem thiết lập hệ thống

Use case	Xem thiết lập hệ thống	
ID	VS	
Mô tả chi tiết	Cho phép người dùng xem các thông số thiết lập hệ thống về âm lượng, độ sáng và thoát game.	
Sự kiện kích hoạt	Người dùng nhấn phím "ESC" hoặc nhấn nút hình "bánh răng cưa".	
Mối liên hệ giữa các usecases	Điều chỉnh âm lượng, Điều chỉnh độ sáng, Thoát game.	
Điều kiện trước	Đăng nhập thành công	
Điều kiện sau	-	
Luồng sự kiện	Người dùng	Hệ thống
	1. Nhấn phím "ESC" hoặc biểu tượng "bánh răng cưa"	1.1 Hiển thị giao diện thiết lập hệ thống.
	2. Điều chỉnh độ sáng hoặc âm lượng.	2.1 Lưu các thay đổi về độ sáng, âm lượng vào 2 biến trong local storage. Truy xuất giá trị của biến từ local storage và hiển thị css tương ứng.
	3. Nhấp vào nút "Exit Game" và xác nhận thoát.	3.1 Hiển thị modal xác nhận thoát game. Nếu có xác nhận thoát game, chuyển status sang offline, xóa session người dùng và điều hướng về trang chủ.
Ngoại lệ (nếu có)	-	

Bảng 7. Đặc tả use case Xem thiết lập hệ thống

3.2.4 Use case Chơi game

Use case	Chơi game	
ID	PG	
Mô tả chi tiết	Trạng thái của người chơi lúc này là online. Người chơi di chuyển nhân vật và thao tác các chức năng đã được tổng hợp bên trong game. Các thông tin về người chơi luôn được lưu lại trên CSDL.	
Sự kiện kích hoạt	Người chơi chọn nhân vật	
Mối liên hệ giữa các usecases	Xem menu game, Lựa chọn nhân vật, Chat trực tuyến, Xem thông tin người chơi, Xem bảng xếp hạng, Xem thiết lập hệ thống, Vào phòng bí mật, Mua thể lực.	
Điều kiện trước	Đăng nhập thành công và chọn nhân vật thành công.	
Điều kiện sau	Trạng thái online.	
Luồng sự kiện	Người dùng	Hệ thống
	1. Vào trong game.	1.1 Truy xuất thông tin từ CSDL về người chơi và hiển thị theo đúng cấp độ, số kinh nghiệm, số xu, số thể lực. 1.2 Vẽ bản đồ canvas sảnh chính.
	2. Điều khiển nhân vật.	2.1 Tính toán tọa độ và hiển thị mô hình nhân vật và các animation tương ứng với các di chuyển của nhân vật.
	3. Bước đến cổng vào phòng bí mật.	3.1 Tiếp nhận sự kiện va chạm và điều hướng người chơi sang view phòng bí mật theo id phòng tương ứng với cổng mà nhân vật bước vào. 3.2 Vẽ bản đồ canvas cho phòng bí mật tương ứng đó. 4.1 Tiếp nhận sự kiện va chạm và hiển thị

	4. Bước vào ô Game mini. 5. Nhấn vào khung chat, nhập nội dung. 6. Nhấn vào avatar để xem thêm thông tin chi tiết. 7. Nhấn phím "ESC" hoặc "bánh răng cưa" 8. Nhấn nút "Buy Stamina".	modal game mini tương ứng và thực hiện xử lí chi tiết bởi use case Chơi game mini. 5.1 Thực hiện xử lí chi tiết bởi use case Chat trực tuyến. 6.1 Thực hiện xử lí chi tiết bởi use case Xem thông tin người chơi và Xem bảng xếp hạng. 7.1 Thực hiện xử lí chi tiết bởi use case Xem thiết lập hệ thống. 8.1 Thực hiện xử lí chi tiết bởi use case Mua thể lực.
Ngoại lệ (nếu có)	-	

Bảng 8. Đặc tả use case Chơi game

3.2.5 Use case Lựa chọn nhân vật

Use case	Lựa chọn nhân vật
ID	SC
Mô tả chi tiết	Người chơi lựa chọn nhân vật nam hoặc nữ
Sự kiện kích hoạt	Người chơi nhấn nút "Play Game" trong menu game.
Mối liên hệ giữa các usecases	Chơi game
Điều kiện trước	Đăng nhập thành công.
Điều kiện sau	Mô hình nhân vật hiển thị theo lựa chọn của người chơi.

Luồng sự kiện	Người dùng	Hệ thống
	1. Nhấn chọn nhân vật nam hoặc nữ	1.1 Hiển thị giao diện lựa chọn nhân vật nam và nữ. 1.2 Lưu lựa chọn nhân vật và hiển thị mô hình nhân vật đã chọn.
Ngoại lệ (nếu có)	-	

Bảng 9. Đặc tả use case Lựa chọn nhân vật

3.2.6 Use case Chơi game mini

Use case	Chơi game mini	
ID	PGM	
Mô tả chi tiết	Người chơi thực hiện các game mini trong từng căn phòng bí mật.	
Sự kiện kích hoạt	Bước vào ô Game mini và nhấn phím “E”	
Mối liên hệ giữa các usecases	Vào phòng bí mật.	
Điều kiện trước	Trạng thái online.	
Điều kiện sau	Trừ và cập nhật số thể lực, Cập nhật số xu và kinh nghiệm, Xét thăng cấp.	
Luồng sự kiện	Người dùng	Hệ thống
	1. Bước vào ô Game Mini và nhấn phím “E”. 2. Xác nhận chơi game mini.	1.1 Hiển thị giao diện game mini. 2.1 Kiểm tra số dư thể lực có đủ thỏa yêu cầu của game hay không. Nếu thỏa, bắt đầu game mini và tính điểm. Nếu không

	3. Hoàn tất game mini	thỏa, thông báo không đủ thể lực để tham gia game mini. 3.1 Quy đổi số điểm đạt được thành số xu và kinh nghiệm tương ứng. Cập nhật số xu và số kinh nghiệm. 3.2 Kiểm tra thanh kinh nghiệm có đầy hay không. Nếu đầy, thăng lên 1 cấp và cập nhật lại số kinh nghiệm. Nếu không, giữ nguyên hiện trạng. 3.3 Đóng game mini.
Ngoại lệ (nếu có)	Số thể lực không đủ.	

Bảng 10. Đặc tả use case Chơi game mini

3.2.7 Use case Chat trực tuyến

Use case	Chat trực tuyến	
ID	CR	
Mô tả chi tiết	Người chơi có thể chat trực tuyến với các người chơi khác theo thời gian thực.	
Sự kiện kích hoạt	Nhập nội dung vào trường input và nhấn phím “Enter” hoặc nút “Send”.	
Mối liên hệ giữa các usecases	Chơi game.	
Điều kiện trước	Trạng thái online.	
Điều kiện sau	Nội dung được gửi và hiển thị lên khung chat thành công	
Luồng sự kiện	Người dùng	Hệ thống
	1. Nhấn vào thanh input trong khung chat, nhập nội	1.1 Hiển thị theo thời gian thực các thông tin của các messages vào khung chat khi có bất kỳ messages nào được lưu vào

	dung và nhấn phím "Enter" hoặc nút "Send".	CSDL. 1.2 Lưu các nội dung, thời gian gửi và tên người gửi vào CSDL. 1.3 Hiển thị nội dung của các messages
Ngoại lệ (nếu có)	-	

Bảng 11. Đặc tả use case Chat trực tuyến

3.2.8 Use case Mua thể lực

Use case	Mua thể lực	
ID	BM	
Mô tả chi tiết	Người chơi mua thể lực từ cửa hàng thể lực.	
Sự kiện kích hoạt	Nhấn vào nút biểu tượng cửa hàng (Buy Stamina) hoặc nhấn phím "B".	
Mối liên hệ giữa các usecases	Chơi game.	
Điều kiện trước	Trạng thái online, Số xu lớn hơn hoặc bằng số xu yêu cầu để mua thể lực.	
Điều kiện sau	Cập nhật lại số thể lực.	
Luồng sự kiện	Người dùng	Hệ thống
	1. Nhấn phím "B" hoặc nút biểu tượng Stamina Store. 2. Chọn gói thể lực và xác nhận mua thể lực.	1.1 Hiển thị giao diện mua thể lực theo gói. 2.1 Kiểm tra số xu có đủ thỏa yêu cầu của gói thể lực. Nếu thỏa, thông báo mua thể lực thành công và cập nhật lại số thể lực và số xu. Nếu không thỏa, thông báo số xu không đủ.

Ngoại lệ (nếu có)	-

Bảng 12. Đặc tả use case Mua thể lực

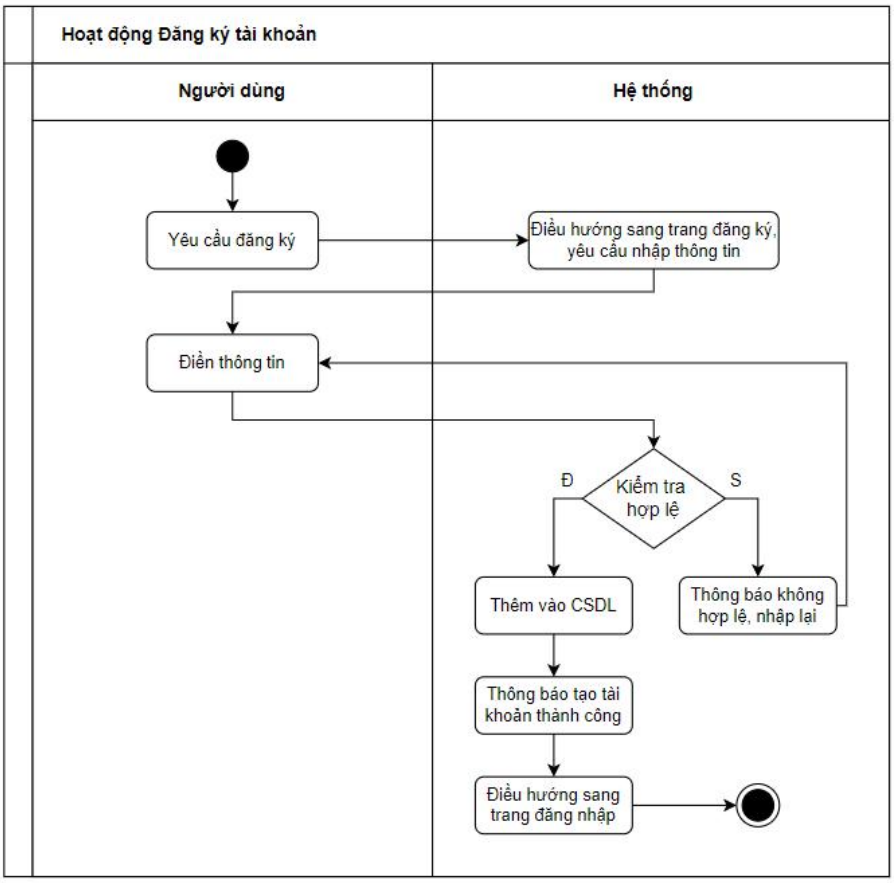
3.2.9 Use case Xem thông tin người chơi

Use case	Xem thông tin người chơi	
ID	VI	
Mô tả chi tiết	Người chơi xem thông tin bản thân và có thể xem danh sách bảng xếp hạng các người chơi khác.	
Sự kiện kích hoạt	Nhấn vào avatar hoặc nhấn phím “I”.	
Mối liên hệ giữa các usecases	Chơi game, Xem bảng xếp hạng.	
Điều kiện trước	Trạng thái online.	
Điều kiện sau	-	
Luồng sự kiện	Người dùng	Hệ thống
	1. Nhấn avatar hoặc nhấn phím “I” để xem thông tin người chơi. 2. Xem danh sách bảng xếp hạng các người chơi.	1.1 Truy xuất thông tin về cấp độ, số kinh nghiệm, số thể lực từ người chơi, điểm số cao nhất và Hiển thị giao diện xem thông tin người chơi tương ứng. 2.1 Truy xuất thông tin danh sách các người chơi có kèm trạng thái online/offline, sắp xếp theo thứ tự giảm dần, hiển thị giao diện danh sách bảng xếp hạng.
Ngoại lệ (nếu có)	-	

Bảng 13. Đặc tả use case Xem thông tin người chơi

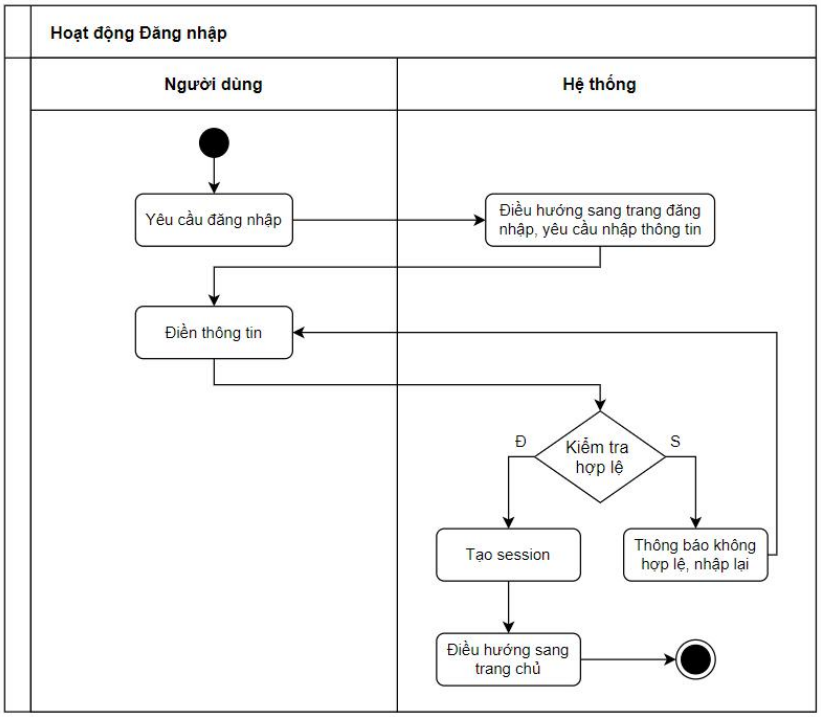
3.3 Sơ đồ hoạt động chính

3.3.1 Sơ đồ hoạt động use case Đăng ký tài khoản



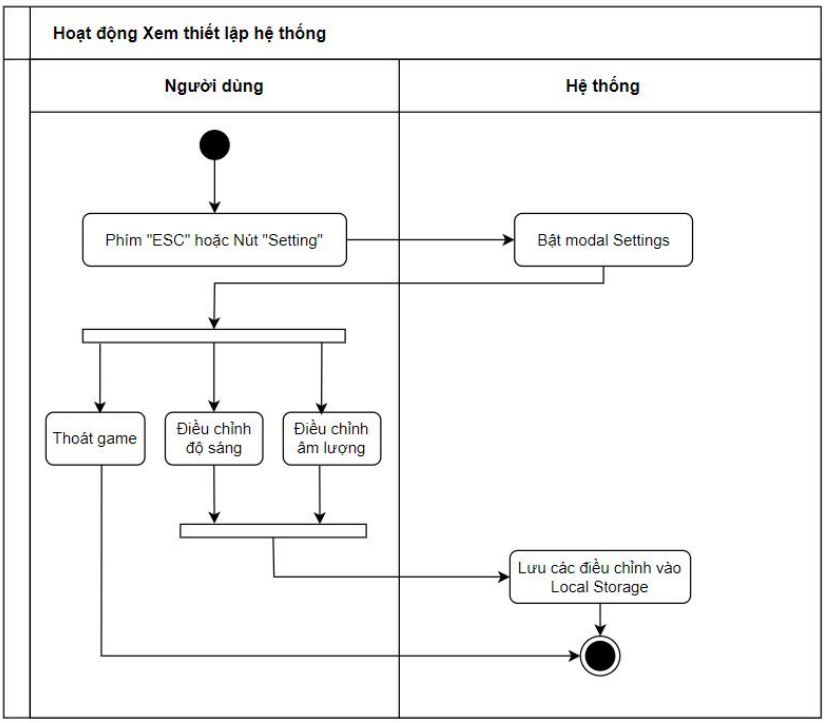
Hình 11. Sơ đồ hoạt động use case Đăng ký tài khoản

3.3.2 Sơ đồ hoạt động use case Đăng nhập



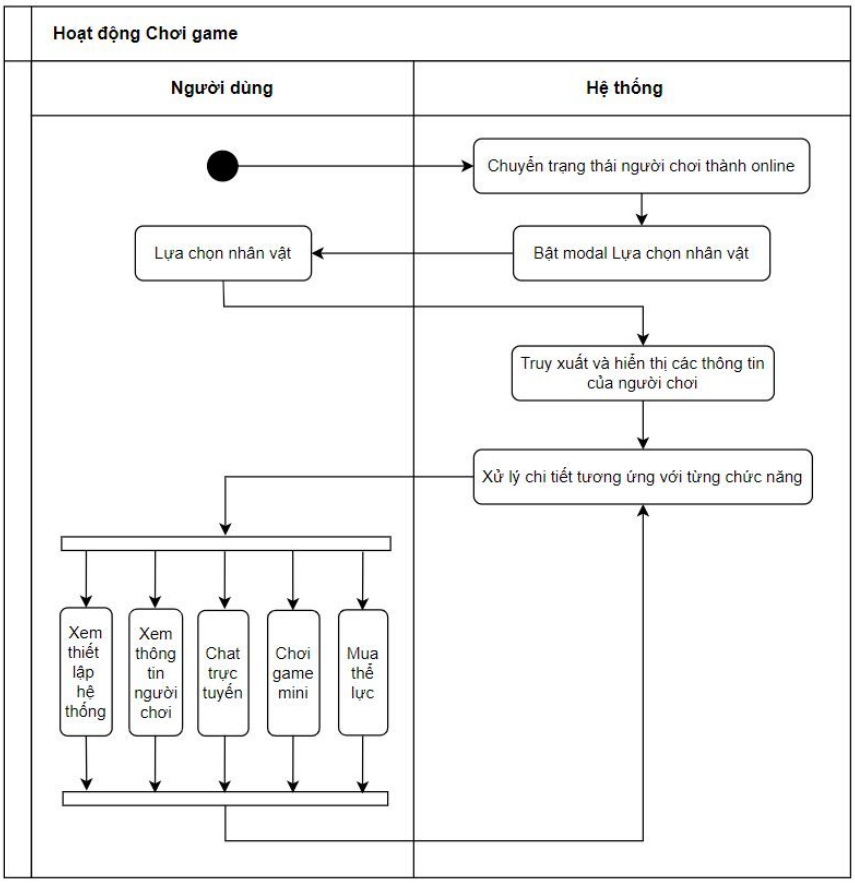
Hình 12. Sơ đồ hoạt động use case Đăng nhập

3.3.3 Sơ đồ hoạt động use case Xem thiết lập hệ thống



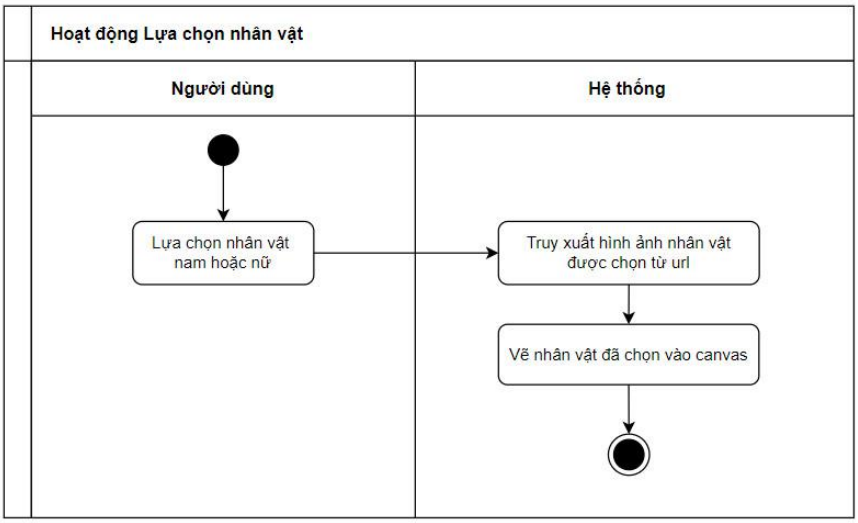
Hình 13. Sơ đồ hoạt động use case Xem thiết lập hệ thống

3.3.4 Sơ đồ hoạt động use case Chơi game



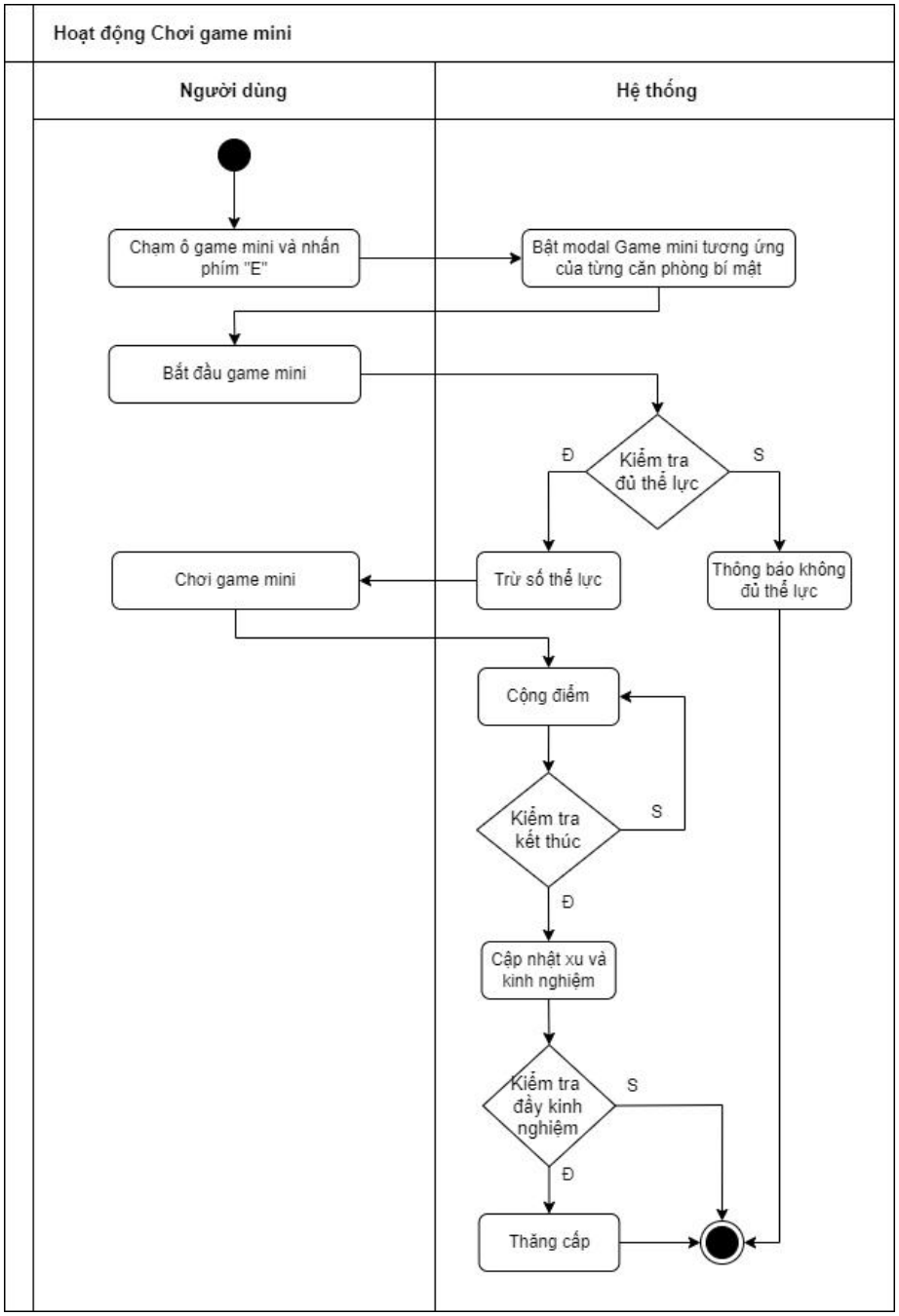
Hình 14. Sơ đồ hoạt động use case Chơi game

3.3.5 Sơ đồ hoạt động use case Lựa chọn nhân vật



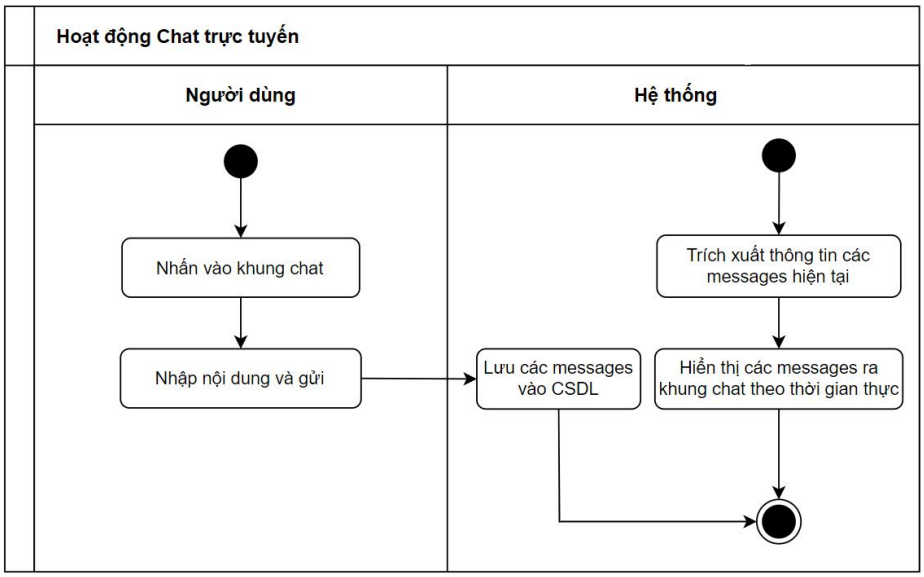
Hình 15. Sơ đồ hoạt động use case Lựa chọn nhân vật

3.3.6 Sơ đồ hoạt động use case Chơi game mini



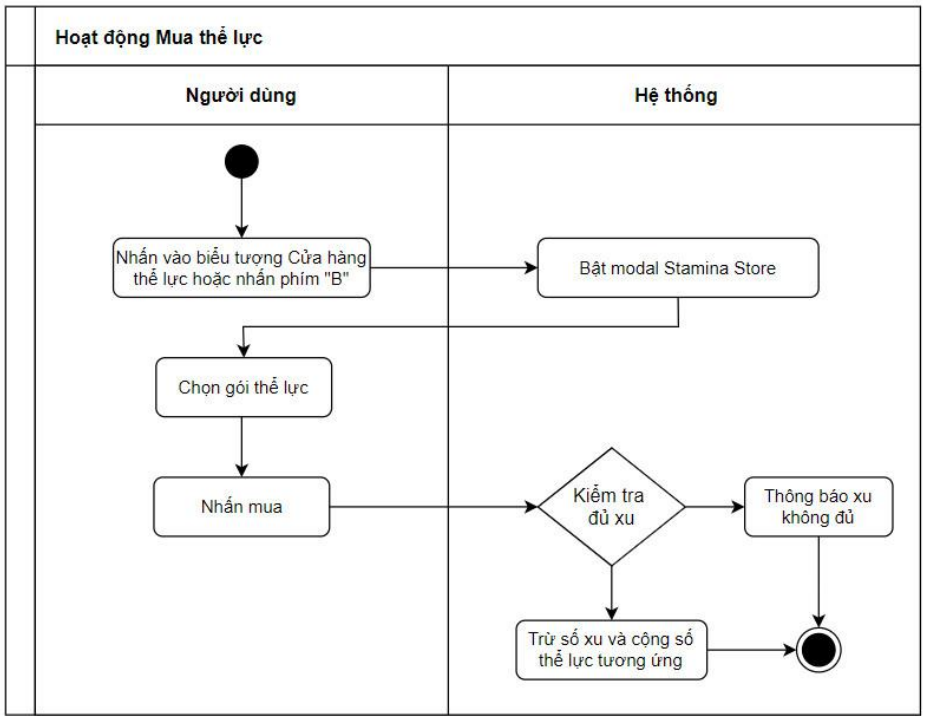
Hình 16. Sơ đồ hoạt động use case Chơi game mini

3.3.7 Sơ đồ hoạt động use case Chat trực tuyến



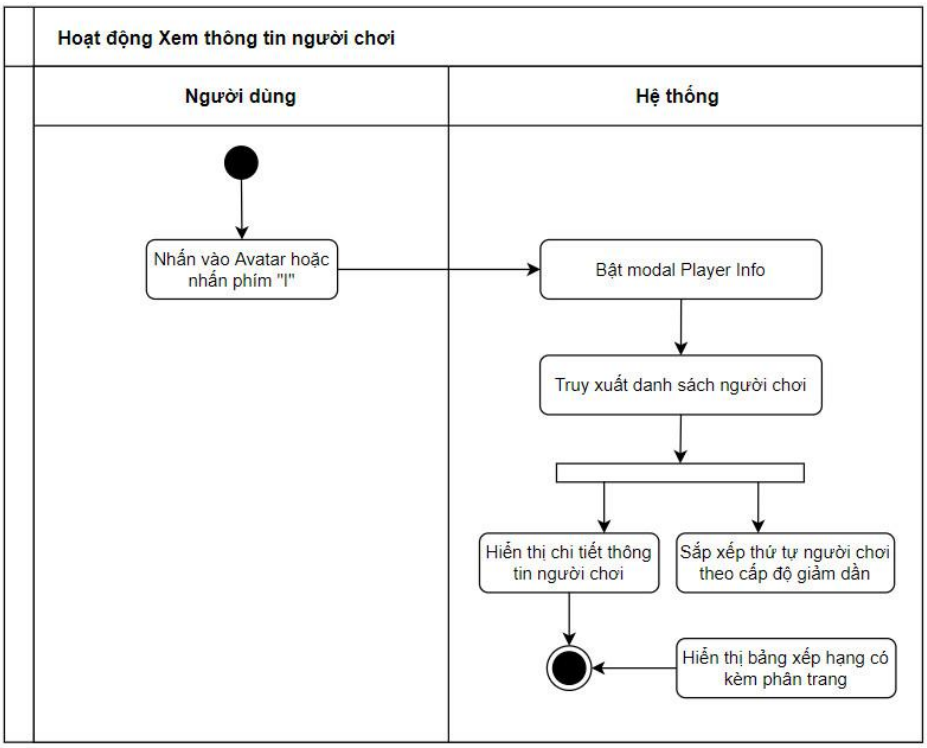
Hình 17. Sơ đồ hoạt động use case Chat trực tuyến

3.3.8 Sơ đồ hoạt động use case Mua thể lực



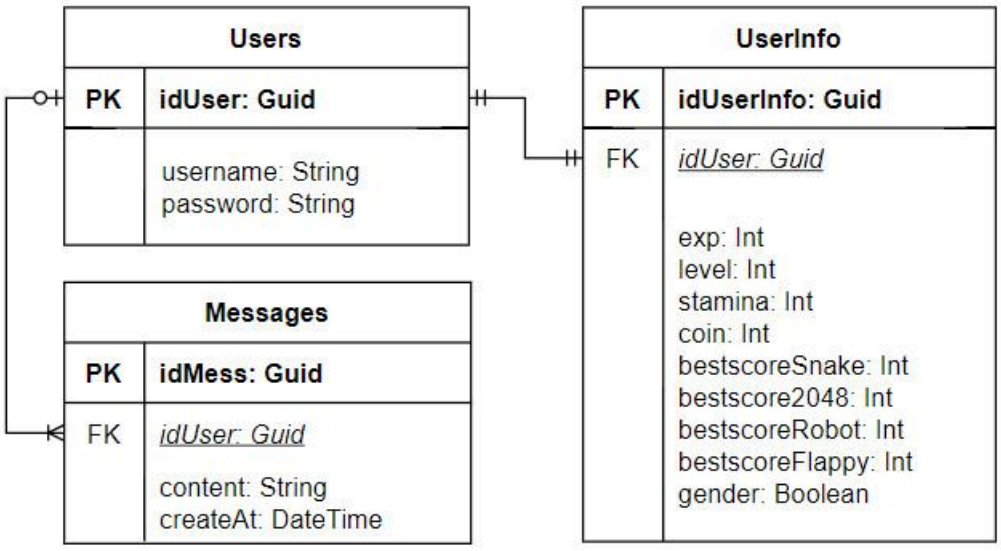
Hình 18. Sơ đồ hoạt động use case Mua thể lực

3.3.9 Sơ đồ hoạt động use case Xem thông tin người chơi



Hình 19. Sơ đồ hoạt động use case Xem thông tin người chơi

3.4 Sơ đồ ERD



Hình 20. Sơ đồ ERD

CHƯƠNG IV: HIỆN THỰC HỆ THỐNG VÀ KẾT QUẢ

Ở chương này, nhóm chúng em chỉ trình bày những đoạn code quan trọng từ Controller trong ASP.NET với một vài phần phụ trợ bởi JavaScript để diễn giải cho việc hiện thực nên các chức năng chính của hệ thống được rõ ràng và dễ hiểu nhất, một số phần code khác như từ thư mục Models, Views, DAO… sẽ không nêu trong này. Các bước để chạy hệ thống sẽ được trình bày rõ ràng trong readme.txt.

4.1 Hiện thực

4.1.1 Về Tài khoản game

4.1.1.1 Đăng ký tài khoản

Tạo một SignUpController cho việc đăng ký tài khoản với Index() để hiện thị giao diện UI và SignUp() được gọi thông qua route signup theo phương thức POST.

```
public class SignUpController : Controller
{
        private GameOnlineContext _context;
        public SignUpController(GameOnlineContext context){
                _context = context;
        }
        public IActionResult Index(){return View();}
        [Route("signup")]
        [HttpPost]
        public IActionResult SignUp(SignUpViewModel signUpViewModel){
                var dao = new UserDAO(_context);
                if (!ModelState.IsValid){return View("Index");}
                if (dao.getByUsername(signUpViewModel.Username) != null){
                        ViewBag.Error = "Username has been used.";
                        return View("Index");
                }
```

```
            if (signUpViewModel.Password !=
                signUpViewModel.ConfirmPassword){
                return View("Index");
            }
            User user = new User(){
                UserId = Guid.NewGuid(),
                Username = signUpViewModel.Username,
                Password = BCrypt.Net.BCrypt.HashPassword(
                    signUpViewModel.Password),
            };
            UserInfo userInfo = new UserInfo(){
                UserInfoId = Guid.NewGuid(),
                Exp = 0, Level = 1, Stamina = 100, Coin = 100,
                User = user,
            };
            dao.createUser(user);
            var infoDAO = new UserInfoDAO(_context);
            infoDAO.createUserInfo(userInfo);
            return Redirect("/signin");
        }
    }
```

DAO lấy dữ liệu và kiểm tra sự tồn tại, nếu có tồn tại thì thông báo qua ViewBag rằng tên này đã có người sử dụng và điều hướng về lại trang index để nhập lại. Nếu mọi thứ đúng thì khởi tạo đối tượng user chứa thông tin gồm UserId sinh mã theo Guid, Username và Password được lấy từ input với Password đã được hash theo kiểu BCrypt có sẵn của ASP.NET framework. Bên cạnh đó, thông tin của userinfo cũng được khởi tạo ban đầu với các thông số của một người chơi mới gồm kinh nghiệm - Exp là 0, cấp độ - Level là 1, thể lực - Stamina là 100, xu - Coin là 100, User chính là user vừa được khởi tạo. Lưu vào cơ sở dữ liệu và điều hướng đến trang đăng nhập.

4.1.1.2 Đăng nhập

Tạo một SignInController cho việc đăng nhập với Index() để hiện thị giao diện UI và SignIn() được gọi thông qua route signin theo phương thức POST.

```
public class SignInController : Controller
{
    private GameOnlineContext _context;
    public SignInController(GameOnlineContext context){
        _context = context;
    }
    public IActionResult Index(){return View();}
    [Route("signin")]
    [HttpPost]
    public IActionResult SignIn(SignInViewModel signInViewModel){
        var dao = new UserDAO(_context);
        var user = dao.getByUsername(signInViewModel.Username);
        if (!ModelState.IsValid){return View("Index");}
        if (user == null){
            ViewBag.Error = "Wrong username.";
            return View("Index");
        }
        if(!BCrypt.Net.BCrypt.Verify(signInViewModel.Password,
        user.Password)){
            ViewBag.Error = "Wrong password.";
            return View("Index");
        }
        HttpContext.Session.SetString("username", user.Username);
        return Redirect("/menu");
    }
}
```

DAO lấy dữ liệu và kiểm tra tính hợp lệ để thông báo tương ứng rằng sai về username hoặc sai về password. Khi mọi thứ đã đúng thì tạo một Session cho việc đăng nhập này với tên Session là username được lấy username hợp lệ khi nãy. Sau đó điều hướng đến trang menu game.

4.1.2 Về Thiết lập hệ thống game

4.1.2.1 Xem menu game

Tạo một MenuController cho việc hiển thị menu game với Index() để hiện thị giao diện UI, PlayGame() dùng cho xác định nhân vật nam hoặc nữ trong game, Exit() cho việc thoát game và được gọi thông qua route signout theo phương thức POST.

```
public class MenuController : Controller
{
        public IActionResult Index(){
                if (HttpContext.Session.GetString("username") != null)
                        return View();
                return Redirect("/signin");
        }
        public IActionResult PlayGame([FromQuery(Name = "gd")]string gd){
                return Redirect("/Game?gd=" + gd);
        }
        [HttpPost]
        [Route("/signout")]
        public IActionResult Exit(){
                HttpContext.Session.Clear();
                return Ok(new { code = "success" });
        }
}
```

Để vào Menu thì cần đăng nhập trước, do đó cần kiểm tra đã tồn tại Session hay chưa trong Index() để điều hướng phù hợp. Tại PlayGame() có tham số gd có 2 giá trị là male và female sẽ được truyền vào. Thực hiện Exit() sẽ clear Session đồng thời gửi thông điệp code là success.

Trong Javascript, tạo một số phím tắt lần lượt là phím "ESC" để thực hiện việc tắt mở modal Settings và phím "M" điều hướng về lại trang menu game cho thuận tiện.

```
// Press to open Settings modal
$(document).keydown((e) => {
    if (e.keyCode == 27) {
        if ($("#myModal1").modal("hide")) {
            $("#myModal1").modal("show");
            $("#myModal2").modal("hide");
            $(".modal-backdrop").css("display", "none");
        }
        else if ($("#myModal1").modal("show")) {
            $("#myModal1").modal("hide");
            $(".modal-backdrop").css("display", "none");
        }
    }
});
// Press M to get back to menu page
window.addEventListener("keydown", (event) => {
    if (event.key === "m") {window.location.href = "/menu";}
});
```

4.1.2.2 Xem thiết lập hệ thống/Điều chỉnh âm lượng và độ sáng

Trong Javascript, tạo nhạc nền cho game cũng như điều chỉnh âm lượng nhạc nền và độ sáng chỉ với một vài thủ thuật nho nhỏ và sẽ lưu trữ trong LocalStorage. Khởi tạo

biến audioSource để chỉ định nguồn nhạc nền được chọn, volumeBar và brightnessBar để chỉ định thanh bar cho người chơi điều chỉnh thay đổi theo sở thích. Đặt giá trị âm lượng và độ sáng có giá trị mặc định ban đầu lần lượt là 0.5 và 1.

```
$(document).ready(() => {
    $("#settingsButton").click(() => {$("#myModal1").modal("show");});
    // Volume and brightness features
    let audioSource = document.getElementById("audioSource");
    let volumeBar = document.getElementById("volumeBar");
    audioSource.volume = 0.5;
    if (!localStorage.getItem("audioVolume")) {
        localStorage.setItem("audioVolume", audioSource.volume);
    }
    // else {
        audioSource.volume = localStorage.getItem("audioVolume");
        volumeBar.value = audioSource.volume * 100;
    // }
    $("#volumeBar").change((event) => {
        audioSource.volume = event.target.value / 100;
        localStorage.setItem("audioVolume", audioSource.volume);
    });
    // Brightness feature
    let brightnessBar = document.getElementById("brightnessBar");
    let brightnessValue = 1;
    if (!localStorage.getItem("brightness")) {
        localStorage.setItem("brightness", brightnessValue);
    }
    else {
        brightnessValue = localStorage.getItem("brightness");
        document.documentElement.style.filter =
        "brightness(" + brightnessValue + ")";
```

```
            brightnessBar.value = brightnessValue * 100;
        }
        $("#brightnessBar").change((event) => {
            document.documentElement.style.filter =
            "brightness(" + event.target.value / 100 + ")";
            localStorage.setItem("brightness", event.target.value / 100);
        });
});
```

Kiểm tra nếu trong localStorage không có item nào thì setItem đó vào và ngược lại nếu đã có thì sử dụng giá trị của item đó từ localStorage. Giá trị của âm lượng và độ sáng thay đổi dựa trên sự thay đổi từ người chơi điều chỉnh thông qua change(event) - Event handler property để chia theo tỉ lệ phần trăm, lưu trữ lại localStorage đồng thời hiển thị phù hợp.

4.1.2.3 Thoát game

```
// Force setting modal to hide when open exit game modal
$("#exitGameConfirm").click(() => {
    $("#myModal1").modal("hide");
    $(".modal-backdrop").css("display", "none");
});
// Sign out
$("#signOutButton").click(async () => {
    const response = await fetch(`${DEV_URL}/signout`, {
        method: "POST",
    });
    const { code } = await response.json();
    if (code === "success") window.location.href = "/";
});
```

Chức năng thoát game cần được xác nhận do đó trong JavaScript cần hiện tại thêm modal xác nhận thoát. Khi người chơi xác nhận thoát thì sẽ fetch đến đường dẫn có route signout theo phương thức POST đã được tạo ở phía trên (mục 4.1.2.1) để thực hiện việc xóa Session.

4.1.3 Về Chức năng trong game

Sau các phần về tài khoản và thiết lập của game đã nêu ở 4.1.1 và 4.1.2, nhóm xin phép đi tiếp vào phần trọng tâm trong game tại mục 4.1.3 này với các nội dung về nhân vật như lựa chọn và vẽ nhân vật trong từng bản đồ, điều khiển nhân vật di chuyển và realtime cho nhiều nhân vật cùng lúc, có thể chat trực tuyến giữa các người chơi với nhau, các thao tác về game mini trong game, dữ liệu thông số về kinh nghiệm, thể lực, xu của người chơi từ game mini được tính toán như thế nào, bảng xếp hạng người chơi theo cấp độ,... sẽ được trình bày chi tiết.

4.1.3.1 Chơi game/Lựa chọn nhân vật

Tạo một GameController dùng để điều phối hiển thị dữ liệu và bản đồ các phòng chơi của người chơi. Với Index() là sảnh chính và các Room là các căn phòng bí mật. Hàm SendUserInfo() phục vụ cho việc vận chuyển thông tin của người chơi thông qua các ViewBag.

```
public class GameController : Controller
{
        private GameOnlineContext _context;
        public GameController(GameOnlineContext context){
                _context = context;
        }
        public IActionResult Index([FromQuery(Name = "gd")]String gd){
                // This is Hall map
                if(HttpContext.Session.GetString("username") != null){
```

```
            SendUserInfo();return View();
        }
        return Redirect("/signin");
}

[Route("userinfo")]
[HttpGet]
public IActionResult GetUserInfo()
{
        if (HttpContext.Session.GetString("username") != null){
            var userDAO = new UserDAO(_context);
            var userInfoDAO = new UserInfoDAO(_context);
            var username = HttpContext.Session.GetString("
            username");
            var user = userDAO.getByUsername(username);
            var userInfo = userInfoDAO.getById(user.UserId);
            return Ok(new {
                code = "success",
                userId = user.UserId,
                infoId = userInfo.UserInfoId,
                level = userInfo.Level,
                stamina = userInfo.Stamina,
                coin = userInfo.Coin, exp = userInfo.Exp
            });
        }
        return Redirect("/signin");
}

[Route("userinfo/update")]
[HttpPost]
public IActionResult SetUserInfo([FromBody] UserInfoViewModel
```

```
userInfo)
{
        if (HttpContext.Session.GetString("username") != null){
                var userInfoDAO = new UserInfoDAO(_context);
                var infoId = Guid.Parse(userInfo.UserInfoId);
                var newInfo = new UpdateUserInfoViewModel(){
                        Stamina = userInfo.Stamina,
                        Coin = userInfo.Coin,
                        Level = userInfo.Level,
                        Exp = userInfo.Exp,
                };
                userInfoDAO.updateUserInfo(newInfo, infoId);
                return Ok(new {code = "success"});
        }
        return BadRequest(new { code = "failed"});
}

public IActionResult GreenRoom(){
        // This is GreenRoom map
        if (HttpContext.Session.GetString("username") != null){
                SendUserInfo();return View();
        }
        return Redirect("/signin");
}
public IActionResult RedRoom(){
        // This is RedRoom map
        if (HttpContext.Session.GetString("username") != null){
                SendUserInfo();return View();
        }
        return Redirect("/signin");
}
```

```
        public IActionResult BlueRoom(){
            // This is BlueRoom map
            if (HttpContext.Session.GetString("username") != null){
                SendUserInfo();return View();
            }
            return Redirect("/signin");
        }
        public IActionResult PurpleRoom(){
            // This is PurpleRoom map
            if (HttpContext.Session.GetString("username") != null){
                SendUserInfo();return View();
            }
            return Redirect("/signin");
        }
        private void SendUserInfo(){
            var userDAO = new UserDAO(_context);
            var userInfoDAO = new UserInfoDAO(_context);
            var username = HttpContext.Session.GetString("username");
            var user = userDAO.getByUsername(username);
            var userInfo = userInfoDAO.getById(user.UserId);
            var exp = userInfo.Exp % 1000;
            var expPercent = (int)Math.Round((double)(100 * exp) / 1000);
            ViewBag.username = user.Username;
            ViewBag.exp = expPercent;
            ViewBag.coin = userInfo.Coin;
            ViewBag.stamina = userInfo.Stamina;
            ViewBag.level = userInfo.Level;
        }
    }
```

4.1.3.2 Điều khiển nhân vật và realtime nhiều người chơi

Đầu tiên cần khai báo một số thứ trong JavaScript.

```
var canvas = document.getElementById(mapName[index]);
var ctx = canvas.getContext("2d");
canvas.height = 478;
canvas.width = 928;
```

Biến canvas dùng để chỉ định chính xác hình bản đồ sẽ hiển thị theo dạng 2d dựa trên index và có giá trị từ 0 đến 4. Với giá trị 0 là sảnh Hall và 1 đến 4 là các căn phòng. Khởi tạo kích thước cho bản đồ đó lần theo chiều cao 478 x chiều rộng 928.

```
// Default positions of player in each room
var keys = [];
var player;
```

Biến keys để nhận biết phím mà người chơi nhấn trong việc di chuyển nhân vật. Biến player sẽ lưu trữ thông tin về vị trí người chơi theo tọa độ chỉ định, các frame, tốc độ và chuyển động của người chơi.

```
if (index == 0) {
        player = {x: 449, y: 200, width: 32, height: 48,
        frameX: 0, frameY: 0, speed: 10, moving: false,};
}
else if (index == 1) {
        player = {x: 865, y: 59, width: 32, height: 48,
        frameX: 0, frameY: 0, speed: 10, moving: false,};
}
else if (index == 2) {
        player = {x: 30, y: 417, width: 32, height: 48,
        frameX: 0, frameY: 0, speed: 10, moving: false,};
}
else if (index == 3) {
        player = {x: 865, y: 417, width: 32, height: 48,
```

```
        frameX: 0,  frameY: 0, speed: 10, moving: false,};
}
else if (index == 4) {
        player = {x: 30, y: 30, width: 32, height: 48,
        frameX: 0, frameY: 0, speed: 10, moving: false,};
}
```

Hàm if() ở đây sẽ kiểm tra index có giá trị bao nhiêu để hiển thị view bản đồ và vị trí nhân vật của người chơi. Sau đó, xác định đường dẫn hình ảnh nhân vật nam hoặc nữ từ url để hiển thị cho đúng với lựa chọn nhân vật của người chơi.

```
// Get character gender from URL
var params = new URLSearchParams(window.location.search);
let playerSpriteURL;
if (params.get("gd") == "male") {
        playerSpriteURL = "../assets/indianajones.png";
} else {
        playerSpriteURL = "../assets/marionravenwood.png";
}
```

Ta thực hiện việc đọc giá trị của bàn phím khi nhấn tổ hợp phím [←][↑][↓][→] hoặc tổ hợp phím [W][A][S][D], cấp trạng thái chuyển động cho nhân vật là true và ngược lại sẽ xóa giá trị phím đọc được và cấp trạng thái chuyển động cho nhân vật là false.

```
// Configure player movements (using W-A-S-D keys or arrow keys)
window.addEventListener("keydown", (e) => {
        keys[e.key] = true; player.moving = true;
});
window.addEventListener("keyup", (e) => {
        delete keys[e.key]; player.moving = false;
});
```

Hàm movePlayer() để điều khiển di chuyển nhân vật trong bản đồ. Ta sẽ cho vị trí x của nhân vật trừ cho tốc độ di chuyển ở phía trên là 10 sẽ ra vị trí x hiện tại. Đồng thời fetch() đến /multiplayer/update theo phương thức POST để nhờ body gửi thông tin vị trí đó lên server qua biến player nhằm mục đích cập nhật lại vị trí hình ảnh nhân vật.

```
function movePlayer(player) {
    if ((keys["ArrowLeft"] || keys["a"]) && player.x > 0) {
        player.x -= player.speed;
        player.frameY = 1; player.moving = true;
        fetch(`${DEV_URL}/multiplayer/update`, {
            headers: {"Content-Type": "application/json",},
            method: "POST",
            body: JSON.stringify({player,
                username: document.getElementById("
                playerInfoUsername").textContent,
            }),
        });
    }
    if ((keys["ArrowRight"] || keys["d"]) &&
        player.x < canvas.width - player.width) {
        player.x += player.speed;
        player.frameY = 2; player.moving = true;
        fetch(`${DEV_URL}/multiplayer/update`, {
            headers: {"Content-Type": "application/json",},
            method: "POST",
            body: JSON.stringify({player,
                username: document.getElementById(
                "playerInfoUsername").textContent,
            }),
        });
    }
```

```
        if ((keys["ArrowUp"] || keys["w"]) && player.y > 0) {
            player.y -= player.speed;
            player.frameY = 3; player.moving = true;
            fetch(`${DEV_URL}/multiplayer/update`, {
                headers: {"Content-Type": "application/json",},
                method: "POST",
                body: JSON.stringify({player,
                    username: document.getElementById(
                    "playerInfoUsername").textContent,
                }),
            });
        }
        if ((keys["ArrowDown"] || keys["s"]) &&
            player.y < canvas.height - player.height) {
            player.y += player.speed;
            player.frameY = 0; player.moving = true;
            fetch(`${DEV_URL}/multiplayer/update`, {
                headers: {"Content-Type": "application/json",},
                method: "POST",
                body: JSON.stringify({player,
                    username: document.getElementById(
                    "playerInfoUsername").textContent,
                }),
            });
        }
    }
```

Khai báo biến players để lưu trữ tất cả các dữ liệu người chơi. Hàm getActivePlayers sẽ fetch() đến /multiplayer để lấy toàn bộ những người chơi đang hoạt động trong game và push vào biến chung players.

```
// Players list to store all players drawing data
let players = [];
// Get all active players
const getActivePlayers = async () => {
        const response = await fetch(`${DEV_URL}/multiplayer`);
        const { activePlayers } = await response.json();
        players.push(...activePlayers);
};
```

Tiếp theo ta cần vẽ bản đồ và các nhân vật. Khởi tạo biến map từ Image(). Sau đó dùng canvas context 2D ctx phía trên để đưa hình bản đồ vào canvas. Việc vẽ vào canvas do drawImage() đảm nhận.

```
// Draw map
var map = new Image();
map.src = "../assets/" + mapName[filteredPlayers[0].mapIndex] + ".png";
ctx.drawImage(map, 0, 0);
// Draw players
for (let player of filteredPlayers) {
        // Draw player image
        let newPlayerSprite = new Image();
        newPlayerSprite.src = player.playerSpriteURL;
        ctx.drawImage(newPlayerSprite,
                player.player.width * player.player.frameX,
                player.player.height * player.player.frameY,
                player.player.width, player.player.height,
                player.player.x, player.player.y,
                player.player.width, player.player.height
        );
```

Sau khi vẽ xong, ta cần phân biệt các nhân vật của người chơi với nhau. Điều đầu tiên cần làm chính là cho hiển thị tên người chơi trên từng nhân vật. Chọn font Poppins, cỡ chữ 14px, in đậm 500, vị trí ở giữa. Đối với người chơi chính thì tên có màu đỏ, các người chơi còn lại có màu trắng.

```
        // Draw player username
        ctx.font = "500 14px Poppins";
        ctx.textAlign = "center";
        if (player.username ===document.getElementById("
        playerInfoUsername").textContent) {
                ctx.fillStyle = "red"; ctx.fillText(player.username,
                player.player.x + 15, player.player.y - 5);
        } else {
                ctx.fillStyle = "white"; ctx.fillText(player.username,
                player.player.x + 15, player.player.y - 5);
        }
}
```

Còn trong ASP.NET, ta cần cấu hình kết nối SignalR cho việc realtime nhiều nhân vật của các người chơi cùng thời điểm với người chơi chính.

```
// Configure SignalR connection for multiplayer
const signalRMultiplayerConnection = new signalR.HubConnectionBuilder()
        .withUrl("/hubs/multiplayer")
        .configureLogging(signalR.LogLevel.Information)
        .build();
```

Một vấn đề cực kì quan trọng khi thao tác với nhiều người chơi cùng lúc để được realtime ta cũng cần khai báo một biến chung players trong lớp Global dưới dạng List. Biến này sẽ hiểu là trung gian cho việc giao tiếp qua lại giữa client-server thông qua kết nối SignalR realtime liên tục.

```
public static class Global
{
        // Global list to store all active players
        public static List<JoinGameViewModel> players = new
        List<JoinGameViewModel>();
 }
```

Trong JavaScript, tạo một hàm bất đồng bộ multiplayerStart() sẽ dùng để start() cho việc kết nối nhiều người chơi. Gửi yêu cầu tham gia vào trong game server bằng cách fetch() đến /multiplayer/active theo phương thức POST, gửi một body chứa thông tin người chơi, đường dẫn hình ảnh nhân vật và bản đồ theo index.

```
// Start multiplayer connection
async function multiplayerStart() {
        try {
                await signalRMultiplayerConnection.start();
                // Send request to join in game server
                await fetch(`${DEV_URL}/multiplayer/active`, {
                        headers: {"Content-Type": "application/json",},
                        method: "POST",
                        body: JSON.stringify({
                                player, playerSpriteURL, mapIndex: index,
                        }),
                });
        } catch (error) {console.error(error);setTimeout(start, 3000);}
}
multiplayerStart();
```

Khi người chơi hoạt động sẽ có trạng thái là active, khi người chơi mất kết nối hoặc thoát game sẽ có trạng thái là inactive. Giả sử trường hợp người chơi bị mất kết nối thì

ta sẽ xóa người chơi khỏi game bằng cách fetch() đến /multiplayer/inactive. Thêm người chơi mới bằng cách push() người chơi vào biến chung players khi kết nối SignalR của client lắng nghe được AddPlayer. Xóa người chơi đó khi lắng nghe được RemovePlayer và Cập nhật khi lắng nghe được UpdatePositions.

```
// Catch if player is disconnected then remove player out of the game
window.addEventListener("beforeunload", async (event) => {
    await fetch(`${DEV_URL}/multiplayer/inactive`);
});
// Event listener to add new player to the game
signalRMultiplayerConnection.on("AddPlayer", (player) => {
    players.push(player);
});
// Event listener to remove player from the game when disconnected
signalRMultiplayerConnection.on("RemovePlayer", (username) => {
    players = players.filter((player) => player.username !== username);
});
// Event listener to update positions of players
signalRMultiplayerConnection.on("UpdatePositions", (updatedPlayers) => {
    players = updatedPlayers;
});
```

Với AddPlayer, RemovePlayer và UpdatePositions được khai báo rất đơn giản trong ASP.NET chỉ với một cái Hub của MultiplayerClient như sau:

```
public class MultiplayerHub : Hub<IMultiplayerClient>{}
public interface IMultiplayerClient{
    Task AddPlayer(JoinGameViewModel player);
    Task RemovePlayer(string username);
    Task UpdatePositions(List<JoinGameViewModel> players);
}
```

Để biết được cách thức hoạt động của từng route cho việc realtime nhiều nhân vật mà khi nãy ta dùng các hàm fetch() phía trên cũng như các SignalR client lắng nghe, thì những điều đó ở phía SignalR server sẽ được trình bày ở đoạn này.

Tạo một MultiplayerController có các route multiplayer, multiplayer/active, multiplayer/inactive, multiplayer/update.

```
public class MultiplayerController : Controller
{
    private readonly IHubContext<MultiplayerHub, IMultiplayerClient>
    _hubContext;
    public MultiplayerController(IHubContext<MultiplayerHub,
    IMultiplayerClient> hubContext){_hubContext = hubContext;}
    // Endpoint to get active players of the game
    [Route("multiplayer")]
    [HttpGet]
    public IActionResult GetActivePlayers(){
        if (HttpContext.Session.GetString("username") != null){
            return Ok(new {
                code = "success", activePlayers = Global.players
            });
        }
        return Redirect("/signin");
    }
```

Ở endpoint đầu tiên theo phương thức GET này chỉ đơn giản là lấy ra các người chơi đang hoạt động trong game. Đối với endpoint thứ hai theo phương thức POST, sử dụng biến chung players để Add() các dữ liệu của người chơi vào, sau đó emit cho SignalR biết để bên client nhận được dữ liệu mới từ AddPlayer vào players bên client. Đồng thời gửi message rằng người chơi có username đó đã vào game.

```
// Endpoint to add player to the game
[Route("multiplayer/active")]
[HttpPost]
public async Task<IActionResult> JoinGame([FromBody]
JoinGameViewModel player){
        if (HttpContext.Session.GetString("username") != null){
                var username =
                HttpContext.Session.GetString("username");
                Global.players.Add(new JoinGameViewModel(){
                        Username = username,
                        Player = player.Player,
                        PlayerSpriteURL = player.PlayerSpriteURL,
                        MapIndex = player.MapIndex
                });
                await _hubContext.Clients.All.AddPlayer(
                new JoinGameViewModel(){
                        Username = username,
                        Player = player.Player,
                        PlayerSpriteURL = player.PlayerSpriteURL,
                        MapIndex = player.MapIndex
                });
                return Ok(new { code = "success",
                        message = username + " has joined"
                });
        }
        return Redirect("/signin");
}
```

Endpoint cho inactive theo phương thức GET để xóa người chơi cần chỉ định chính xác người chơi cần xóa. Biến playerToRemove cho ra kết quả người chơi cần xóa, Where() sẽ thực hiện việc tìm trong biến chung players xem có username của player

nào trùng khớp thì lấy nó ra. Hàm Remove() sẽ xóa bỏ nó. Sau đó emit cho SignalR biết để bên client nhận được dữ liệu mới từ RemovePlayer vào players bên client. Đồng thời gửi message rằng người chơi có username đó đã thoát game.

```
// Endpoint to remove player out of the game
// when player is disconnected
[Route("multiplayer/inactive")]
[HttpGet]
public async Task<IActionResult> LeaveGame()
{
    if (HttpContext.Session.GetString("username") != null){
        var username =
        HttpContext.Session.GetString("username");
        var playerToRemove = Global.players.Where(
            player => player.Username ==
        username).FirstOrDefault();
        Global.players.Remove(playerToRemove);
        await _hubContext.Clients.All.RemovePlayer(username);
        return Ok(new {
            code = "success",
            message = username + " has left"
        });
    }
    return Redirect("/signin");
}
```

Endpoint cho việc cập nhật vị trí nhân vật theo phương thức POST, sử dụng tiếp tục biến dùng chung players với một hàm lặp foreach để lấy ra vị trí và view nhân vật. Sau đó emit cho SignalR biết để bên client nhận được dữ liệu mới từ UpdatePositions vào players bên client. Đồng thời gửi message rằng vị trí đã được cập nhật.

```
        // Endpoint to update positions of all players in the game
        [Route("multiplayer/update")]
        [HttpPost]
        public async Task<IActionResult> UpdatePositions([FromBody]
        UpdatePositionsViewModel updatePositionsViewModel){
            if (HttpContext.Session.GetString("username") != null){
                foreach (var p in Global.players){
                    if (p.Username ==
                    updatePositionsViewModel.Username){
                        p.Player =
                        updatePositionsViewModel.Player;
                    }
                }
                await _hubContext.Clients.All.UpdatePositions(
                Global.players);
                return Ok(new { code = "success",
                    message = "All positions of players are updated"
                });
            }
            return Redirect("/signin");
        }
    }
```

4.1.3.3 Xem thông tin người chơi và bảng xếp hạng

Về ý nghĩa, bảng xếp hạng đã được tích hợp vào xem thông tin người chơi. Trong JavaScript, tạo phím tắt "I" để bật nhanh modal playerInfo. Hàm renderActivePlayers() sẽ hiển thị danh sách các người chơi đang hoạt động vào trong này.

```
    // Press I to open player info modal
    $(document).keydown((event) => {
```

```
    if (event.key === "i") {
        if ($("#playerInfoModal").modal("hide")) {
            $("#playerInfoModal").modal("show");
            renderActivePlayers(players);
        } else if ($("#playerInfoModal").modal("show")) {
            $("#playerInfoModal").modal("hide");
        }
    }
});
```

Ở bảng xếp hạng, có hỗ trợ chức năng tìm kiếm. Dễ dàng lọc ra danh sách theo chữ cái đầu cho đến toàn bộ tên người chơi. Và danh sách luôn luôn được lọc theo cấp độ - Level của người chơi từ cao xuống thấp.

```
// Search for active players in list
$("#searchPlayerInput").change((event) => {
    const searchInput = event.target.value;
    if (searchInput) {
        const results = players.filter((player) =>
        player.username.includes(searchInput));
        if (results.length) {renderActivePlayers(results);}
        else {$("#activePlayersList").html("");
            $("#activePlayersList").append(`
            <li class="d-flex align-items-center justify-
            content-center pr-2">
            <span>No players found</span>
            </li>`);
        }
    } else {renderActivePlayers(players);}
});
```

4.1.3.4 Chat trực tuyến

Cũng tương tự như cấu hình kết nối SignalR cho việc realtime nhiều nhân vật ở trên, ta cấu hình kết nối SignalR cho chat nhiều người cùng thời điểm như sau:

```
// Configure SignalR connection for chat
const signalRChatConnection = new signalR.HubConnectionBuilder()
    .withUrl("/hubs/chat")
    .configureLogging(signalR.LogLevel.Information)
    .build();
```

Bắt đầu việc kết nối cho chat:

```
// Start connection
async function start() {
    try {await signalRChatConnection.start();}
    catch (error) {console.error(error);setTimeout(start, 3000);}
}
start();
```

SignalR client của chat này cũng sẽ lắng nghe các message nhận được từ ReceiveMessage và append() vào trong khung chat các thông tin về thời gian, username và nội dung của message.

```
// Event listener to render message
signalRChatConnection.on("ReceiveMessage", (message) => {
    $("#chatBox").append(`
        <li class="chat-message">
        <span class="chat-message-time" title="${new Date(
            message.createdAt
            ).toLocaleTimeString()}">
            [${new Date(message.createdAt).toDateString()}]
        </span>
```

```
                <span class="chat-message-username">${message.username}
                </span>
                <span class="chat-message-content">${message.content}
                </span>
                </li>
            `);
            $("#chatInput").val("");
            document.getElementById("chatBox").scrollTop =
            document.getElementById("chatBox").scrollHeight;
    });
```

Với ReceiveMessage được khai báo cũng rất đơn giản trong một cái Hub của ChatClient như sau:

```
    public class ChatHub : Hub<IChatClient>{}
    public interface IChatClient{Task ReceiveMessage(Message message);}
```

Tạo một MessageController dùng để lấy các messages và tạo message mới, gồm route message theo phương thức GET và message/create theo phương thức POST.

```
    public class MessageController : Controller
    {
        private readonly GameOnlineContext _context;
        private readonly IHubContext<ChatHub, IChatClient> _hubContext;
        public MessageController(GameOnlineContext context,
        IHubContext<ChatHub, IChatClient> hubContext){
            _context = context;_hubContext = hubContext;
        }
        [Route("message")]
        [HttpGet]
        public IActionResult GetAllMessages()
```

```
{
        if (HttpContext.Session.GetString("username") != null){
                var messageDAO = new MessageDAO(_context);
                var messages = messageDAO.GetAllMessages();
                return Ok(new { code = "success", messages });
        }
        return Redirect("/signin");
}
[Route("message/create")]
[HttpPost]
public async Task<IActionResult> CreateMessage([FromBody]
MessageViewModel message)
{
        if (HttpContext.Session.GetString("username") != null){
                var messageDAO = new MessageDAO(_context);
                var userDAO = new UserDAO(_context);
                var username =
                HttpContext.Session.GetString("username");
                var user = userDAO.getByUsername(username);
                var messageToCreate = new Message(){
                        MessageId = Guid.NewGuid(),
                        Uid = user.UserId,
                        Username = user.Username,
                        Content = message.Content,
                        CreatedAt = DateTime.Now
                };
                messageDAO.CreateMessage(messageToCreate);
                await _hubContext.Clients.All.ReceiveMessage(
                messageToCreate);
                return Ok(new { code = "success",
                        message = messageToCreate
```

```
                });
            }
            return Redirect("/signin");
        }
    }
```

Để tạo mới message cần có các thông tin về MessageId sinh mã theo Guid, Uid từ UserId, Username, Content chứa nội dung message input, CreatedAt là thời gian hiện tại của message được tạo. DAO tạo message và emit cho SignalR biết để bên client nhận được dữ liệu messageToCreate mới từ ReceiveMessage. Đồng thời gửi message cũng chính là messageToCreate.

4.1.3.5 Vào phòng bí mật

Để vào trong từng căn phòng bí mật, người chơi sẽ đến vị trí từng cánh cổng và nhấn phím "E". Trong JavaScript, khai báo tọa độ vị trí của cánh cổng vào redDoor và cổng ra redDoorBack.

```
// Red door enter and get back locations
var redDoor = {x: 96, y: 96,width: 32,height: 32,};
var redDoorBack = {x: 900,y: 415,width: 59,height: 63,};
```

Tiến hành đọc phím nhấn E và kiểm tra xem nếu đúng là vị trí của người chơi hợp lệ với vị trí cổng mà ta định trước thì cho vào hoặc cho ra tương ứng. Các phòng khác cũng được làm tương tự.

```
if (keys["e"] && x >= redDoor.x && x <= redDoor.x + redDoor.width &&
        y >= redDoor.y && y <= redDoor.y + redDoor.height) {
        window.location.pathname = "/Game/RedRoom";
}
if (keys["e"] && x >= redDoorBack.x && x <= redDoorBack.x +
        redDoorBack.width && y >= redDoorBack.y &&
        y <= redDoorBack.y + redDoorBack.height) {
        window.location.pathname = "/Game";
}
```

4.1.3.6 Chơi game mini

Ở phần này sẽ không đưa hết tất cả 4 game mini mà chỉ lấy ví dụ 1 game mini, các game mini còn lại được thực hiện tương tự, chỉ khác nhau một ít bởi vì luật chơi mỗi game mini là khác nhau thì cách tính điểm, cách điều khiển riêng cho từng game mini sẽ khác nhau.

Khái quát các phần trong việc hiện thực game mini gồm những bước như sau:
1.Khai báo ban đầu gồm canvas, biến chứa các trạng thái game mini là 0-hiện tại (current), 0-sẵn sàng (ready), 1-đang chơi (playing), 2-kết thúc (over), các biến chứa thông số người chơi được mặc định ban đầu bằng 0.
2.Lấy và cập nhật thông số người chơi qua CSDL.
3.Tạo switch để thực hiện các cases tương ứng với từng trạng thái.
4.Tạo các assets cho game mini và điều khiển chuyển động nhân vật.
5.Vẽ game mini và tính toán các thông số khi chơi.

Ví dụ trong trường hợp này là game mini “Cosmic Snake”:

```
const snakeCanvas = document.getElementById("snakeCanvas")
const snakeCtx = snakeCanvas.getContext("2d")
const gameState = {current: 0, ready: 0, playing: 1, over: 2,}
let userInfoId = ""
let stamina = 0
let level = 0
let exp = 0
let coin = 0
```

Tạo 2 hàm bất đồng bộ getUserInfo() và setUserInfo(). Hàm đầu tiên là lấy thông tin người chơi từ CSDL thông qua fetch đến /userinfo. Hàm thứ hai là cập nhật lại thông tin người chơi lên CSDL thông qua fetch đến /userinfo/update theo phương thức POST.

```
async function getUserInfo() {
    try {
```

```
            const response = await fetch(`${DEV_URL}/userinfo`);
            const res = await response.json()
            userInfoId = res.infoId
            stamina = res.stamina
            level = res.level
            exp = res.exp
            coin = res.coin
        }catch (err) {console.log(err)}
    }
    async function setUserInfo() {
        try {
            await fetch(`${DEV_URL}/userinfo/update`, {
                headers: {"Content-Type": "application/json",},
                method: "POST",
                body: JSON.stringify({
                    UserInfoId: userInfoId, Stamina: stamina,
                    Level: level, Exp: exp, Coin: coin
                }),
            })
        }catch (err) {console.log(err)}
    }
```

Tạo switch kiểm tra trạng thái hiện tại của game mini đang thao tác:

+ Với trạng thái hiện tại là 0-sẵn sàng, kiểm tra thể lực người chơi, nếu lớn hơn 10 nghĩa là đủ thể lực thì tiến hành trừ 10 điểm thể lực của người chơi, trạng thái hiện tại được chuyển sang trạng thái 1-đang chơi và tiến hành vẽ game, khi vẽ đã bao gồm việc điều khiển và tính toán thông số chơi game. Ngược lại, thông báo thể lực không đủ và kết thúc switch.

+ Với trạng thái hiện tại là 1-đang chơi, kết thúc switch để game mini được tiếp diễn.

+ Với trạng thái hiện tại là 2-kết thúc, trả về lại trạng thái ban đầu là 0-sẵn sàng, cũng như các thông số và vị trí được lại lại mặc định ban đầu.

```
snakeCanvas.addEventListener("click", e => {
    switch (gameState.current) {
        case gameState.ready:
            if (stamina >= 10) {stamina -= 10}
            else {console.log("You don't have enough stamina")
                break
            }
            gameState.current = gameState.playing
            game = setInterval(draw, 100)
            break
        case gameState.playing:
            break
        case gameState.over:
            gameState.current = gameState.ready
            drawReady()
            snake = []
            snake[0] = {x: 9 * box,y: 10 * box}
            score = 0
            direction = ""
            break
    }
})
```

4.1.3.7 Mua thể lực

Trong JavaScript, tạo phím tắt “B” để bật nhanh modal cửa hàng thể lực.

```
// Press B to stamina store modal
$(document).keydown((event) => {
    if (event.key === "b") {
        if ($("#storeModal").modal("hide")) {
            getUserInfo()
            setTimeout(() => $('#dp-coin').html("Coin: " + coin), 100)
```

```
                $("#storeModal").modal("show");
            }else if ($("#storeModal").modal("show")) {
                $("#storeModal").modal("hide");
            }
        }
});
```

Khi bật sẽ lấy thông tin người chơi và hiển thị tổng số xu mới nhất đang có. Trong cửa hàng thể lực có 3 gói lựa chọn là +10,+30,+50 điểm thể lực. Tương ứng với số tiền người chơi phải bỏ ra cho việc mua nó lần lượt là -10,-25,-30 xu. Khi click vào gói lựa chọn nào thì hệ thống sẽ kiểm tra số xu cũng như điểm thể lực để trả về messages phù hợp với người chơi. Ví dụ trong đoạn code về việc mua gói thể lực đầu tiên (+10 điểm thể lực) như sau:

```
$("#buy-sta-10").click(() => {
    if (coin >= 10) {
        if (stamina <= 290) {
            stamina += 10
            coin -= 10
            setUserInfo()
            $('#dp-stamina').html(stamina + "%")
            $('#dp-stamina-progress').css("width", stamina + "%")
            $('#dp-coin').html("Coin: " + coin)
            $('#dp-error-message').attr("class", "text-success")
            $('#dp-error-message').html("Purchased!!!")
        }else {
            $('#dp-error-message').attr("class", "text-danger")
            $('#dp-error-message').html("Your stamina is full")
        }
    }else {
        $('#dp-error-message').attr("class", "text-danger")
```

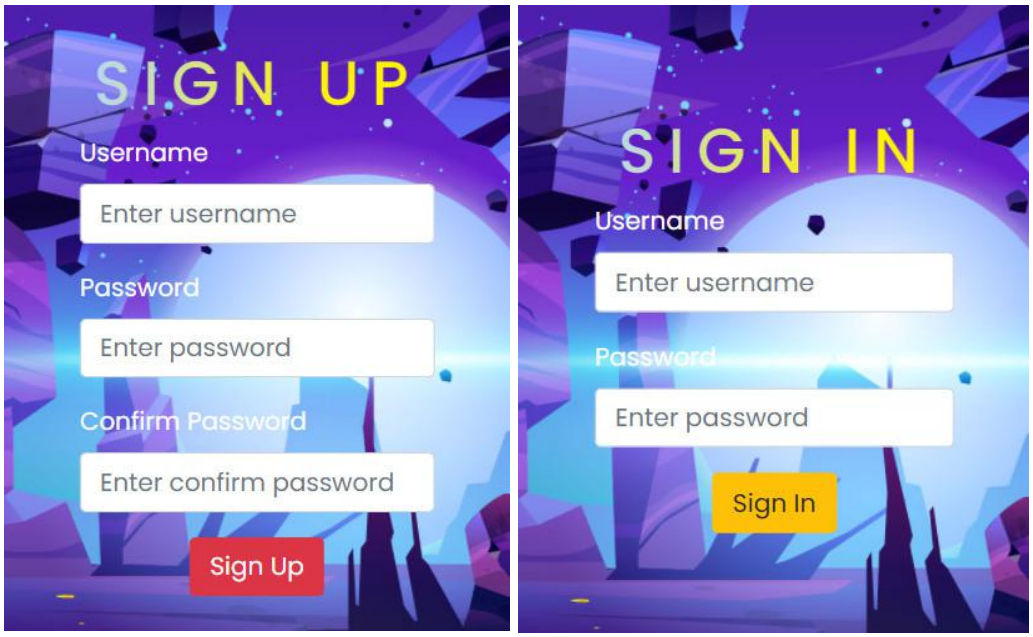
```
                $('#dp-error-message').html("Your coin is not enough")
            }
        });
```

Bên cạnh đó, nhóm còn bổ sung thêm việc tự động cộng điểm thể lực kể cả khi người chơi không hoạt động bằng cách gọi hàm StartAsync() của ASP.NET. Một biến đếm thời gian gọi là timer sẽ được chạy với thời gian được tính bằng giây.

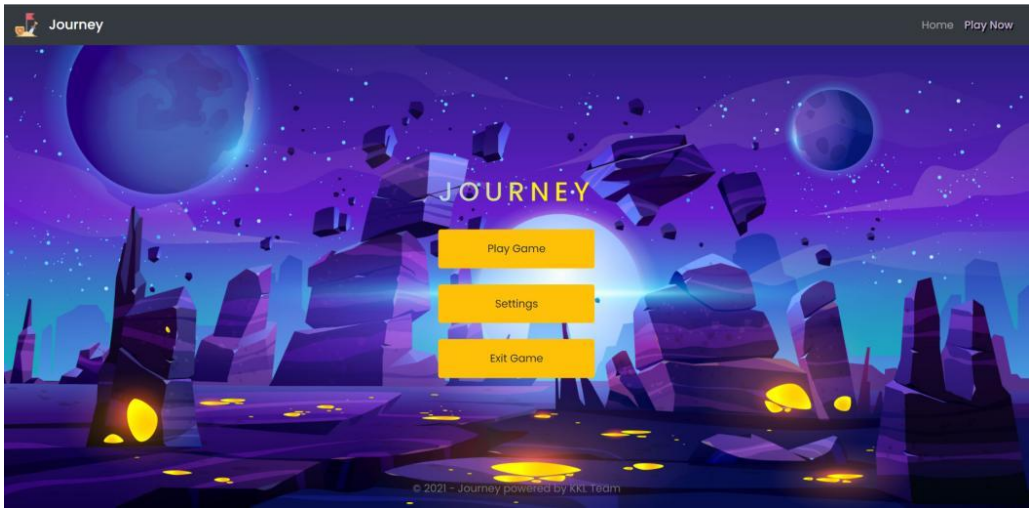
```
    public Task StartAsync(CancellationToken stoppingToken){
            _context = _scopeFactory.CreateScope().ServiceProvider
                    .GetRequiredService<GameOnlineContext>();
            _logger.LogInformation("Timed Hosted Service running.");
            _timer = new Timer(DoWork, null, TimeSpan.Zero,
                    TimeSpan.FromSeconds(3600));
             return Task.CompletedTask;
    }
    private void DoWork(object? state){
            var count = Interlocked.Increment(ref executionCount);
            _logger.LogInformation("Timed Hosted Service is working.
                    Count: {Count}", count);
            var userInfoDAO = new UserInfoDAO(_context);
            // Get list player with low stamina (stamina < 100)
            var user = userInfoDAO.getLowStaPlayers();
            user.ForEach(u => {
                    u.Stamina += 10; userInfoDAO.updateUserSta(u);
             });
    }
```

Hàm DoWork() sẽ thực hiện việc lấy danh sách những người chơi có điểm thể lực < 100. Khi thời gian đếm đủ 3600 giây thì thực hiện việc cộng 10 điểm thể lực và cập nhật lại tổng thể lực.

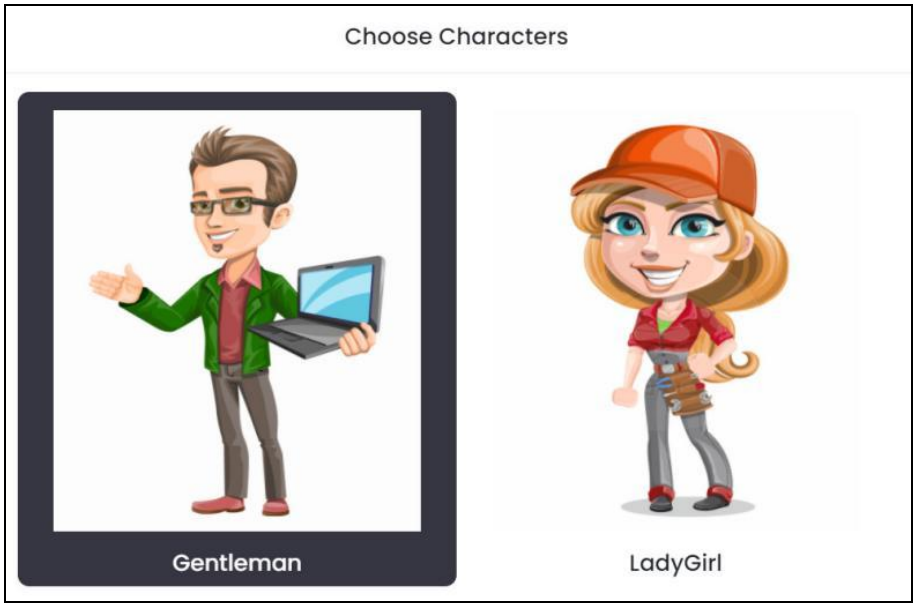
4.2 Kết quả đạt được



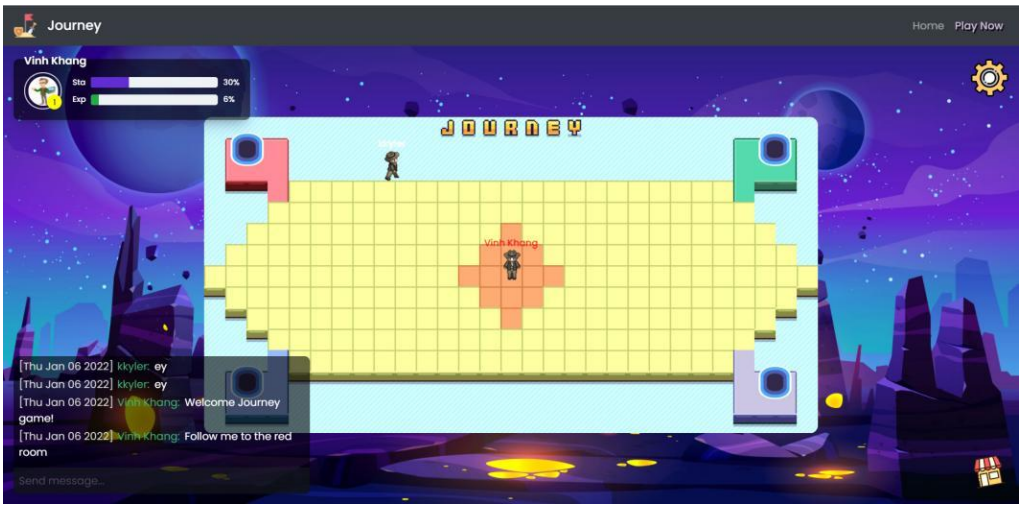
Hình 21. Hình ảnh về Đăng ký và Đăng nhập



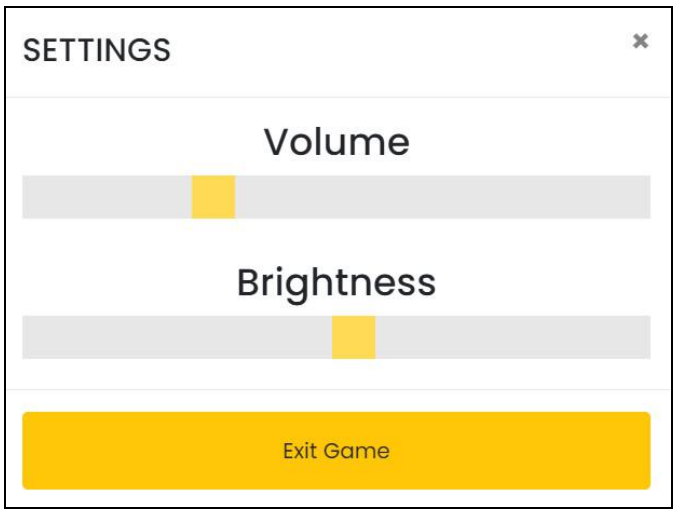
Hình 22. Hình ảnh về Menu game



Hình 23. Hình ảnh về Lựa chọn nhân vật



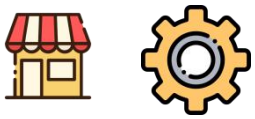
Hình 24. Hình ảnh về Giao diện game tại sảnh chính



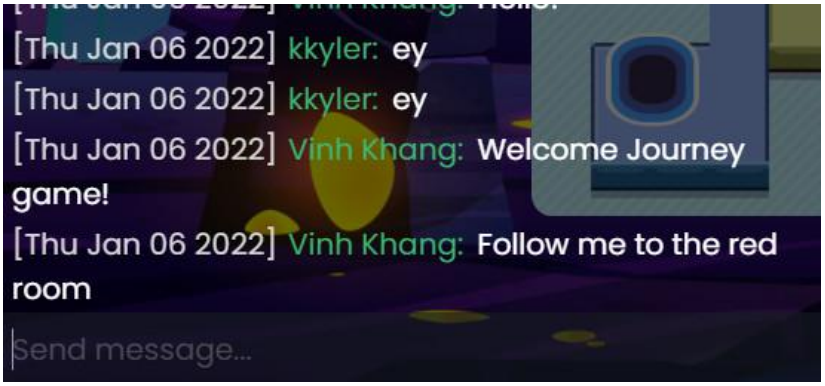
Hình 25. Hình ảnh về Điều chỉnh âm lượng và độ sáng

Hình 26. Hình ảnh về Nhân vật người chơi khi vào trong game

Hình 27. Hình ảnh về NPC Guide, Cổng dịch chuyển và ô Game Mini

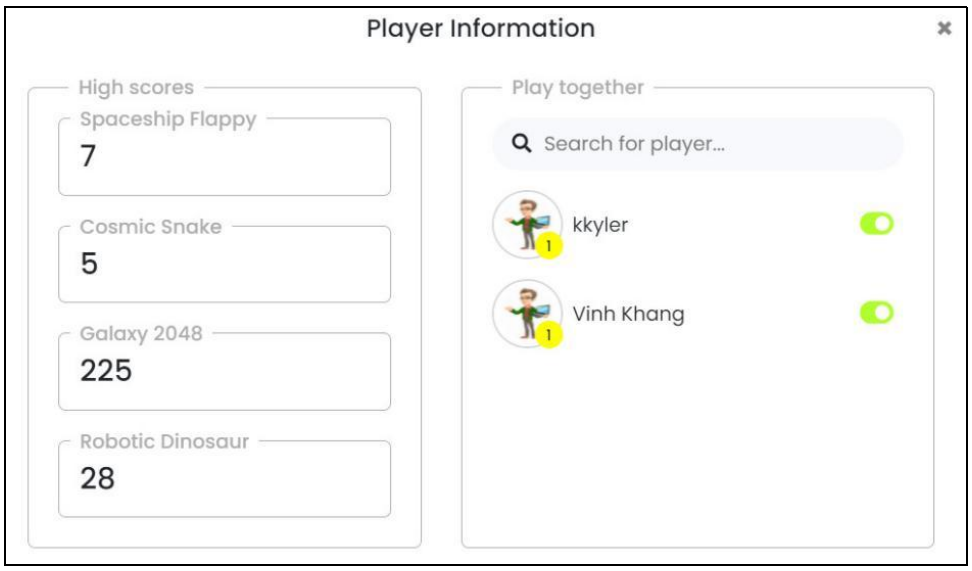


Hình 28. Hình ảnh về nút bật Cửa hàng thể lực và Thiết lập hệ thống

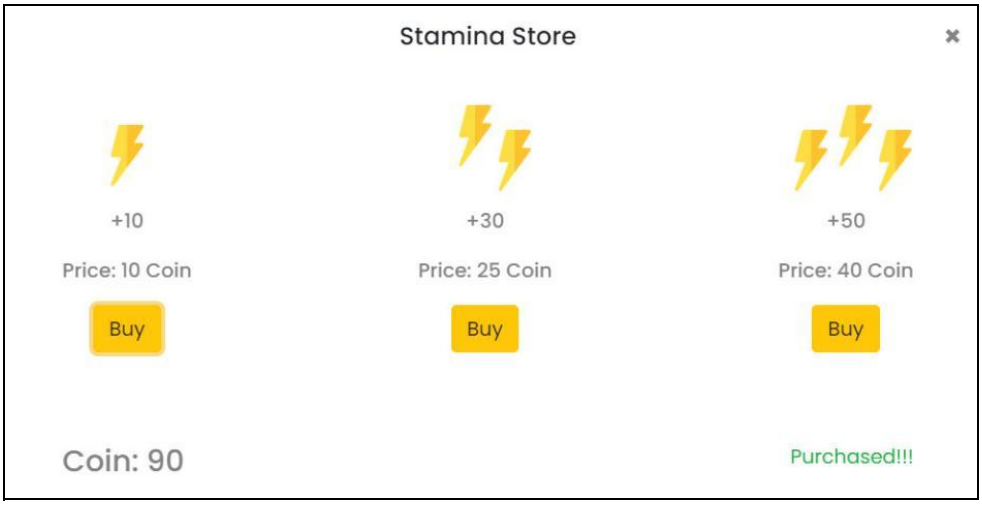


Hình 29. Hình ảnh về Khung chat trực tuyến

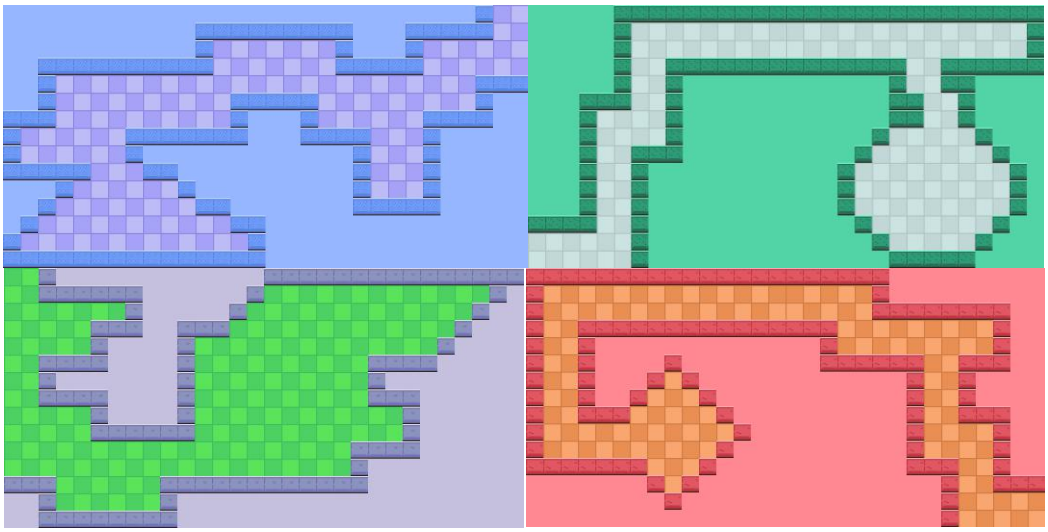
Hình 30. Hình ảnh về Thanh thông tin người chơi



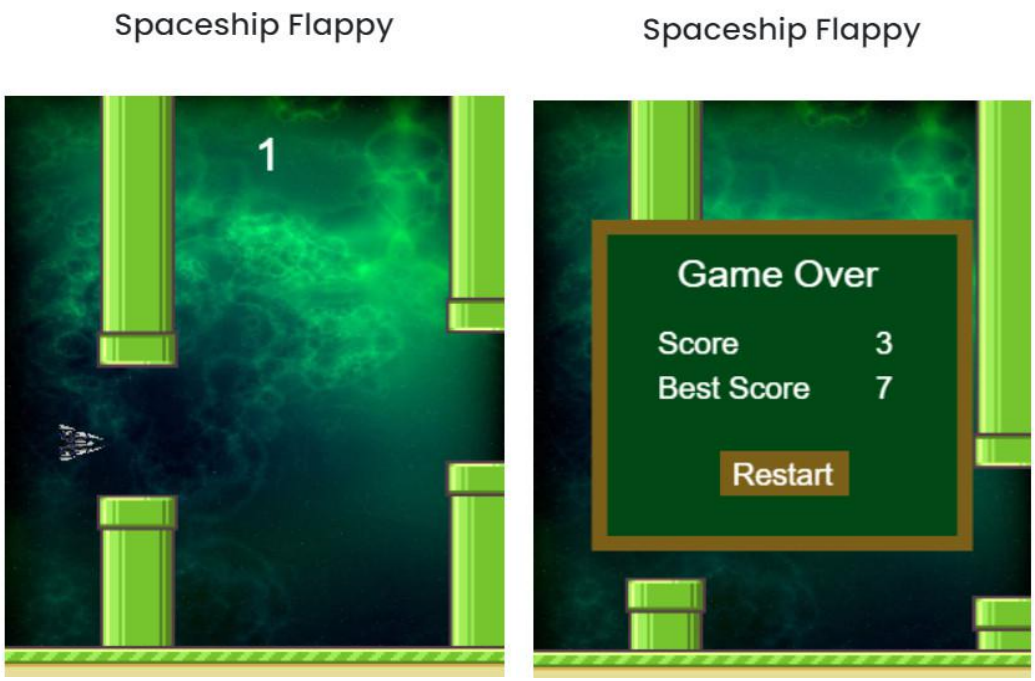
Hình 31. Hình ảnh về Điểm số cao nhất và Bảng xếp hạng



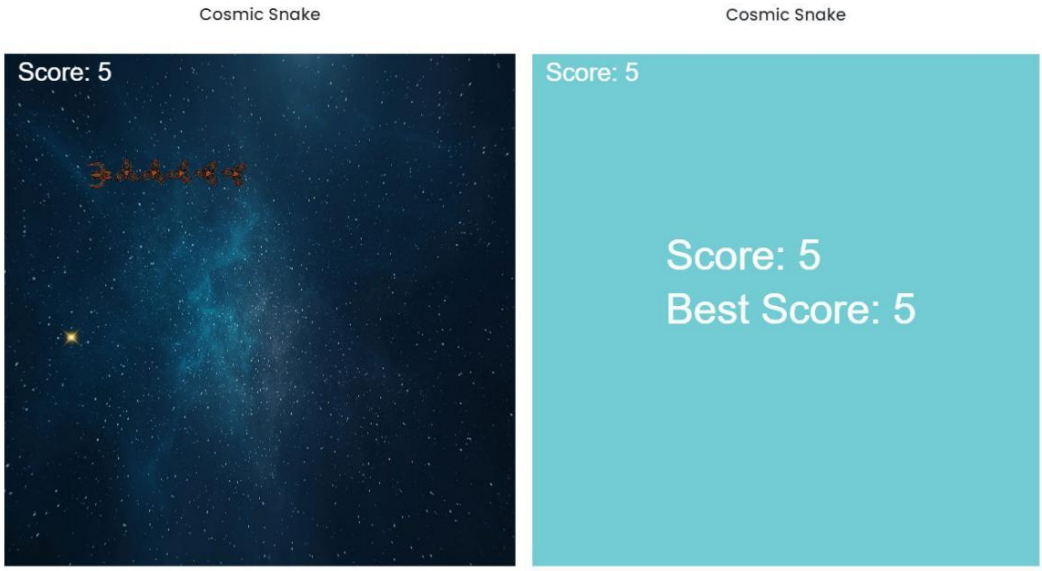
Hình 32. Hình ảnh về Mua thể lực trong Cửa hàng thể lực



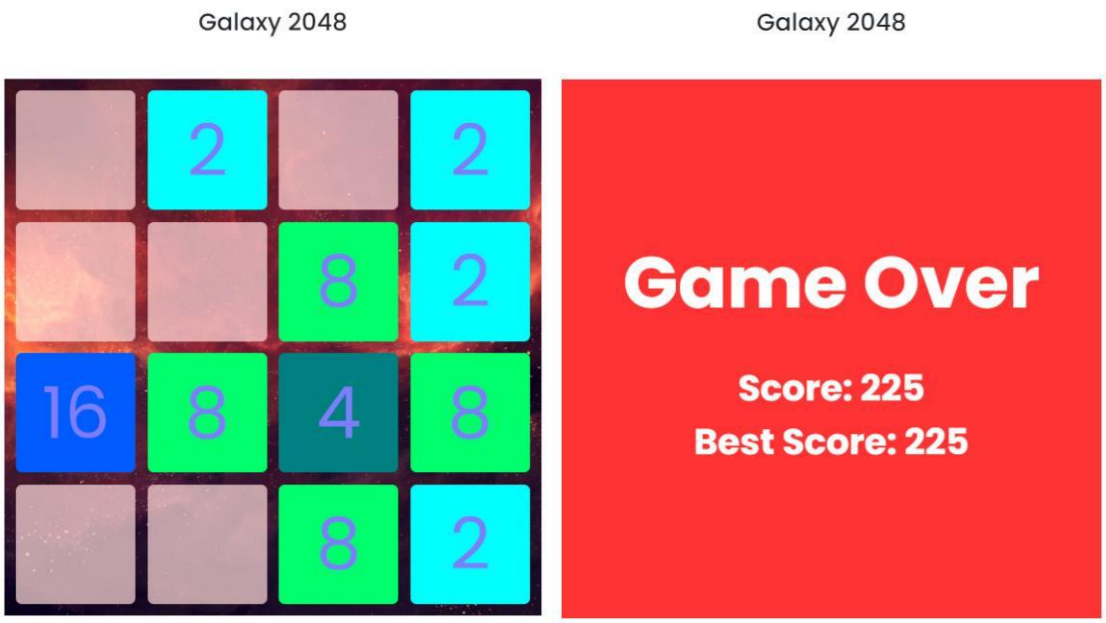
Hình 33. Hình ảnh về Bản đồ của 4 căn phòng bí mật



Hình 34. Hình ảnh về Game mini Spaceship Flappy



Hình 35. Hình ảnh về Game mini Cosmic Snake



Hình 36. Hình ảnh về Game mini Galaxy 2048

Hình 37. Hình ảnh về Game mini Robotic Dinosaur

CHƯƠNG V: TỔNG KẾT

Bài báo cáo này đã giới thiệu tổng quát nhất về Game online HTML5 kết hợp ASP.NET - Journey. Thông qua các use cases, đặc tả các use case cùng với các sơ đồ hoạt động và ERD đã minh họa cho việc hiện thực một web game online 2D một cách đơn giản nhất, dễ hiểu nhất. Đồng thời cũng giới thiệu qua về một thư viện SignalR rất có ích trong việc phát triển game online bằng ASP.NET. Cú pháp ngôn ngữ C# quen thuộc dễ sử dụng.

Nhóm đã hoàn thành cơ bản những chức năng cần thiết cũng như hiện thực một số ý tưởng mới lạ nhưng vẫn dễ hiểu dễ chơi ở mọi lứa tuổi. Tuy nhiên, việc triển khai gặp một số khó khăn nhất định vì phải tốn kha khá thời gian cho việc đọc tài liệu, hiện thực ý tưởng 4 game mini huyền thoại (Cosmic Snake, Galaxy 2048, Spaceship Flappy và Robotic Dinosaur) trong game chính, tìm hiểu cách sử dụng SignalR.

Trong tương lai, với thời gian thoải mái hơn thì nhóm sẽ hoàn thiện game những chức năng khác bổ sung thêm giúp game có nhiều thứ hay và tăng sự thích thú cho người chơi như nuôi thú cưng đi theo sau nhân vật, tăng thêm nhiều NPC, thay đổi trang phục cho nhân vật, mở rộng thêm nhiều căn phòng bí mật,…cũng như thiết kế xây dựng game tương thích với nhiều loại thiết bị khác nhau. Bên cạnh đó, nhóm đã phần nào tối ưu hóa được các đoạn code, ngắn gọn hơn, có khả năng cải tiến bổ sung chức năng cho game mini nếu cần. Ngoài ra, nhóm còn giúp cho trải nghiệm người dùng tốt hơn với việc tăng nhiều loại phím tắt đa dạng cho một số chức năng.

Nhóm cũng đã thử hosting trên web services AWS EC2 cho bài đồ án này, và nhận thấy vẫn còn nhiều bất cập, hệ thống khi development ở local thì hoạt động tốt nhưng khi đưa vào production thì không còn hoạt động ổn định nữa. Cụ thể là bị giật khá nhiều, có thể do việc hosting miễn phí nên chưa đủ mạnh.

Qua đề tài “Game online HTML5 kết hợp ASP.NET” thú vị này, nhóm đã tích lũy được rất nhiều kinh nghiệm trong việc tiếp cận công nghệ mới cũng như lập kế hoạch, lên ý tưởng cho game, hiện thực và xử lý các vấn đề nảy sinh trong quá trình phát triển phần mềm game online này.

TÀI LIỆU THAM KHẢO

[1] V.D. Hong, [2021], Slide Lecture Advanced Software Engineering, Ton Duc Thang University, Vietnam.

[2] B. Joshi, [2012], HTML5 Programming for ASP.NET Developers, USA.

[3] D. Roth, R. Anderson, [2021], Introduction to ASP.NET Core, USA.

[4] J. M. Aguilar, [2014], SignalR Programming in Microsoft ASP.NET, USA.

[5] K. Nayyeri, D. White, [2014], Pro ASP.NET SignalR, USA.

[6] J.W. Satzinger, R.B. Jackson, S.D. Burd, [2011], Systems Analysis and Design in a Changing World, Australia.

[7] A. Verma, Advanced Software Engineering, Jharkhand Rai University, India.

[8] Docker | https://docs.docker.com/

[9] Amazon EC2 | https://aws.amazon.com/ec2/

[10] Game Among Us | https://en.wikipedia.org/wiki/Among_Us

[11] Game Flappy Bird | https://en.wikipedia.org/wiki/Flappy_Bird

[12] Game Snake | https://en.wikipedia.org/wiki/Snake_(video_game_genre)

[13] Game 2048 | https://en.wikipedia.org/wiki/2048_(video_game)

[14] Game Dinosaur | https://en.wikipedia.org/wiki/Dinosaur_Game

[15] Draw Map 2D | https://codepen.io/punkydrewster713

[16] Characters | http://www.untamed.wild-refuge.net/rmxpresources.php?characters