TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

KHOA CÔNG NGHỆ THÔNG TIN

🙥🕮🙧



BÁO CÁO BÀI TẬP LỚN

PROJECT 1

NHÓM 5

ĐỀ TÀI: THIẾT KẾ VÀ XÂY DỰNG GAME AUTO-CAT

**Giảng viên hướng dẫn: Hoàng Văn Thông**

**Sinh viên thực hiện:**

|  |  |  |
| --- | --- | --- |
| STT | Họ và tên | Mã sinh viên |
| 1 | Nguyễn Quỳnh Chi |  |
| 2 | Đỗ Thị Hải | 211243882 |
| 3 | Trịnh Thành Nam |  |
| 4 | Nguyễn Hoài Sơn |  |

**Lớp: Công nghệ thông tin 1 – Khoá 62**

**Hà Nội – 2024**

LỜI CẢM ƠN

Chúng em xin chân thành cảm ơn!

**Nhóm sinh viên thực hiện**

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

Hà Nội, ngày…..tháng 12 năm 2023

Giảng viên hướng dẫn

MỤC LỤC

[LỜI MỞ ĐẦU 1](#_Toc184119673)

[CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI 2](#_Toc184119674)

[1.1. Đặt vấn đề 2](#_Toc184119675)

[1.2. Tổng quan về game đối kháng 2](#_Toc184119676)

[1.2.1. Lịch sử phát triển 2](#_Toc184119677)

[1.2.2. Đặc trưng của game đối kháng 3](#_Toc184119678)

[1.2.3. Các loại game đối kháng phổ biến 4](#_Toc184119679)

[1.2.4. Thị trường game đối kháng 4](#_Toc184119680)

[1.3. Tổng quan về AutoCat 4](#_Toc184119681)

[1.4. Nội dung đề tài 5](#_Toc184119682)

[1.5. Phạm vi đề tài 6](#_Toc184119683)

[1.5.1. Phạm vi thiết kế và phát triển 6](#_Toc184119684)

[1.5.2. Phạm vi công nghệ 6](#_Toc184119685)

[1.5.3. Giới hạn của đề tài 6](#_Toc184119686)

[1.5.4. Mục tiêu phát triển 7](#_Toc184119687)

[CHƯƠNG 2: KHẢO SÁT 8](#_Toc184119688)

[2.1. Mục tiêu khảo sát 8](#_Toc184119689)

[2.2. Khảo sát các game đối kháng tương tự 8](#_Toc184119690)

[2.2.1. Mục tiêu khảo sát: 8](#_Toc184119691)

[2.2.2. Phương pháp khảo sát 9](#_Toc184119692)

[2.3. Khảo sát nhu cầu người dùng 12](#_Toc184119693)

[2.3.1. Mục tiêu khảo sát 12](#_Toc184119694)

[2.3.2. Phương pháp khảo sát 12](#_Toc184119695)

[2.3.3. Nội dung khảo sát 13](#_Toc184119696)

[2.3.4. Phân tích dữ liệu 14](#_Toc184119697)

[CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ 18](#_Toc184119698)

[3.1. Phân tích đề tài 18](#_Toc184119699)

[3.1.1. Giới thiệu game AutoCat 18](#_Toc184119700)

[3.1.2. Kịch bản game 18](#_Toc184119701)

[3.1.3. Đặc tả yêu cầu 18](#_Toc184119702)

[3.2. Biểu đồ hành động 19](#_Toc184119703)

[3.2.1. Biểu đồ hành động đăng nhập và đăng ký 19](#_Toc184119704)

[3.2.2. Biểu đồ hành động mua và bán trang bị 21](#_Toc184119705)

[3.2.3. Biểu đồ hành động nâng cấp trang bị 23](#_Toc184119706)

[3.2.4. Biểu đồ hành động chế tạo xe chiến đấu 24](#_Toc184119707)

[3.2.5. Biểu đồ hành động ghép trận 25](#_Toc184119708)

[3.2.6. Biểu đồ hành động chiến đấu 25](#_Toc184119709)

[3.2.7. Biểu đồ hành động Sau chiến đấu 26](#_Toc184119710)

[3.2.8. Biểu đồ hành động leaderboard 27](#_Toc184119711)

[3.3. Thiết kế kiến trúc hệ thống 29](#_Toc184119712)

[3.3.1. Kiến trúc tổng thể 29](#_Toc184119713)

[3.3.2. Các thành phần trong hệ thống 29](#_Toc184119714)

[3.3.3. Mô hình dữ liệu 30](#_Toc184119715)

[3.3.4. Lưu đồ hoạt động của hệ thống Host-Client 30](#_Toc184119716)

[3.4. Thiết kế cơ sở dữ liệu 30](#_Toc184119717)

[3.5. Thiết kế giao diện người dùng (UI) 31](#_Toc184119718)

[3.5.1. Sơ đồ cấu trúc (Site Map) 31](#_Toc184119719)

[3.5.2. Wireframe các màn hình chính 31](#_Toc184119720)

[CHƯƠNG 4: THIẾT KẾ GAO DIỆN GAME 32](#_Toc184119721)

[4.1. Giao diện mở đầu 32](#_Toc184119722)

[4.2. Giao diện chiến đấu 32](#_Toc184119723)

[4.3. Giao diện mở rương 32](#_Toc184119724)

[CHƯƠNG 5: XÂY DỰNG CHƯƠNG TRÌNH 33](#_Toc184119725)

[5.1. Công nghệ sử dụng 33](#_Toc184119726)

[5.1.1. Netcode for GameObjects 33](#_Toc184119727)

[5.1.2. Unity Services (Unity Cloud) 34](#_Toc184119728)

[5.2. Công cụ và môi trường phát triển 35](#_Toc184119729)

[5.2.1. Rider IDE / Visual Studio 35](#_Toc184119730)

[5.2.2. Aseprite: 36](#_Toc184119731)

[5.3. Design Pattern 37](#_Toc184119732)

[5.3.1. Observer Pattern: 37](#_Toc184119733)

[5.3.2. Singleton Pattern: Thiết lập các module mà cho phép mọi module khác có quyền truy cập thông tin. 38](#_Toc184119734)

[5.3.3. State Pattern: Phần gameplay-loop, chia các phần công việc thành các trạng thái (state). 38](#_Toc184119735)

[5.4. Kỹ thuật sử dụng 39](#_Toc184119736)

[5.4.1. HeartbeatLobbyRequest: 39](#_Toc184119737)

[5.4.2. RefreshLobbyRequest: 39](#_Toc184119738)

[5.4.3. Remote Procedure Call (RPC): 40](#_Toc184119739)

[5.4.4. Client-side prediction: 41](#_Toc184119740)

[5.4.5. Server Reconciliation: 41](#_Toc184119741)

[5.5. Triển khai chức năng chính 42](#_Toc184119742)

[5.5.1. Tạo Lobby 42](#_Toc184119743)

[5.5.2. Tham gia Lobby thông qua mã code 42](#_Toc184119744)

[5.5.3. Quay lại Lobby sau khi thoát game 43](#_Toc184119745)

[5.6. Luồng hoạt động của game 43](#_Toc184119746)

[5.6.1. Từ lúc vào game cho tới khi thoát 43](#_Toc184119747)

[5.6.2. Quản lý trạng thái Lobby 44](#_Toc184119748)

[CHƯƠNG 6: KIỂM THỬ VÀ TỐI ƯU HÓA 46](#_Toc184119749)

[6.1. Các loại kiểm thử 46](#_Toc184119750)

[6.1.1. Kiểm thử chức năng 46](#_Toc184119751)

[6.1.2. Kiểm thử hiệu năng 46](#_Toc184119752)

[6.1.3. Kiểm thử tính cân bằng gameplay 46](#_Toc184119753)

[6.2. Tối ưu hóa hệ thống 46](#_Toc184119754)

[6.2.1. Tối ưu hóa hiệu suất game 46](#_Toc184119755)

[6.2.2. Cân bằng game 46](#_Toc184119756)

[CHƯƠNG 7: TỔNG KẾT VÀ ĐỊNH HƯỚNG PHÁT TRIỂN 47](#_Toc184119757)

[7.1. Tổng kết dự án 47](#_Toc184119758)

[7.2. Kết quả đạt được 47](#_Toc184119759)

[7.3. Hạn chế và bài học kinh nghiệm 47](#_Toc184119760)

[7.4. Định hướng phát triển trong tương lai 47](#_Toc184119761)

[KẾT LUẬN 48](#_Toc184119762)

[BẢNG PHÂN CÔNG CÔNG VIỆC 49](#_Toc184119763)

[TÀI LIỆU THAM KHẢO 50](#_Toc184119764)

MỤC LỤC HÌNH ẢNH

[Hình 1. Giao diện trận đấu game Robot Wars Arena 9](#_Toc184118733)

[Hình 2. Giao diện game Crossout 10](#_Toc184118734)

[Hình 3. Giao diện trận đấu trong game CrossOut 10](#_Toc184118735)

[Hình 4. Biểu đồ hành động đăng nhập 20](#_Toc184118736)

[Hình 5. Biểu đồ hành động đăng ký 21](#_Toc184118737)

[Hình 6. Biểu đồ hành động mua trang bị 22](#_Toc184118738)

[Hình 7. Biểu đồ hành động bán trang bị 23](#_Toc184118739)

[Hình 8. Biểu đồ hành động nâng cấp trang bị 24](#_Toc184118740)

[Hình 9. Biểu đồ hành động chế tạo xe chiến đấu 25](#_Toc184118741)

[Hình 10. Biểu đồ hành động ghép trận 25](#_Toc184118742)

[Hình 11. Biểu đồ hành động chiến đấu 26](#_Toc184118743)

[Hình 12. Biểu đồ hành động sau trận đấu 27](#_Toc184118744)

[Hình 13. Biểu đồ hành động leaderboard 28](#_Toc184118745)

[Hình 14. HeartbeatLobbyRequest 38](#_Toc184118746)

[Hình 15. RefreshLobbyRequest 39](#_Toc184118747)

[Hình 16. Server RPCs 39](#_Toc184118748)

[Hình 17. Client RPCs 40](#_Toc184118749)

LỜI MỞ ĐẦU

Ngày nay, công nghệ thông tin đã có những bước tiến mạnh mẽ, ảnh hưởng sâu rộng đến mọi lĩnh vực trong cuộc sống, từ giáo dục, y tế, cho đến giải trí. Trong đó, ngành công nghiệp game đã phát triển vượt bậc, không chỉ trong việc cung cấp những trò chơi giải trí mà còn là nơi thể hiện sự sáng tạo, khả năng lập trình và thiết kế. Game không chỉ mang lại những phút giây giải trí mà còn là công cụ giáo dục, phát triển kỹ năng và chiến lược cho người chơi.

Ở Việt Nam, ngành công nghiệp game đã có những bước phát triển mạnh mẽ trong những năm gần đây, đặc biệt là với sự gia tăng nhanh chóng của các trò chơi trên các nền tảng PC và mobile. Sự chuyển mình từ các trò chơi 2D đơn giản sang các game 3D với đồ họa sinh động, âm thanh sống động, và hệ thống gameplay phức tạp đã làm thay đổi hoàn toàn cách người chơi tiếp cận và trải nghiệm game.

Trong bối cảnh đó, nhóm chúng em đã quyết định thực hiện dự án "AutoCat" - một tựa game đối kháng độc đáo, để áp dụng những kiến thức đã học về lập trình game, thiết kế đồ họa và quản lý dự án vào một sản phẩm thực tế. Đây là một dự án đột phá, kết hợp giữa yếu tố xây dựng và chiến đấu tự động, cho phép người chơi tự tay tạo dựng những chiếc xe chiến đấu và tham gia vào những trận chiến kịch tính.

Game **AutoCat** không chỉ là một sản phẩm giải trí mà còn là cơ hội để nhóm vận dụng các kiến thức về lập trình, thiết kế game, cũng như phát triển các kỹ năng làm việc nhóm và quản lý dự án.Tuy nhiên, do thời gian và nguồn lực có hạn, trong quá trình thực hiện đồ án, không thể tránh khỏi những thiếu sót. Nhóm rất mong thầy cô thông cảm và đóng góp ý kiến để nhóm có thể hoàn thiện đồ án này trong tương lai.

1. TỔNG QUAN VỀ ĐỀ TÀI
   1. Đặt vấn đề

Với sự phát triển không ngừng của công nghệ hiện đại, các thiết bị như smartphone, laptop, máy tính bảng,… đã trở nên phổ biến và dễ tiếp cận hơn, mở ra cơ hội cho người dùng tham gia vào các trò chơi giải trí một cách thuận tiện. Mặc dù vẫn tồn tại những ý kiến trái chiều về tác động của việc chơi game, không thể phủ nhận rằng game mang lại nhiều lợi ích tích cực nếu được sử dụng đúng cách.

Các nghiên cứu được đăng tải trên Scientific Reports đã chỉ ra rằng, những người chơi game lâu năm, đặc biệt là các tựa game hành động hoặc chiến thuật, thường sở hữu khả năng quan sát nhạy bén, tư duy nhanh, và phản xạ linh hoạt. Việc chơi game đòi hỏi sự kết hợp nhịp nhàng giữa tay, mắt và khả năng xử lý tình huống, qua đó kích thích hoạt động của não bộ. Những kỹ năng này không chỉ hỗ trợ tốt trong môi trường ảo mà còn hữu ích khi áp dụng vào đời sống thực tế, đặc biệt trong các tình huống cần phản xạ nhanh.

Dự án “Thiết kế và xây dựng game đối kháng AutoCat” ra đời với mục tiêu mang đến trải nghiệm giải trí đầy thú vị và sáng tạo. Lấy cảm hứng từ những trận đấu xe chiến độc đáo, AutoCat không chỉ tập trung vào sự đơn giản và hấp dẫn của gameplay mà còn khơi dậy sự sáng tạo trong việc xây dựng chiến lược và nâng cấp xe chiến đấu. Hãy hóa thân thành những chiến binh mèo dũng mãnh, điều khiển những cỗ máy chiến đấu đầy uy lực để bảo vệ lãnh thổ. Đây là một tựa game kết hợp giữa tính chiến thuật và yếu tố hành động tự động, phù hợp với nhiều đối tượng người chơi và đảm bảo mang đến trải nghiệm thú vị.

* 1. Tổng quan về game đối kháng

Game đối kháng là một thể loại trò chơi điện tử trong đó hai hoặc nhiều người chơi tham gia vào các trận đấu với nhau trong một môi trường ảo. Mục tiêu chính của thể loại này thường là đánh bại đối thủ bằng cách sử dụng các kỹ năng chiến đấu, chiến thuật, và đôi khi là yếu tố sáng tạo trong cách chơi. Game đối kháng đã trở thành một trong những thể loại phổ biến và có sức ảnh hưởng lớn trong ngành công nghiệp game.

* + 1. Lịch sử phát triển

1. Thời kỳ đầu:

Game đối kháng xuất hiện từ những năm 1970-1980, được triển khai trên các máy chơi game console và máy tính cá nhân. Một số tựa game tiêu biểu trong giai đoạn này như:

* Karate Champ (1984): Một trong những tựa game đầu tiên định hình thể loại đối kháng với lối chơi 1vs1.
* Street Fighter (1987): Đặt nền móng cho dòng game đối kháng hiện đại, đặc biệt với cơ chế combo lần đầu tiên xuất hiện.

Những tựa game này tuy đơn giản về đồ họa và âm thanh nhưng đã thu hút được sự quan tâm lớn nhờ gameplay sáng tạo, đặt nền tảng cho sự phát triển sau này.

1. Sự phát triển của e-sports:

Vào những năm 1990, sự phát triển mạnh mẽ của các giải đấu game đã biến game đối kháng thành một phần không thể thiếu của e-sports.

* Street Fighter II (1991): Tựa game đối kháng kinh điển giúp đưa thể loại này trở thành tâm điểm trong các giải đấu.
* EVO Championship Series (2002): Một giải đấu e-sports chuyên nghiệp đầu tiên dành riêng cho game đối kháng, thu hút hàng ngàn người chơi và hàng triệu người theo dõi.

Cùng với sự phát triển của công nghệ phát sóng trực tiếp và nền tảng trực tuyến như YouTube hay Twitch, các trận đấu game đối kháng đã tiếp cận được đến đông đảo khán giả toàn cầu, giúp thể loại này ngày càng phổ biến.

1. Xu hướng hiện tại:

Đầu tư mạnh về công nghệ:

* Đồ họa 3D chân thực, hiệu ứng âm thanh sống động, và gameplay linh hoạt đã làm tăng chất lượng trải nghiệm cho người chơi.
* Một số tựa game hiện đại tiêu biểu như Tekken 7, Mortal Kombat 11, và Dragon Ball FighterZ.

Tăng cường tính kết nối:

* Chơi game đa nền tảng (cross-platform) giúp người chơi trên các thiết bị khác nhau có thể đấu với nhau.
* Các máy chủ trực tuyến và hệ thống xếp hạng toàn cầu đã tạo nên sự cạnh tranh sôi nổi.

Hướng đến cộng đồng rộng lớn:

* Các giải đấu e-sports lớn dành riêng cho game đối kháng ngày càng thu hút nhiều nhà tài trợ và người chơi chuyên nghiệp.
* Công nghệ thực tế ảo (VR) và thực tế tăng cường (AR) được tích hợp vào game, mang lại trải nghiệm mới lạ và độc đáo.
  + 1. Đặc trưng của game đối kháng

Tương tác trực tiếp: Người chơi có thể thực hiện các hành động như tấn công, phòng thủ, hoặc né tránh để đối đầu với đối thủ trong thời gian thực, tạo cảm giác căng thẳng và hấp dẫn.

Hệ thống chiến đấu đa dạng: Game đối kháng thường cung cấp nhiều loại vũ khí, kỹ năng và chiến thuật khác nhau, mang lại sự đa dạng và thử thách.

Tính cạnh tranh cao: Các trận đấu đối kháng luôn đòi hỏi kỹ năng cao từ người chơi, tạo nên động lực để cải thiện bản thân và giành chiến thắng.

Cộng đồng mạnh mẽ: Game đối kháng thường thu hút những cộng đồng người chơi lớn, nơi họ có thể giao lưu, học hỏi và thi đấu cùng nhau trong các giải đấu hoặc sự kiện.

* + 1. Các loại game đối kháng phổ biến

Game đối kháng 1vs1: Các trận đấu một đối một như Street Fighter hoặc Mortal Kombat.

Game đối kháng theo đội: Người chơi điều khiển các đội hình nhân vật, phổ biến trong các tựa game như Marvel vs. Capcom.

Game đối kháng MOBA: Kết hợp yếu tố chiến thuật và hành động, ví dụ như League of Legends hay Dota 2.

Game đối kháng Battle Royale: Các tựa game như PUBG hoặc Fortnite, nơi người chơi cạnh tranh để trở thành người sống sót cuối cùng.

* + 1. Thị trường game đối kháng

Kích thước thị trường: Thể loại này tiếp tục phát triển mạnh mẽ với doanh thu hàng tỷ USD mỗi năm, đặc biệt trong lĩnh vực eSports.

Các nhà phát hành lớn: Những công ty như Capcom, Bandai Namco, và Riot Games dẫn đầu thị trường với các tựa game kinh điển và hiện đại.

Xu hướng phát triển: Game đối kháng trong tương lai sẽ tập trung nhiều hơn vào yếu tố cộng đồng, khả năng chơi đa nền tảng, và tích hợp công nghệ tiên tiến như VR/AR.

* 1. Tổng quan về AutoCat

AutoCat là một tựa game đối kháng mới lạ, kết hợp giữa yếu tố hành động tự động và tính chiến thuật. Trong game, người chơi nhập vai vào những chú mèo chiến binh và điều khiển các cỗ máy chiến đấu được tùy chỉnh bởi chính họ. Trò chơi mang lại trải nghiệm độc đáo với ba giai đoạn chính:

* Xây dựng máy chiến đấu (Building Phase):
* Người chơi lựa chọn các bộ khung và linh kiện để lắp ráp một cỗ máy chiến đấu.
* Các bộ phận được thiết kế đa dạng, từ vũ khí, khung xe, cho đến hệ thống hỗ trợ phòng thủ, cho phép người chơi thỏa sức sáng tạo.
* Chiến đấu tự động (Combat Phase):
* Sau khi hoàn thành cỗ máy, người chơi sẽ tham gia vào các trận đấu tự động.
* Kết quả của trận đấu phụ thuộc vào chiến lược xây dựng xe và khả năng tận dụng các linh kiện đã chọn.
* Hậu chiến (Post-Combat Phase):
* Người thắng sẽ nhận được điểm và phần thưởng, giúp mở khóa các linh kiện mới để nâng cấp xe chiến đấu.

Điểm nổi bật của AutoCat:

* Gameplay sáng tạo: Sự kết hợp giữa tự động hóa và tính chiến thuật tạo nên trải nghiệm giải trí nhẹ nhàng nhưng không kém phần thú vị.
* Đồ họa và âm thanh: Thiết kế hình ảnh sinh động, đáng yêu, phù hợp với nhiều độ tuổi. Âm thanh sống động tăng cảm giác hồi hộp trong các trận đấu.
* Phù hợp nhiều đối tượng: Tựa game không chỉ hấp dẫn game thủ yêu thích hành động mà còn thu hút những người có niềm đam mê sáng tạo và chiến thuật.
  1. Nội dung đề tài

Dự án “Thiết kế và xây dựng game đối kháng AutoCat” tập trung vào các nội dung chính sau:

* Phân tích và nghiên cứu:
* Tìm hiểu các đặc điểm của game đối kháng.
* Phân tích thị trường và xu hướng phát triển của thể loại game này.
* Thiết kế hệ thống:
* Xây dựng cơ chế hoạt động của từng giai đoạn trong game (xây dựng, chiến đấu, hậu chiến).
* Thiết kế giao diện người dùng (UI) và trải nghiệm người dùng (UX).
* Phát triển game:
* Lập trình và triển khai các chức năng chính của game.
* Tích hợp đồ họa và âm thanh để nâng cao trải nghiệm.
* Kiểm thử và đánh giá:
* Tiến hành kiểm tra tính ổn định và trải nghiệm của người chơi.
* Thu thập ý kiến phản hồi để hoàn thiện game.
  1. Phạm vi đề tài
     1. Phạm vi thiết kế và phát triển

Gameplay: Xây dựng chế độ chơi PvP (Player versus Player), nơi hai người chơi được tự động ghép cặp và tham gia vào một trận đấu trực tuyến.

Chức năng cốt lõi: Tập trung vào ba tính năng chính:

* Lắp ráp chiến xa: Người chơi tự chọn và lắp ghép các bộ phận của xe chiến đấu, bao gồm khung, vũ khí và hệ thống hỗ trợ.
* Chiến đấu tự động: Các trận chiến diễn ra tự động dựa trên thiết kế chiến xa và chiến thuật của người chơi.
* Nâng cấp sau trận đấu: Người thắng nhận phần thưởng để nâng cấp xe, mở khóa linh kiện và cải thiện khả năng chiến đấu.

Đồ họa và giao diện: Thiết kế hình ảnh và giao diện đơn giản nhưng trực quan, đảm bảo tính hấp dẫn đối với nhóm người chơi phổ thông.

* + 1. Phạm vi công nghệ

Công cụ phát triển: Sử dụng Unity Engine để xây dựng toàn bộ trò chơi, từ giao diện, logic game đến kết nối mạng.

Dịch vụ kết nối: Tích hợp Unity Relay và Unity Lobby để xử lý:

* Ghép cặp tự động: Đảm bảo người chơi được kết nối nhanh chóng và mượt mà.
* Đồng bộ hóa thời gian thực: Duy trì sự đồng nhất về trạng thái giữa các người chơi trong trận đấu.

Mạng và thời gian thực: Thông tin trận đấu được đồng bộ hóa qua Unity Relay, đảm bảo độ trễ thấp và trải nghiệm liền mạch.

* + 1. Giới hạn của đề tài

Chế độ chơi: Chỉ tập trung vào chế độ PvP trực tuyến, không triển khai chế độ chơi đơn hoặc chế độ chơi theo đội.

Nội dung: Hạn chế về số lượng linh kiện và vũ khí:

* Chỉ cung cấp các linh kiện cơ bản để lắp ráp xe trong phiên bản đầu tiên.
* Các linh kiện nâng cao sẽ được phát triển và bổ sung trong các bản cập nhật sau.

Hỗ trợ nền tảng: Trò chơi chỉ được tối ưu hóa trên nền tảng PC, chưa hỗ trợ đầy đủ trên các nền tảng khác như iOS, Android, hoặc console.

* + 1. Mục tiêu phát triển

Hoàn thiện sản phẩm: Đảm bảo game hoạt động mượt mà, chức năng đầy đủ và mang lại trải nghiệm chơi hấp dẫn cho người dùng.

Nền tảng mở rộng: Xây dựng nền tảng cơ bản có khả năng mở rộng, phục vụ việc bổ sung tính năng hoặc nội dung mới trong tương lai.

Trải nghiệm người chơi: Tạo ra một sản phẩm giải trí sáng tạo, dễ tiếp cận nhưng vẫn đảm bảo tính cạnh tranh và tính chiến thuật cao.

1. KHẢO SÁT
   1. Mục tiêu khảo sát

Đánh giá sự quan tâm của người chơi: Xác định mức độ yêu thích của cộng đồng game thủ Việt Nam đối với thể loại game đối kháng nói chung và ý tưởng game AutoCat nói riêng.

Xác định yếu tố mong đợi: Tìm hiểu các tính năng và yếu tố gameplay mà người chơi mong đợi trong một tựa game đối kháng có chủ đề về xe chiến đấu tự động.

So sánh với đối thủ cạnh tranh: Đánh giá AutoCat so với các tựa game đối kháng nổi bật khác trên thị trường để nhận diện điểm mạnh và hạn chế của ý tưởng.

Phân tích điểm mạnh và điểm yếu: Xác định những ưu điểm cần phát huy và các vấn đề cần cải thiện trong ý tưởng phát triển AutoCat, đặc biệt là về lối chơi, thiết kế, và khả năng kết nối trực tuyến.

Thu thập ý kiến người chơi: Lắng nghe những phản hồi, đề xuất, và mong muốn từ người chơi để điều chỉnh sản phẩm phù hợp hơn với nhu cầu thực tế.

Phân tích đối tượng khảo sát: Xác định các đặc điểm của đối tượng khảo sát, bao gồm:

* Loại hình người chơi: game thủ chuyên nghiệp, nghiệp dư hoặc người chơi giải trí.
* Nhóm độ tuổi và giới tính.
* Sở thích chơi game (game đối kháng, chiến thuật, hành động,...).
  1. Khảo sát các game đối kháng tương tự
     1. Mục tiêu khảo sát:

Xác định đối thủ cạnh tranh trực tiếp:

* Tìm ra những tựa game đối kháng cùng thể loại (xe chiến đấu, nhân vật tự động) hoặc có lối chơi tương tự.
* Xác định các trò chơi hướng đến cùng đối tượng mục tiêu như AutoCat (game thủ trẻ, yêu thích sáng tạo và chiến thuật).

Phân tích điểm mạnh và điểm yếu:

* Đánh giá các yếu tố như đồ họa, âm thanh, gameplay, và khả năng kết nối cộng đồng của các trò chơi.
* Tìm ra các điểm yếu của đối thủ mà AutoCat có thể cải thiện hoặc tận dụng.

Xác định xu hướng thị trường:

* Ghi nhận các công nghệ hoặc phong cách thiết kế mới nhất mà người chơi ưa chuộng.
* Tìm hiểu cách các trò chơi đang duy trì và mở rộng cộng đồng.
  + 1. Phương pháp khảo sát

Bước 1: Lập danh sách các game đối thủ:

* Robot Wars Arena: Tập trung vào thiết kế và chiến đấu xe tự động.



1. Giao diện trận đấu game Robot Wars Arena

Điểm mạnh:

* Hệ thống lắp ráp xe phong phú với nhiều tùy chỉnh.
* Gameplay đậm tính chiến thuật, cho phép người chơi sáng tạo và thử nghiệm nhiều chiến lược.
* Đồ họa chi tiết, mang phong cách thực tế với hiệu ứng chiến đấu ấn tượng.
* Cộng đồng game thủ tích cực nhờ các giải đấu và nội dung được cập nhật thường xuyên.

Điểm yếu:

* Độ phức tạp cao có thể khiến người chơi mới gặp khó khăn khi bắt đầu.
* Yêu cầu cấu hình máy tính mạnh để trải nghiệm mượt mà.
* Crossout: Game chiến thuật với xe chiến tự lắp ráp.



1. Giao diện game Crossout



1. Giao diện trận đấu trong game CrossOut

Điểm mạnh:

* Kết hợp yếu tố bắn súng và chiến thuật, mang đến trải nghiệm đa dạng.
* Hệ thống lắp ráp xe phong phú, từ xe tăng đến các phương tiện kỳ lạ.
* Hệ thống PvP cạnh tranh cao, phù hợp với người chơi hardcore.
* Nền kinh tế trong game được xây dựng tốt, khuyến khích giao dịch giữa người chơi.

Điểm yếu:

* Cần thời gian dài để thu thập tài nguyên và nâng cấp xe.
* Người chơi mới thường bị bất lợi khi đối đầu với game thủ kỳ cựu.

Bước 2: So sánh và phân tích

|  |  |  |
| --- | --- | --- |
| Tính năng | Robot Wars Arena | Crossout |
| Gameplay | Chiến thuật, tập trung vào việc xây dựng và tùy chỉnh xe | Hành động nhanh, bắn phá, tập trung vào PvP |
| Cơ chế chiến đấu | Vũ khí đa dạng, hệ thống module, tập trung vào chiến thuật | Vũ khí đa dạng, hệ thống module, tập trung vào PvP và PvE |
| Môi trường chiến đấu | Bản đồ mở, nhiều địa hình khác nhau | Bản đồ đa dạng, có các khu vực đặc biệt |
| Đồ họa | Chi tiết, phong cách thực tế | Sắc nét, sáng tạo với nhiều hiệu ứng độc đáo |
| Cộng đồng | Cộng đồng lớn, nhiều hoạt động, giải đấu | Cộng đồng lớn, hoạt động sôi nổi, nền kinh tế trong game phát triển |
| Tiềm năng phát triển | Tiềm năng mở rộng nội dung, cập nhật thường xuyên | Tiềm năng phát triển eSports |

Bước 3: Ứng dụng vào AutoCat:

* Điểm cần học hỏi:

Phát triển hệ thống lắp ráp xe linh hoạt nhưng thân thiện với người chơi mới.

Đảm bảo gameplay kết hợp giữa chiến thuật và hành động tự động để thu hút cả người chơi casual và hardcore.

Tăng tính cạnh tranh thông qua các giải đấu PvP hoặc sự kiện cộng đồng.

* Điểm cần cải thiện:

Giảm bớt độ phức tạp ban đầu bằng cách tích hợp hướng dẫn hoặc chế độ chơi dành cho người mới.

Tối ưu hóa đồ họa để hỗ trợ cấu hình máy tính trung bình mà vẫn giữ được tính hấp dẫn.

* Sự khác biệt hóa:

Sử dụng phong cách đồ họa dễ thương với nhân vật chính là mèo, mang nét độc đáo và thu hút nhóm game thủ trẻ.

Tích hợp tính năng xã hội như bảng xếp hạng bạn bè, mời bạn bè thi đấu để tăng mức độ gắn kết người chơi.

Phân tích SWOT cho AutoCat :

Phân tích SWOT là một công cụ hữu ích để đánh giá một dự án, một sản phẩm, một công ty hoặc một ý tưởng. Nó giúp ta có cái nhìn tổng quan về tình hình hiện tại, xác định những yếu tố nội tại và ngoại tại tác động đến sự thành công của dự án.

|  |  |  |  |
| --- | --- | --- | --- |
| S (Strengths) | S (Strengths) | O (Opportunities) | T (Threats) |
| Chủ đề độc đáo, phong cách dễ thương | Có thể bị đánh giá là game dành cho trẻ em | Thị trường game mobile đang phát triển mạnh | Cạnh tranh khốc liệt từ các game đối thủ |
| Hệ thống tùy biến xe linh hoạt | Độ phức tạp của hệ thống có thể làm khó người chơi mới | Xu hướng eSports đang phát triển | Thay đổi sở thích của người chơi |
| Cộng đồng gắn kết | Cần thời gian để xây dựng cộng đồng | Công nghệ mới (VR, AR) có thể thay đổi cách người chơi trải nghiệm game |  |

* 1. Khảo sát nhu cầu người dùng
     1. Mục tiêu khảo sát

Hiểu rõ người chơi: Nắm bắt sở thích, mong đợi và hành vi của người chơi đối với thể loại game đối kháng và ý tưởng AutoCat.

Đánh giá mức độ hấp dẫn của ý tưởng: Kiểm tra xem ý tưởng về AutoCat có đáp ứng được nhu cầu và mong đợi của người chơi không.

Thu thập ý kiến đóng góp: Lắng nghe phản hồi, đề xuất từ người chơi để cải tiến và hoàn thiện trò chơi trước khi ra mắt.

Xác định phân khúc khách hàng: Phân tích dữ liệu về độ tuổi, giới tính, thói quen chơi game để xác định đối tượng mục tiêu cụ thể.

* + 1. Phương pháp khảo sát

Bảng hỏi trực tuyến:

* Công cụ: Google Forms
* Câu hỏi đa dạng:

Câu hỏi đóng (Yes/No, thang điểm Likert).

Câu hỏi mở (Ý kiến đóng góp, nhận xét).

Phỏng vấn:

* Cách thực hiện: Phỏng vấn trực tiếp hoặc qua điện thoại với một nhóm game thủ đại diện.
* Đối tượng: Người chơi thuộc nhiều phân khúc (casual, hardcore).
  + 1. Nội dung khảo sát

Câu hỏi phỏng vấn:

1. Bạn thường chơi game trên nền tảng nào? (PC, mobile, console)
2. Thể loại game mà bạn yêu thích nhất là gì? (Đối kháng, chiến thuật, hành động, nhập vai, phiêu lưu,...)
3. Bạn thường chọn game dựa trên yếu tố nào? (Đồ họa, gameplay, thương hiệu, đánh giá cộng đồng,...)
4. Bạn cảm thấy thế nào về ý tưởng một game đối kháng xoay quanh xe chiến với nhân vật là mèo? (Thang điểm từ 1 đến 5)
5. Điều gì sẽ khiến bạn chơi thử AutoCat? (Gameplay độc đáo, nhân vật thú vị, miễn phí chơi thử, khuyến mãi,...)
6. Bạn nghĩ các yếu tố nào làm nên một game đối kháng hấp dẫn? (Gameplay, đồ họa, âm thanh, cộng đồng, nội dung thường xuyên được cập nhật,...)
7. Có điều gì bạn không thích ở các tựa game đối kháng hiện tại? (Độ khó, giao diện, thiếu tính năng, mất cân bằng,...)
8. Bạn có ý tưởng hoặc đề xuất nào để cải thiện AutoCat? (Câu hỏi mở)
9. Nếu có một sự kiện thử nghiệm beta cho AutoCat, bạn có muốn tham gia không? (Có, không)

Bảng câu hỏi trực tuyến:

1. Bạn thường chơi game trên nền tảng nào? (Chọn tất cả những gì phù hợp)

* PC
* Mobile
* Console
* Khác (Vui lòng ghi rõ): \_\_\_\_\_\_\_\_\_\_\_

1. Thể loại game mà bạn yêu thích nhất là gì? (Chọn một)

* Đối kháng
* Chiến thuật
* Hành động
* Phiêu lưu
* Khác (Vui lòng ghi rõ): \_\_\_\_\_\_\_\_\_\_\_

1. Bạn cảm thấy thế nào về ý tưởng một game đối kháng xoay quanh xe chiến với nhân vật là mèo? (Chọn một)

* 1 - Không hấp dẫn
* 2 - Ít hấp dẫn
* 3 - Bình thường
* 4 - Hấp dẫn
* 5 - Rất hấp dẫn

1. Những tính năng nào bạn mong muốn có trong AutoCat? (Chọn tất cả những gì phù hợp)

* Tùy chỉnh xe
* Đấu hạng PvP
* Sự kiện đặc biệt
* Cộng đồng người chơi sôi động
* Khác (Vui lòng ghi rõ): \_\_\_\_\_\_\_\_\_\_\_

1. Bạn thích kiểu đồ họa nào hơn cho AutoCat? (Chọn một)

* Phong cách cartoon
* Thực tế
* Tối giản
* Pixel art
* Khác (Vui lòng ghi rõ): \_\_\_\_\_\_\_\_\_\_\_

1. Theo bạn, yếu tố nào làm nên một game đối kháng hấp dẫn? (Chọn tối đa 3 mục)

* Gameplay độc đáo
* Đồ họa đẹp mắt
* Âm thanh sống động
* Cộng đồng mạnh mẽ
* Nội dung thường xuyên được cập nhật
* Khác (Vui lòng ghi rõ): \_\_\_\_\_\_\_\_\_\_\_

1. Có điều gì bạn không thích ở các tựa game đối kháng hiện tại? (Câu hỏi mở)
2. Bạn có muốn tham gia sự kiện thử nghiệm beta cho AutoCat không? (Chọn một)

* Có
* Không
  + 1. Phân tích dữ liệu
* Phân tích định lượng:

Nền tảng chơi game phổ biến:

PC: 45%

Mobile: 35%

Console: 15%

Khác: 5%

Kết luận: PC và mobile là hai nền tảng phổ biến nhất, cho thấy AutoCat nên tập trung vào việc tối ưu hóa cả hai nền tảng này.

Thể loại game yêu thích:

Đối kháng: 40%

Chiến thuật: 30%

Hành động: 20%

Phiêu lưu: 5%

Khác: 5%

Kết luận: Game đối kháng chiếm ưu thế, cho thấy AutoCat cần giữ thể loại đối kháng làm trọng tâm.

Mức độ hấp dẫn với ý tưởng AutoCat (thể loại xe chiến đấu, mèo):

1 - Không hấp dẫn: 5%

2 - Ít hấp dẫn: 10%

3 - Bình thường: 25%

4 - Hấp dẫn: 35%

5 - Rất hấp dẫn: 25%

Kết luận: Hầu hết người chơi có cái nhìn tích cực về ý tưởng game, với hơn 60% đánh giá ở mức "hấp dẫn" hoặc "rất hấp dẫn". Điều này là tín hiệu tốt cho sự sáng tạo của AutoCat.

Các tính năng mong muốn trong AutoCat:

Tùy chỉnh xe: 50%

Đấu hạng PvP: 40%

Sự kiện đặc biệt: 25%

Cộng đồng người chơi sôi động: 30%

Khác (sáng tạo nhân vật, chế độ chơi đặc biệt): 5%

Kết luận: Người chơi rất chú trọng vào tính năng tùy chỉnh xe, theo sau là đấu hạng PvP và các sự kiện đặc biệt. Tập trung vào phát triển các tính năng này sẽ giúp AutoCat thu hút người chơi.

Phong cách đồ họa yêu thích:

Phong cách cartoon: 50%

Thực tế: 25%

Tối giản: 15%

Pixel art: 10%

Kết luận: Phong cách đồ họa cartoon được ưa chuộng nhất, cho thấy AutoCat nên lựa chọn phong cách đồ họa vui nhộn, dễ tiếp cận và hấp dẫn.

Yếu tố quan trọng để một game đối kháng hấp dẫn:

Gameplay độc đáo: 60%

Đồ họa đẹp mắt: 50%

Âm thanh sống động: 30%

Cộng đồng mạnh mẽ: 40%

Nội dung thường xuyên được cập nhật: 45%

Kết luận: Gameplay độc đáo và đồ họa đẹp mắt là hai yếu tố quan trọng nhất. AutoCat cần đặc biệt chú trọng vào việc tạo ra trải nghiệm gameplay thú vị và đồ họa nổi bật.

* Phân tích định tính:

1. Điều không thích ở các game đối kháng hiện tại:

Độ khó không phù hợp (quá cao hoặc quá thấp): 35%

Thiếu tính năng đa dạng: 25%

Mất cân bằng trong gameplay (game thủ cũ dễ dàng vượt trội): 20%

Giao diện và điều khiển không thân thiện: 15%

Khả năng kết nối cộng đồng yếu: 5%

Kết luận: Người chơi đang tìm kiếm sự cân bằng và tính dễ tiếp cận trong các game đối kháng. AutoCat cần chú ý đến việc tạo ra một hệ thống gameplay cân bằng và dễ tiếp cận cho mọi đối tượng người chơi.

* Ý tưởng hoặc đề xuất để cải thiện AutoCat:

"Thêm nhiều chế độ chơi mới và các sự kiện đặc biệt để không bị nhàm chán."

"Cải thiện hệ thống tìm trận đấu PvP để không gặp phải đối thủ quá mạnh."

"Tạo thêm tính năng hợp tác với bạn bè trong PvE."

Kết luận: Người chơi mong muốn có thêm chế độ chơi mới, sự kiện đặc biệt, và cải tiến PvP để nâng cao trải nghiệm.

* Kết luận tổng quan:

Đồ họa cartoon và gameplay độc đáo là hai yếu tố quan trọng nhất đối với người chơi.

Các tính năng như tùy chỉnh xe, đấu hạng PvP, và sự kiện đặc biệt cần được ưu tiên phát triển.

Cải thiện PvP, cân bằng độ khó và tăng cường cộng đồng là những cải tiến quan trọng để làm hài lòng người chơi.

Phân tích này sẽ giúp đội ngũ phát triển AutoCat tập trung vào các yếu tố quan trọng nhất và hướng đi chiến lược phù hợp với thị trường mục tiêu.

1. PHÂN TÍCH VÀ THIẾT KẾ
   1. Phân tích đề tài
      1. Giới thiệu game AutoCat

AutoCat là một trò chơi đối kháng độc đáo, kết hợp giữa thể loại game hành động và sáng tạo, nơi người chơi điều khiển những nhân vật mèo thông qua các chiến xa tự chế để tham gia vào các trận đấu quyết liệt. Trò chơi không chỉ tập trung vào việc chiến đấu mà còn tạo điều kiện cho người chơi sáng tạo và tùy chỉnh chiến xa của mình. Mỗi người chơi sẽ có thể thiết kế, lắp ráp và lựa chọn các linh kiện cho chiến xa trước khi bước vào trận đấu. Bằng cách kết hợp giữa sáng tạo và chiến thuật, AutoCat mang lại một trải nghiệm hoàn toàn mới cho những người yêu thích thể loại game đối kháng.

* + 1. Kịch bản game

AutoCat diễn ra trong một thế giới giả tưởng nơi mèo và các sinh vật khác chiến đấu để giành quyền kiểm soát những khu vực lãnh thổ quan trọng. Trong thế giới này, các đội mèo được trang bị chiến xa tự chế để tham gia vào các trận đấu, bảo vệ và mở rộng vùng lãnh thổ của mình. Người chơi sẽ vào vai một chiến binh mèo, tham gia vào các trận chiến PvP (Player vs Player), đấu với những đối thủ khác trong môi trường chiến đấu đầy thử thách.

Kịch bản của trò chơi được chia thành các pha, bao gồm:

* Giai đoạn xây dựng: Người chơi sẽ thiết kế và lắp ráp chiến xa từ các linh kiện có sẵn.
* Giai đoạn chiến đấu: Các chiến xa sẽ tự động chiến đấu với nhau, nhưng người chơi có thể can thiệp vào chiến thuật và điều chỉnh chiến xa khi cần thiết.
* Hậu chiến đấu: Sau mỗi trận đấu, người chơi sẽ nhận thưởng và có thể mở khóa thêm các linh kiện mới để nâng cấp hoặc thay đổi chiến xa của mình.
  + 1. Đặc tả yêu cầu

Yêu cầu về gameplay

* Chế độ chơi: AutoCat sẽ có các chế độ chơi chính gồm:

PvP trực tuyến: Người chơi sẽ được ghép cặp đấu trực tiếp với đối thủ qua hệ thống server.

* Cơ chế xây dựng chiến xa: Trò chơi yêu cầu một hệ thống xây dựng chiến xa mạnh mẽ, nơi người chơi có thể lựa chọn linh kiện như khung xe, động cơ, vũ khí, và các phụ kiện để tạo ra những chiến xa độc đáo. Mỗi linh kiện có chỉ số riêng biệt ảnh hưởng đến tốc độ, khả năng tấn công, phòng thủ và sự linh hoạt của chiến xa.
* Chiến đấu tự động: Chiến xa sẽ tự động di chuyển và tấn công đối thủ theo chiến thuật đã được người chơi thiết lập. Người chơi có thể điều chỉnh chiến thuật hoặc can thiệp trực tiếp trong trận đấu bằng cách thay đổi các chế độ tấn công, phòng thủ của chiến xa.

Yêu cầu về đồ họa và âm thanh

Đồ họa: Phong cách đồ họa sẽ mang tính hoạt hình (cartoon), dễ tiếp cận và phù hợp với đối tượng người chơi trẻ tuổi. Các chiến xa và nhân vật sẽ có thiết kế dễ thương nhưng đầy đủ chi tiết và sắc nét.

Hiệu ứng âm thanh: Âm thanh sẽ bao gồm nhạc nền sôi động và các hiệu ứng âm thanh từ chiến xa, vũ khí, va chạm trong trận đấu. Âm thanh đóng vai trò quan trọng trong việc tạo ra sự căng thẳng và kịch tính cho mỗi trận chiến.

Yêu cầu về giao diện người dùng (UI)

Giao diện xây dựng chiến xa: Cần có giao diện đơn giản nhưng chi tiết để người chơi dễ dàng lựa chọn và thay đổi linh kiện chiến xa. Hệ thống kéo thả và các menu dễ hiểu sẽ giúp người chơi tùy chỉnh xe nhanh chóng.

Giao diện trận đấu: Trong quá trình trận đấu, giao diện phải cho phép người chơi theo dõi các chỉ số của chiến xa và tình hình trận đấu. Các tính năng như zoom in/out và chế độ xem lại chiến thuật sẽ giúp người chơi dễ dàng theo dõi trận đấu.

Yêu cầu về kết nối và cộng đồng

Kết nối trực tuyến: Trò chơi yêu cầu một kết nối internet ổn định để tham gia vào các trận đấu PvP trực tuyến. Các trận đấu cần được ghép cặp một cách công bằng, dựa trên trình độ và điểm số của người chơi.

Cộng đồng người chơi: Trò chơi sẽ hỗ trợ các tính năng xã hội như chat nhóm, phòng chơi, và giải đấu để người chơi có thể kết nối và thi đấu với nhau.

Yêu cầu về tính năng mở rộng

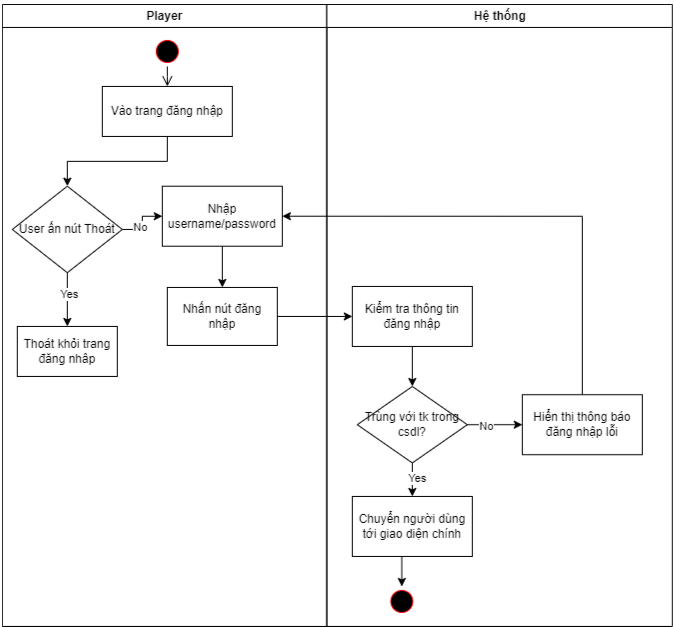
Mở khóa linh kiện: Người chơi có thể mở khóa thêm linh kiện qua các trận đấu thắng, sự kiện đặc biệt hoặc thông qua các giao dịch trong game.

Cập nhật định kỳ: Trò chơi cần có kế hoạch cập nhật thường xuyên các sự kiện, tính năng mới, và các linh kiện để giữ chân người chơi lâu dài.

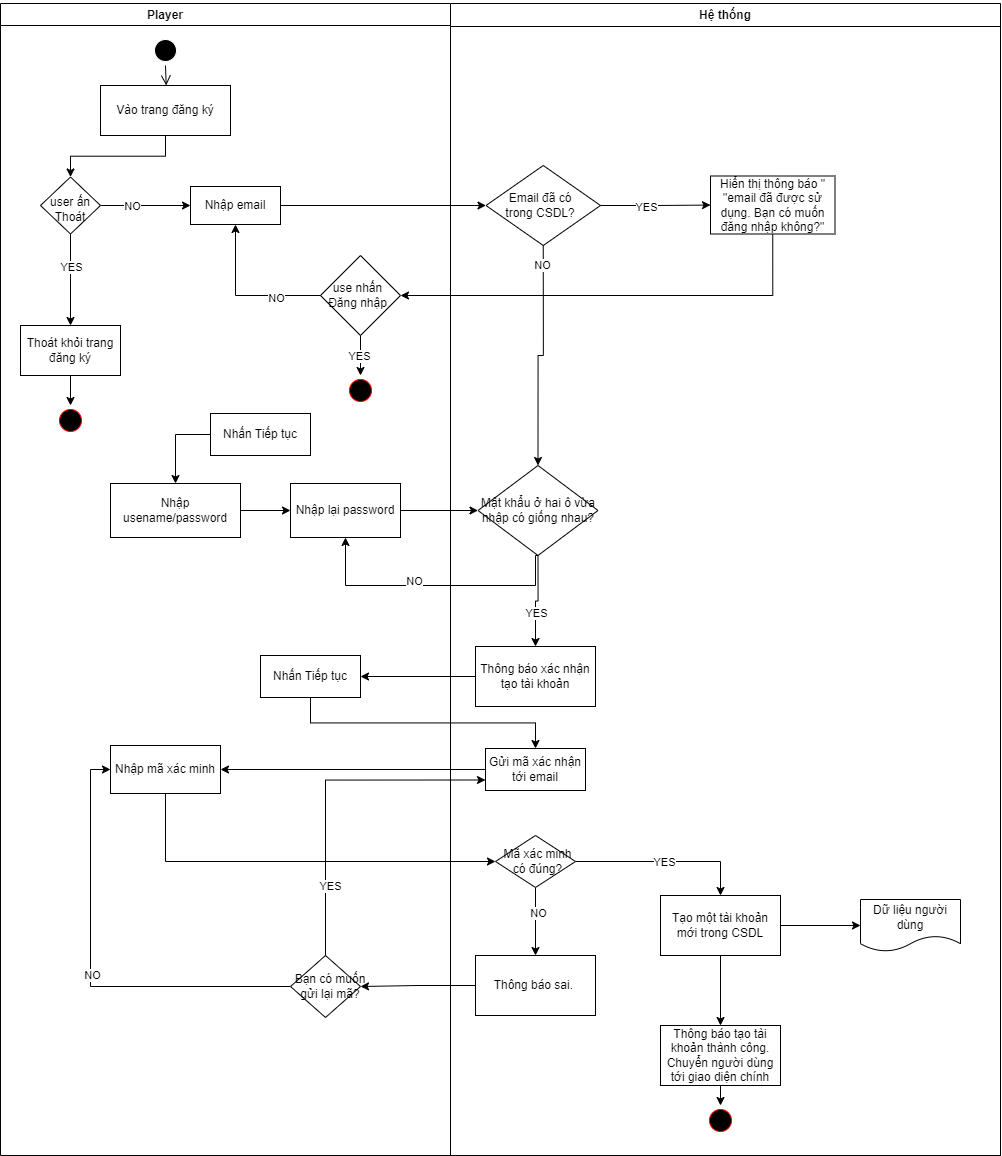
* 1. Biểu đồ hành động
     1. Biểu đồ hành động đăng nhập và đăng ký

Mục tiêu: Cho phép người chơi tạo tài khoản mới hoặc đăng nhập vào tài khoản đã có.

Biểu đồ chức năng:



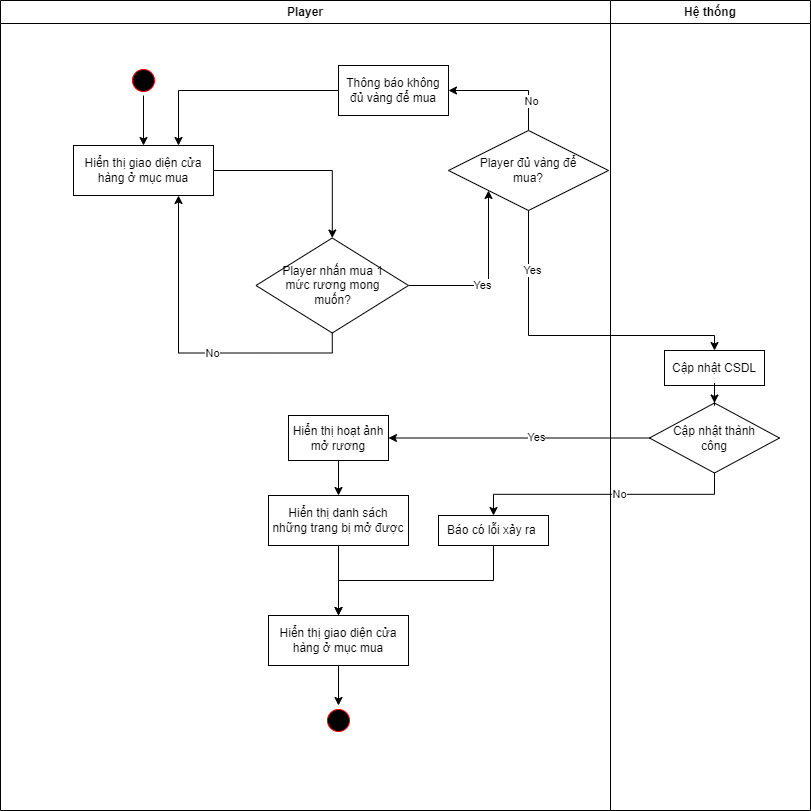
1. Biểu đồ hành động đăng nhập



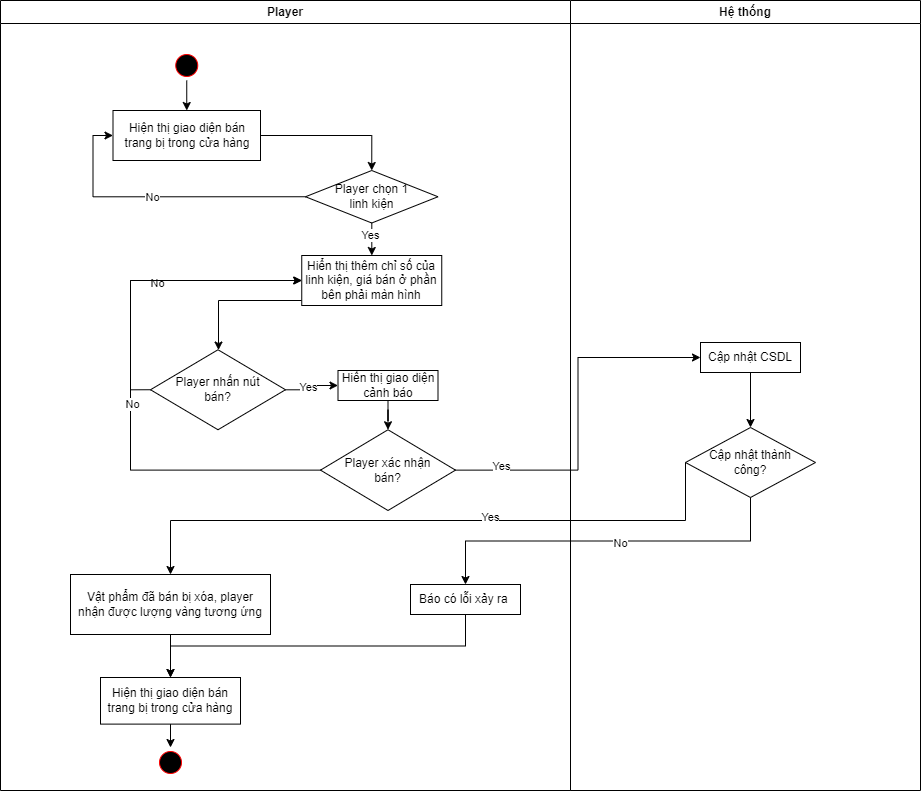
1. Biểu đồ hành động đăng ký
   * 1. Biểu đồ hành động mua và bán trang bị

Mục tiêu: Cho phép người chơi mua và bán các trang bị để nâng cấp chiến xa.

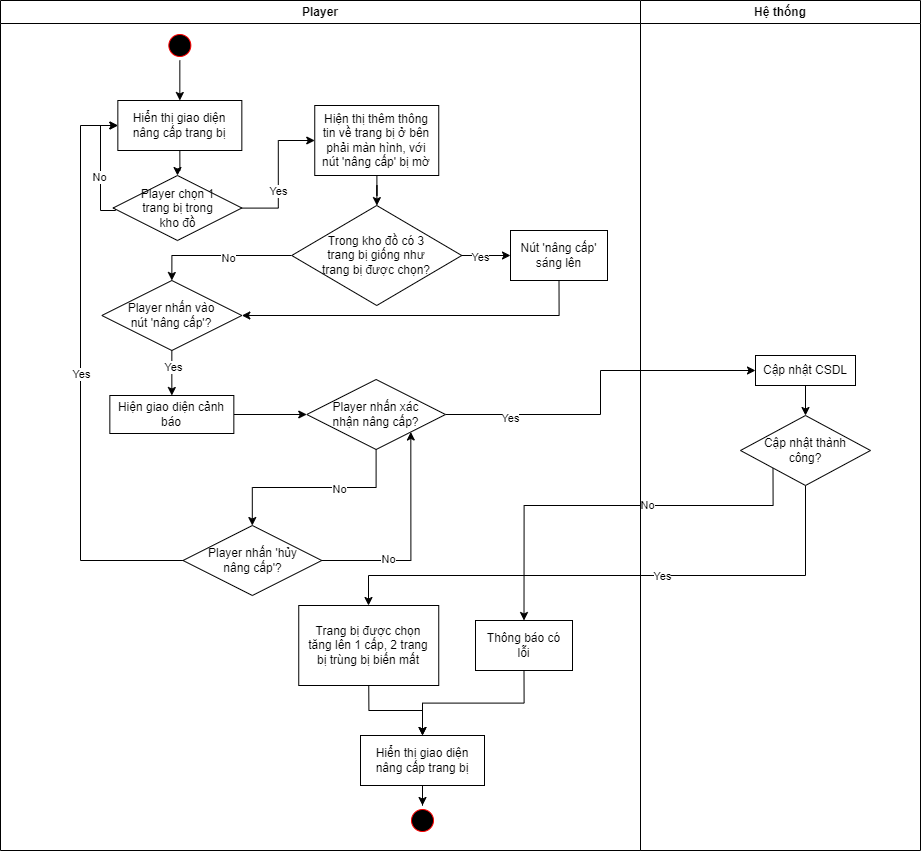
Biểu đồ chức năng:



1. Biểu đồ hành động mua trang bị



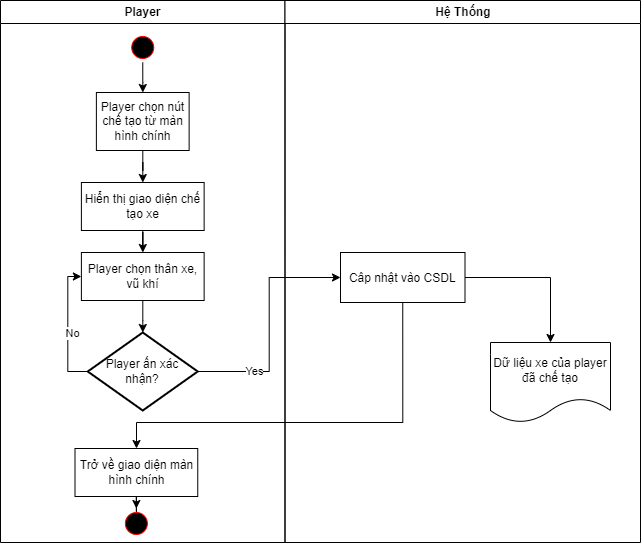
1. Biểu đồ hành động bán trang bị
   * 1. Biểu đồ hành động nâng cấp trang bị



1. Biểu đồ hành động nâng cấp trang bị
   * 1. Biểu đồ hành động chế tạo xe chiến đấu

Mục tiêu: Cho phép người chơi tự thiết kế và lắp ráp chiến xa.

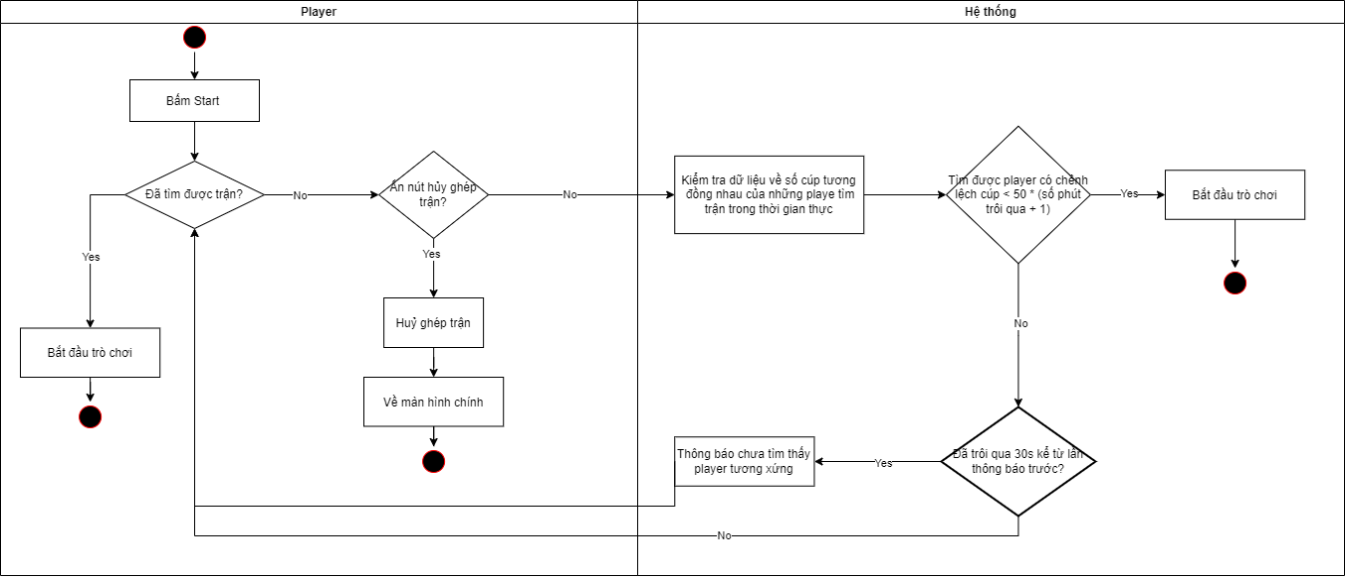
Biểu đồ chức năng:



1. Biểu đồ hành động chế tạo xe chiến đấu
   * 1. Biểu đồ hành động ghép trận

Mục tiêu: Tìm kiếm một đối thủ phù hợp để bắt đầu trận đấu.

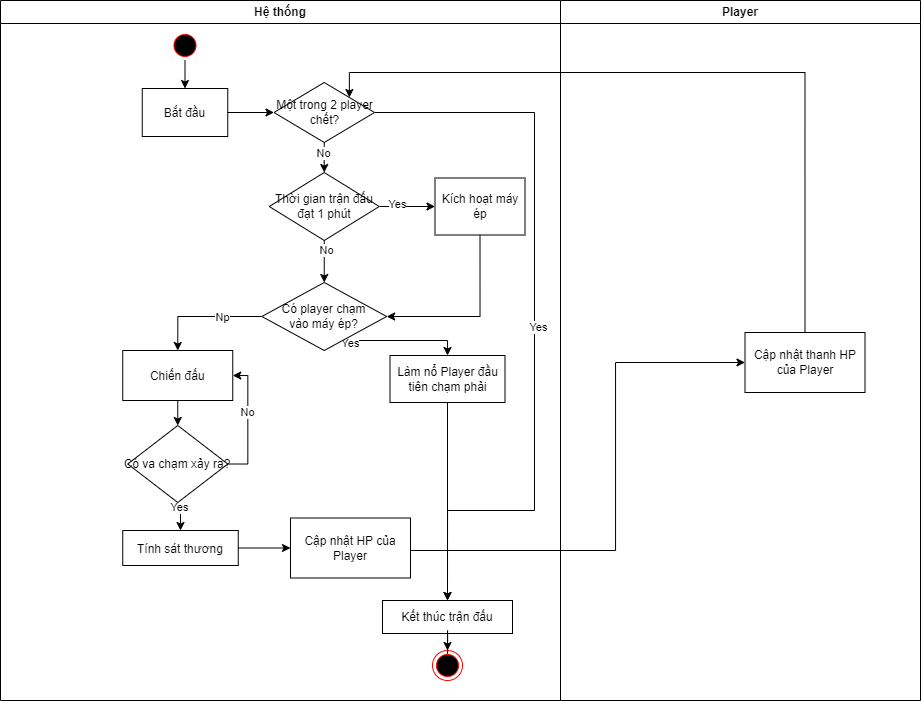
Biểu đồ chức năng:



1. Biểu đồ hành động ghép trận
   * 1. Biểu đồ hành động chiến đấu

Mục tiêu: Hai người chơi điều khiển chiến xa của mình để tiêu diệt đối thủ.

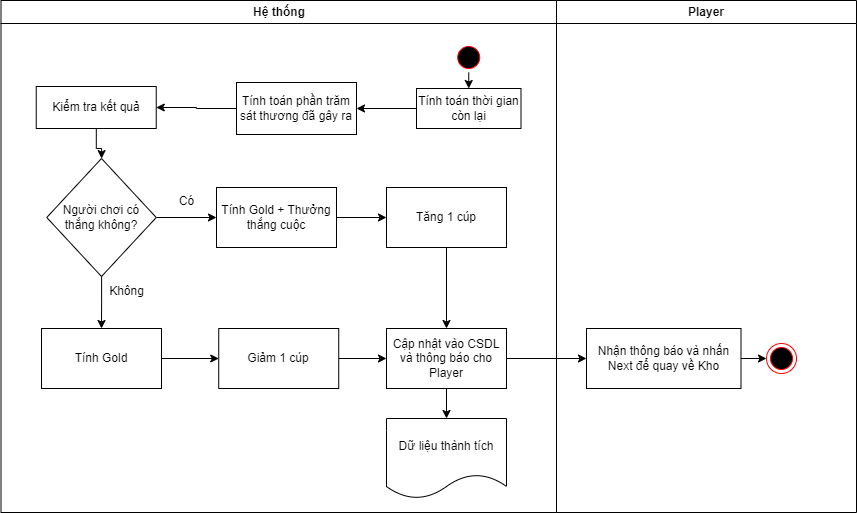
Biểu đồ chức năng:



1. Biểu đồ hành động chiến đấu
   * 1. Biểu đồ hành động Sau chiến đấu

Mục tiêu: Cập nhật thông tin người chơi và cấp thưởng.

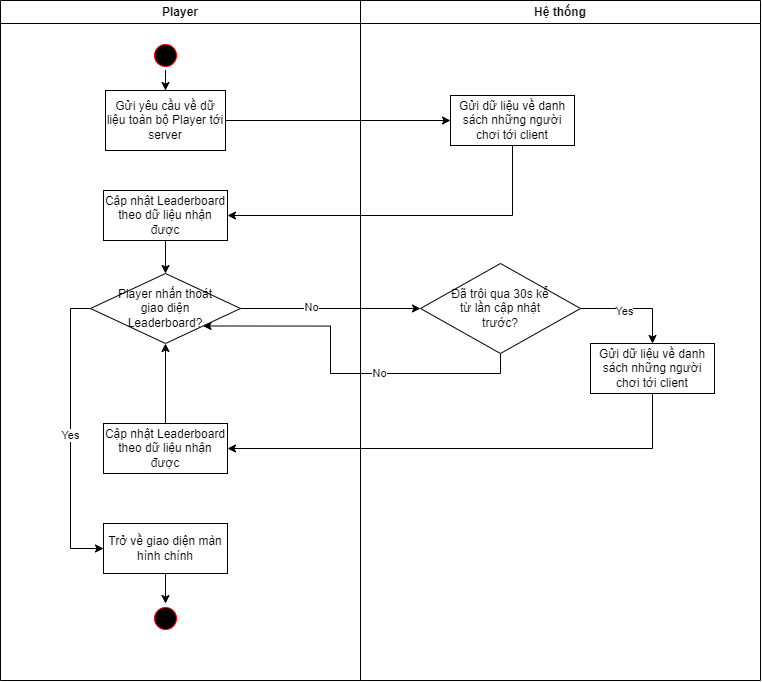
Biểu đồ chức năng:



1. Biểu đồ hành động sau trận đấu
   * 1. Biểu đồ hành động leaderboard

Mục tiêu: Hiển thị bảng xếp hạng của người chơi.

Biểu đồ chức năng:



1. Biểu đồ hành động leaderboard
   1. Thiết kế kiến trúc hệ thống
      1. Kiến trúc tổng thể

AutoCat sử dụng mô hình Host – Client để kết nối giữa máy chủ và các thiết bị của người chơi. Trong mô hình này, máy chủ (Host) sẽ chịu trách nhiệm xử lý các logic quan trọng của trò chơi như đồng bộ dữ liệu và quản lý các trận đấu, trong khi đó các thiết bị của người chơi (Client) sẽ chỉ xử lý các hoạt động như gửi thông tin đầu vào và hiển thị đồ họa.

Hệ thống sẽ kết nối thông qua Unity Service để đảm bảo việc xử lý mạng và giao tiếp giữa các thiết bị người chơi, cũng như sử dụng Unity Cloud để lưu trữ và đồng bộ hóa dữ liệu giữa các client và host một cách hiệu quả.

* + 1. Các thành phần trong hệ thống

Thành phần phía Host

* Xử lý logic trận đấu: Máy chủ sẽ điều khiển toàn bộ quá trình của trận đấu, bao gồm các sự kiện xảy ra trong quá trình chiến đấu, tính toán các chỉ số, kết quả trận đấu, và việc quản lý chiến xa của người chơi.
* Đồng bộ hóa dữ liệu, xử lý thông tin đầu vào: Các thông tin từ client (như thao tác của người chơi, hành động trong trận đấu) sẽ được gửi tới máy chủ để xử lý và đồng bộ hóa với các client còn lại. Máy chủ đảm bảo rằng mỗi client nhận được thông tin cập nhật chính xác về trạng thái của trận đấu.
* Quản lý người chơi (các clients): Máy chủ có nhiệm vụ quản lý số lượng người chơi hiện có, xác định người chơi nào đang tham gia vào trận đấu nào, và duy trì kết nối liên tục với tất cả các client trong suốt quá trình chơi.
* Lưu trữ tạm thời (Temporary Storage): Máy chủ sẽ lưu trữ thông tin tạm thời của các trận đấu như trạng thái trận đấu, chỉ số của các chiến xa và thông tin người chơi trong suốt trận đấu. Sau khi trận đấu kết thúc, thông tin này sẽ được ghi vào cơ sở dữ liệu chính thức (nếu có) hoặc xóa bỏ.

Thành phần phía Client

* Gửi thông tin đầu vào: Mỗi client sẽ gửi thông tin đầu vào của người chơi (như di chuyển chiến xa, thay đổi chiến thuật, điều chỉnh các vũ khí) tới máy chủ để xử lý và cập nhật trạng thái trận đấu.
* Hiển thị đồ họa của game: Client sẽ chịu trách nhiệm hiển thị đồ họa, các hiệu ứng chiến đấu, môi trường trong game, và tất cả các thành phần giao diện người dùng (UI). Nó sẽ nhận các thông tin từ máy chủ và cập nhật lại màn hình của người chơi tương ứng với trạng thái hiện tại của trận đấu.
  + 1. Mô hình dữ liệu

Mô hình dữ liệu trong AutoCat bao gồm các thành phần sau:

* Dữ liệu người chơi: Các thông tin của người chơi như tài khoản, điểm số, linh kiện đã mở khóa, chiến xa đã tạo, lịch sử trận đấu.
* Dữ liệu trận đấu: Bao gồm các thông tin về trạng thái trận đấu hiện tại như: thông tin về các chiến xa, trạng thái sức khỏe, điểm số, thời gian còn lại trong trận đấu, v.v.
* Dữ liệu linh kiện: Các thông tin chi tiết về các linh kiện có sẵn trong game, bao gồm các thông số kỹ thuật, tên, loại linh kiện (vũ khí, khung xe, động cơ...), trạng thái mở khóa.

Các dữ liệu này sẽ được lưu trữ trong một cơ sở dữ liệu trung tâm (có thể là trên Unity Cloud) và đồng bộ giữa các client và host trong suốt quá trình chơi.

* + 1. Lưu đồ hoạt động của hệ thống Host-Client

Bước 1: Khởi tạo trận đấu

* Client A và Client B kết nối với máy chủ (Host) qua Unity Service.
* Máy chủ kiểm tra trạng thái kết nối và khởi tạo trận đấu, bao gồm việc ghép cặp người chơi và đồng bộ dữ liệu trận đấu.

Bước 2: Xử lý đầu vào

* Các client gửi thông tin đầu vào của người chơi (như thay đổi chiến thuật, di chuyển) tới máy chủ.
* Máy chủ nhận thông tin từ các client, xử lý và cập nhật trạng thái trận đấu.

Bước 3: Đồng bộ hóa trạng thái

* Máy chủ gửi lại thông tin về trạng thái hiện tại của trận đấu (ví dụ: vị trí chiến xa, chỉ số sức khỏe) cho tất cả các client.
* Client nhận dữ liệu từ máy chủ và cập nhật màn hình hiển thị, đảm bảo người chơi thấy đúng trạng thái của trận đấu.

Bước 4: Kết thúc trận đấu

* Sau khi trận đấu kết thúc (một trong hai chiến xa bị phá hủy), máy chủ sẽ tính toán kết quả và thông báo cho các client.
* Dữ liệu trận đấu (kết quả, điểm số) sẽ được lưu trữ trong hệ thống của máy chủ và có thể được đồng bộ hóa với cơ sở dữ liệu.

Bước 5: Kết thúc và chuẩn bị cho trận đấu tiếp theo

* Các client nhận thông báo kết quả và chuẩn bị cho trận đấu tiếp theo (hoặc thoát ra ngoài).
* Máy chủ đồng bộ thông tin và chuẩn bị cho các trận đấu mới.
  1. Thiết kế cơ sở dữ liệu
  2. Thiết kế giao diện người dùng (UI)
     1. Sơ đồ cấu trúc (Site Map)
     2. Wireframe các màn hình chính

1. THIẾT KẾ GAO DIỆN GAME
   1. Giao diện mở đầu
   2. Giao diện chiến đấu
   3. Giao diện mở rương
2. XÂY DỰNG CHƯƠNG TRÌNH
   1. Công nghệ sử dụng

Unity là một công cụ phát triển game phổ biến cho phép tạo ra các trò chơi 2D và 3D với khả năng hỗ trợ đa nền tảng như PC, console, mobile, và web. Unity có các công cụ mạnh mẽ để thiết kế đồ họa, lập trình, và quản lý các yếu tố của trò chơi.

Tại sao chọn Unity:

* Đa nền tảng: Unity cho phép bạn phát triển trò chơi và xuất bản trên nhiều nền tảng mà không phải viết lại mã nguồn cho mỗi nền tảng, tiết kiệm thời gian và công sức so với việc sử dụng các công cụ khác như Unreal Engine, mà chỉ mạnh mẽ hơn trong các dự án có yêu cầu đồ họa cao.
* Cộng đồng lớn: Unity có một cộng đồng người dùng rộng lớn và nhiều tài liệu hỗ trợ. Điều này rất hữu ích cho việc giải quyết vấn đề nhanh chóng và tiếp cận với các nguồn tài nguyên sẵn có.
* Công cụ mạnh mẽ: Unity hỗ trợ các công cụ thiết kế game toàn diện, từ đồ họa 2D/3D đến scripting (lập trình), giúp việc phát triển game trở nên dễ dàng hơn.
  + 1. Netcode for GameObjects

Netcode for GameObjects là một framework của Unity cho phép phát triển các trò chơi multiplayer với các kết nối mạng ổn định. Nó hỗ trợ các tính năng như trạng thái đồng bộ, quản lý kết nối và giao tiếp giữa các client và server, phù hợp với các trò chơi yêu cầu tính tương tác thời gian thực giữa các người chơi.

Các tính năng nổi bật:

* Đồng bộ hóa trạng thái: Netcode for GameObjects sử dụng các cơ chế như RPC (Remote Procedure Call) và snapshotting để đồng bộ hóa trạng thái trò chơi giữa các client, đảm bảo rằng tất cả người chơi đều có một trải nghiệm nhất quán trong thời gian thực. Snapshotting giúp ghi lại trạng thái của trò chơi tại một thời điểm nhất định và truyền tải chúng đến các client, trong khi RPC cho phép gọi các hàm từ xa giữa server và client.
* Quản lý kết nối: Netcode hỗ trợ các tính năng như kết nối, ngắt kết nối, và xử lý các lỗi kết nối, giúp đảm bảo trò chơi có thể hoạt động mượt mà ngay cả khi có sự cố về mạng. Điều này rất quan trọng trong môi trường multiplayer để duy trì tính ổn định và độ chính xác của trận đấu.
* Bảo mật: Netcode for GameObjects cung cấp các cơ chế bảo mật để bảo vệ server khỏi các cuộc tấn công và gian lận, bao gồm mã hóa dữ liệu và xác thực kết nối, giúp ngăn chặn các hành vi gian lận của người chơi và bảo vệ thông tin cá nhân.

Sử dụng trong thực tế:

* Netcode for GameObjects rất phù hợp cho các trò chơi multiplayer có tính cạnh tranh cao, bao gồm:
* Game bắn súng góc nhìn thứ nhất (FPS): Cung cấp trải nghiệm multiplayer mượt mà với khả năng đồng bộ hóa các yếu tố như vị trí, hành động và hiệu ứng giữa các người chơi.
* Game MOBA (Multiplayer Online Battle Arena): Đảm bảo rằng tất cả các client đều có thể tham gia vào các trận đấu phức tạp với nhiều người chơi mà không gặp phải độ trễ.
* Game Battle Royale: Hỗ trợ hàng trăm người chơi trong một bản đồ rộng lớn, giúp giữ cho các sự kiện trong trò chơi đồng bộ và chính xác.
* Game đối kháng trực tuyến: Cho phép các trận đấu đối kháng trực tiếp giữa các người chơi với độ trễ thấp và đồng bộ hóa hiệu quả.
  + 1. Unity Services (Unity Cloud)

Unity Cloud cung cấp một loạt các dịch vụ hỗ trợ các game online và multiplayer, giúp giảm tải công việc về cơ sở hạ tầng cho nhà phát triển.

* Authentication Service (Dịch vụ Xác thực): Dịch vụ này giúp xác thực người chơi thông qua các phương thức như tài khoản Unity, Google, Facebook, hoặc email. Việc này giúp bảo mật và quản lý người dùng một cách dễ dàng và hiệu quả.
* Lobby Service (Dịch vụ Lobby): Dịch vụ này cho phép tạo phòng chờ (lobby), nơi người chơi có thể tìm kiếm và tham gia vào các trận đấu trực tuyến. Nó hỗ trợ matchmaking để tìm đối thủ phù hợp, cũng như tạo một không gian cho người chơi trò chuyện trước khi bắt đầu.
* Relay Service (Dịch vụ Relay): Relay Service giúp thiết lập kết nối P2P (peer-to-peer) giữa các client, hỗ trợ truyền tải dữ liệu giữa các người chơi mà không yêu cầu kết nối trực tiếp giữa các client. Điều này giúp giảm độ trễ trong các trò chơi online và cải thiện trải nghiệm người chơi.

Lợi ích khi sử dụng Unity Services:

* Tăng tốc độ phát triển: Các dịch vụ như Cloud Build và Analytics giúp giảm thiểu thời gian và công sức cần thiết để thiết lập cơ sở hạ tầng, cho phép các nhà phát triển tập trung vào việc thiết kế gameplay và các tính năng sáng tạo.
* Tập trung vào gameplay: Với việc sử dụng Unity Services, các nhà phát triển có thể để các công việc liên quan đến cơ sở hạ tầng và quản lý hệ thống vào tay các công cụ của Unity, thay vì phải tự phát triển các hệ thống này, giúp họ tập trung vào việc tối ưu hóa trải nghiệm người chơi.
* Tiết kiệm chi phí: Unity Services cung cấp các giải pháp đám mây, giúp nhà phát triển tiết kiệm chi phí về cơ sở hạ tầng, máy chủ và các chi phí vận hành khác, đồng thời giúp trò chơi có thể mở rộng quy mô dễ dàng.

Những công nghệ và dịch vụ trên không chỉ giúp tiết kiệm thời gian trong quá trình phát triển, mà còn đảm bảo trò chơi hoạt động mượt mà, bảo mật và dễ mở rộng. Unity, với các công cụ mạnh mẽ và dịch vụ đám mây tích hợp, mang đến một nền tảng lý tưởng cho việc phát triển game nhiều người chơi và game trực tuyến.

* 1. Công cụ và môi trường phát triển
     1. Rider IDE / Visual Studio

Rider IDE và Visual Studio là hai công cụ phát triển phần mềm phổ biến, đặc biệt đối với lập trình C# và phát triển các trò chơi Unity. Mỗi công cụ có những tính năng và ưu điểm riêng biệt hỗ trợ quá trình lập trình và phát triển game.

Rider là một IDE (Integrated Development Environment) được phát triển bởi JetBrains, tối ưu cho các ngôn ngữ lập trình như C# và hỗ trợ Unity game development rất mạnh mẽ. Rider mang lại các tính năng như:

* Intelligent code completion: Hỗ trợ tự động hoàn thành mã với tính năng phân tích cú pháp thông minh, giúp lập trình viên viết mã nhanh chóng và chính xác.
* Refactoring: Các công cụ để tái cấu trúc mã nguồn, giúp cải thiện và tối ưu hóa mã mà không làm thay đổi hành vi của ứng dụng.
* Debugging: Rider cung cấp các công cụ debug mạnh mẽ, cho phép theo dõi các tiến trình trong code, kiểm tra lỗi và sửa lỗi dễ dàng.
* Cross-platform support: Rider hỗ trợ cả Windows và macOS, giúp lập trình viên phát triển trên nhiều hệ điều hành.

Visual Studio là một IDE phổ biến khác dành cho lập trình C# và phát triển ứng dụng với Unity. Với một bộ công cụ mạnh mẽ cho lập trình viên, Visual Studio giúp quản lý dự án, sửa lỗi và tối ưu hóa quá trình phát triển:

* Rich Debugging: Visual Studio cung cấp một hệ thống debug rất mạnh mẽ, cho phép lập trình viên kiểm tra và sửa lỗi trong thời gian thực, theo dõi biến, bộ nhớ và luồng chương trình.
* Extensions and Plugins: Visual Studio có một hệ thống plugin và mở rộng rộng rãi, cho phép các lập trình viên thêm các công cụ bổ sung phù hợp với nhu cầu phát triển của họ.
* Code Analysis and Refactoring: Tích hợp công cụ phân tích mã và tái cấu trúc mã, giúp đảm bảo mã nguồn luôn rõ ràng và dễ bảo trì.
* Cả Rider và Visual Studio đều cung cấp các tính năng mạnh mẽ, nhưng Rider có thể được ưu tiên cho các nhà phát triển Unity nhờ sự tích hợp chặt chẽ với Unity và hỗ trợ C#.

Tại sao chọn Rider IDE / Visual Studio:

* Tích hợp với Unity: Cả Rider IDE và Visual Studio đều cung cấp tích hợp tuyệt vời với Unity, giúp việc lập trình, gỡ lỗi, và kiểm tra mã nguồn trở nên đơn giản hơn.
* Môi trường làm việc mạnh mẽ: Các công cụ này đều có khả năng hỗ trợ phân tích mã nguồn, tìm lỗi, gỡ rối trực tiếp trong quá trình phát triển game, giúp tiết kiệm thời gian.
* Cộng đồng hỗ trợ và tài liệu phong phú: Rider và Visual Studio có cộng đồng phát triển lớn và tài liệu hỗ trợ đầy đủ, giúp người lập trình dễ dàng giải quyết vấn đề gặp phải trong quá trình phát triển.
* Hỗ trợ các tính năng nâng cao: Cả hai IDE này cung cấp các tính năng như IntelliSense (hoàn thành mã tự động), hỗ trợ quản lý dự án, công cụ gỡ lỗi mạnh mẽ, và tính năng kiểm tra mã nguồn (code refactoring).
  + 1. Aseprite:

Aseprite là một phần mềm chuyên dụng để tạo và chỉnh sửa đồ họa pixel (pixel art), rất phổ biến trong việc phát triển các trò chơi 2D. Aseprite cung cấp một bộ công cụ đơn giản nhưng hiệu quả cho việc tạo ra các hình ảnh pixel với các tính năng sau:

* Pixel Art Creation: Aseprite cho phép tạo các sprite và đồ họa pixel chi tiết, dễ dàng điều chỉnh các pixel riêng biệt, rất thích hợp cho các trò chơi 2D hoặc các trò chơi có phong cách retro.
* Animation Support: Aseprite cung cấp công cụ tạo và chỉnh sửa animation pixel art, giúp bạn tạo các chuỗi hoạt hình đơn giản hoặc phức tạp trực tiếp trong phần mềm.
* Layer Management: Aseprite hỗ trợ quản lý nhiều lớp hình ảnh (layers), giúp dễ dàng tổ chức các phần khác nhau của hình ảnh, từ đó tạo ra các hiệu ứng hoạt hình hoặc chỉnh sửa chi tiết.
* Color Palettes: Aseprite hỗ trợ các bảng màu tùy chỉnh, giúp tạo ra các tác phẩm với màu sắc đồng bộ và dễ điều chỉnh theo yêu cầu của trò chơi.

Tại sao chọn Aseprite:

* Tính chuyên dụng: Aseprite được thiết kế đặc biệt để tạo pixel art, mang đến giao diện trực quan và các công cụ phù hợp giúp tối ưu hóa quy trình vẽ các sprite 2D. Đây là công cụ lý tưởng cho những ai phát triển game có phong cách đồ họa pixel hoặc 2D.
* Công cụ hoạt hình mạnh mẽ: Aseprite hỗ trợ vẽ hoạt hình pixel với các khung hình liên tiếp, giúp tạo ra các hoạt ảnh sống động cho nhân vật, đối tượng trong game. Điều này rất hữu ích trong việc tạo các hoạt động như di chuyển, tấn công, hoặc các chuyển động khác trong game.
* Dễ sử dụng: Aseprite có giao diện người dùng dễ tiếp cận, giúp nhà phát triển và họa sĩ đồ họa tạo ra các hình ảnh 2D mà không gặp phải sự phức tạp của các phần mềm thiết kế đồ họa khác như Photoshop.
* Hỗ trợ xuất sprite sheets: Aseprite cho phép xuất các sprite sheets và animation sheets để sử dụng trực tiếp trong Unity, giúp tích hợp trực tiếp vào game mà không cần phải chuyển đổi qua lại giữa các định dạng khác.
  1. Design Pattern

Design Pattern (Mẫu thiết kế) là các giải pháp đã được kiểm chứng cho các vấn đề thiết kế phần mềm thường gặp. Chúng là những cấu trúc, phương thức hoặc quy trình có thể tái sử dụng và giúp các lập trình viên giải quyết những vấn đề phổ biến trong việc xây dựng phần mềm. Các mẫu thiết kế giúp tổ chức mã nguồn một cách hiệu quả, dễ duy trì và mở rộng, đồng thời cải thiện khả năng tái sử dụng mã.

Các mẫu thiết kế thường được phân loại thành ba nhóm chính:

* Creational Patterns: Các mẫu thiết kế liên quan đến việc tạo đối tượng.
* Structural Patterns: Các mẫu thiết kế liên quan đến cách tổ chức và cấu trúc các lớp và đối tượng.
* Behavioral Patterns: Các mẫu thiết kế liên quan đến việc xử lý giao tiếp và các hành vi của các đối tượng trong hệ thống.
  + 1. Observer Pattern:

Observer Pattern là một mẫu thiết kế hành vi, cho phép các đối tượng (gọi là Observers) đăng ký và nhận thông báo khi trạng thái của đối tượng khác (gọi là Subject) thay đổi. Thay vì các đối tượng phải gọi trực tiếp nhau, các đối tượng quan sát sẽ tự động nhận thông báo từ đối tượng chủ khi có sự thay đổi.

Cách thức hoạt động:

* Subject là đối tượng mà các đối tượng quan sát sẽ lắng nghe thay đổi trạng thái.
* Observer là các đối tượng sẽ được thông báo khi có sự thay đổi từ Subject.
* Khi có sự thay đổi trong Subject, tất cả các Observer đã đăng ký sẽ nhận được thông báo và thực hiện hành động tương ứng.

Ứng dụng trong game: Dùng để cập nhật giao diện hoặc các thành phần khác khi có thay đổi trong game, như cập nhật điểm số hoặc trạng thái người chơi

* + 1. Singleton Pattern: Thiết lập các module mà cho phép mọi module khác có quyền truy cập thông tin.

Singleton Pattern là một mẫu thiết kế cấu trúc đảm bảo chỉ có một thể hiện (instance) duy nhất của một lớp trong toàn bộ hệ thống và cung cấp một phương thức toàn cục để truy cập đến thể hiện này.

Cách thức hoạt động:

* Mẫu này đảm bảo rằng trong suốt vòng đời ứng dụng, chỉ có một thể hiện duy nhất của một lớp.
* Global Access Point: Singleton cung cấp một điểm truy cập toàn cục (thường thông qua một phương thức tĩnh) để các phần khác trong hệ thống có thể truy cập thể hiện này mà không cần phải tạo mới nó.
* Lợi ích: Giảm thiểu việc tạo ra các đối tượng dư thừa và giúp quản lý tài nguyên một cách hiệu quả.

Ứng dụng trong game: GameManager: Quản lý các trạng thái của trò chơi (chế độ chơi, điểm số, tiến độ), chỉ có một thể hiện của GameManager trong toàn bộ ứng dụng. Các phần khác như AudioManager cũng có thể là Singleton để đảm bảo chỉ có một điểm điều khiển âm thanh duy nhất trong game.

* + 1. State Pattern: Phần gameplay-loop, chia các phần công việc thành các trạng thái (state).

State Pattern là một mẫu thiết kế hành vi giúp quản lý các trạng thái khác nhau của một đối tượng mà không cần phải dùng quá nhiều câu lệnh điều kiện. Mẫu này phân chia các trạng thái của đối tượng thành các lớp riêng biệt, giúp đối tượng thay đổi hành vi tùy theo trạng thái của nó.

Cách thức hoạt động:

* State: Mỗi trạng thái của đối tượng được biểu diễn bởi một lớp riêng biệt. Mỗi lớp này chứa các hành vi đặc trưng cho trạng thái đó.
* Context: Đối tượng quản lý trạng thái, có thể thay đổi trạng thái của mình thông qua các hành động cụ thể.
* Khi trạng thái thay đổi, đối tượng Context sẽ chuyển sang lớp trạng thái mới, và hành vi của đối tượng cũng thay đổi theo trạng thái mới.

Ứng dụng trong game: Gameplay Loop: Trong game, có thể có các trạng thái như Building Phase, Combat Phase, và Post-Combat Phase. Mỗi trạng thái này có các hành vi khác nhau (ví dụ: trong Building Phase, người chơi có thể chọn các thành phần cho máy chiến đấu, trong khi trong Combat Phase, họ sẽ điều khiển nhân vật để chiến đấu). Khi trạng thái thay đổi, game chuyển sang hành vi phù hợp với giai đoạn mới.

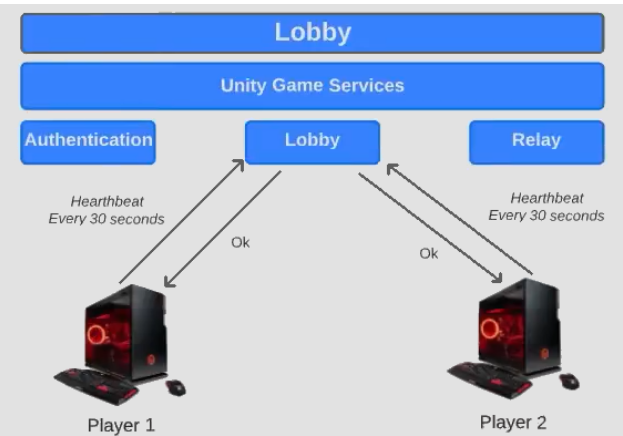
* 1. Kỹ thuật sử dụng
     1. HeartbeatLobbyRequest:

Mô tả: Một loại message chứa ít thông tin mà client gửi định kì tới server.

Mục đích: Cho server biết client này vẫn đang hoạt động.

Lợi ích:

* Nhận biết sự cố mất kết nối: nếu server không nhận được 1 heartbeat nào trong khoảng thời gian nhất định, client này sẽ được coi là đã mất kết nối (có thể do vấn đề về mạng, crash,..). Sau đó, server có thể xóa client này, và cập nhật lại lobby
* Duy trì session state: Heartbeats có thể mang 1 số thông tin cơ bản về client's state (Hp, trạng thái sẵn sàng), cho phép server nắm được 1 số thông tin mà không cần phải cập nhật liên tục

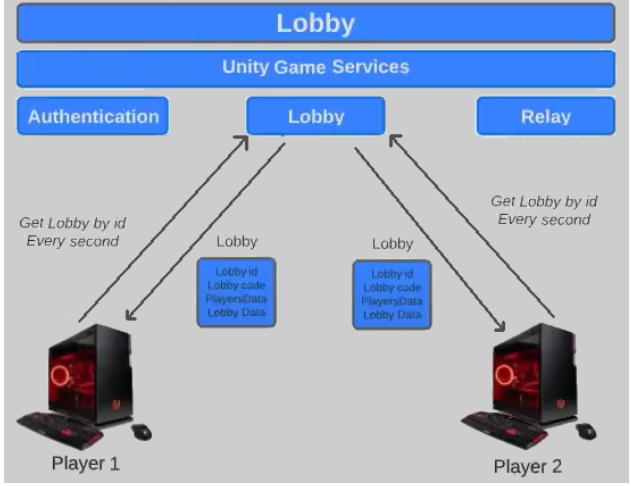


1. HeartbeatLobbyRequest
   * 1. RefreshLobbyRequest:

Mô tả: Một loại message clinet dùng để yêu cầu server gửi thông tin hiện tại của lobby.

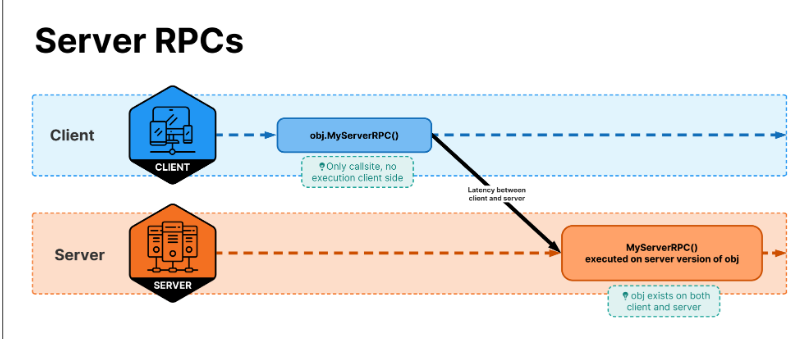
Mục đích: Client cập nhật những thông tin mới nhất của lobby tại local(danh sách player, trạng thái sẵn sàng của các player, các thuộc tính của lobby,..).

Lợi ích: Cập nhật lại lobby: trong 1 lobby, nhiều sự kiện có thể diễn ra: các player tham gia hoặc rời lobby, player thay đổi trạng thái sẵn sàng. Tất cả player trong lobby đều phải nhận được thông tin về những sự thay đổi này.

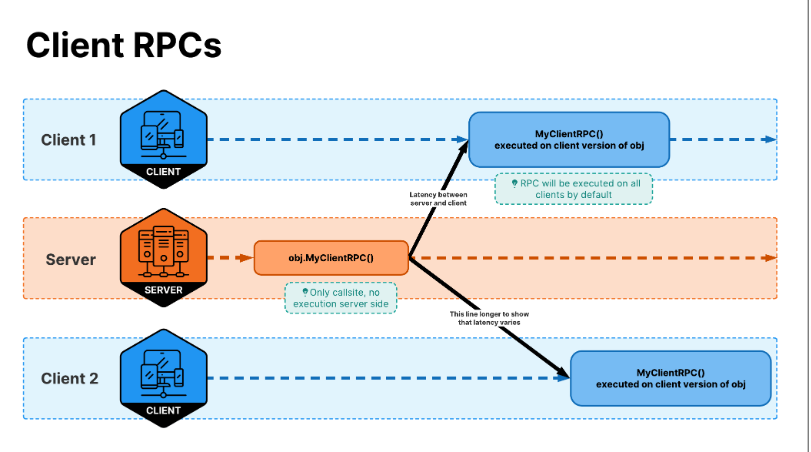


1. RefreshLobbyRequest
   * 1. Remote Procedure Call (RPC):

Mô tả: Kỹ thuật được sử dụng trong các hệ thống phân tán (vd: multiplayer game), nó cho phép một phần của hệ thống (client/server) thực thi một function hoặc procedure trên một phần khác (server/client).



1. Server RPCs



1. Client RPCs
   * 1. Client-side prediction:

Mô tả: Client lập tức cập nhật local (bắn ra viên đạn, di chuyển nhân vật,..) dựa trên thông tin đầu vào (player nhấn nút bắn, player di chuyển,..) ngay khi gửi cho server mà không cần quan tâm server có nhận được hay không

Mục đích: Phòng tránh hiện tượng lag: khi client phải đợi quá lâu để server gửi thông tin xác minh rồi mới cập nhật ở local

* + 1. Server Reconciliation:

Mô tả: Sau khi client tự cập nhật ở local, server sẽ xác minh lại dựa trên thông tin đầu vào nhận được, và gửi lại data về game-state cho client.

Mục đích: Tránh sai sót: client có thể tính toán sai, dựa trên nhiều yếu tố (khi client A cập nhật thông tin, A chưa nhận được thông tin mới nhất từ client B)

Client-side prediction & Server Reconciliation: client dự đoán (predict) để tạo cảm giác game có phản hồi tốt hơn, server điều chỉnh (reconcile) để tránh việc tính toán sai hoặc gian lận

* 1. Triển khai chức năng chính
     1. Tạo Lobby

Mô tả chức năng: Chức năng này cho phép người chơi tạo một phòng chờ (lobby) trong game để kết nối với những người chơi khác. Lobby này sẽ được tạo thông qua Lobby Service của Unity

Quy trình triển khai:

* Người chơi có thể tạo một lobby mới bằng cách sử dụng dịch vụ Lobby Service từ Unity.
* Khi tạo lobby, người chơi có thể chọn tên phòng, số lượng người chơi tối đa, và một số cài đặt khác như chế độ chơi.
* Lobby sẽ được lưu trữ trên server Unity và có thể chia sẻ với những người chơi khác để họ có thể tham gia.
* Sau khi tạo lobby, người chơi sẽ được chuyển đến phòng chờ (lobby) nơi có thể trò chuyện với các người chơi khác và chờ đợi trước khi bắt đầu trận đấu.

Lợi ích:

* Giảm thiểu việc tạo các máy chủ riêng biệt.
* Tăng tính linh hoạt trong việc quản lý các phòng chờ và kết nối người chơi.
  + 1. Tham gia Lobby thông qua mã code

Mô tả chức năng: Chức năng này cho phép người chơi tham gia vào một lobby đã có sẵn thông qua mã code được tạo ra khi lobby được tạo.

Quy trình triển khai:

* Khi người chơi tạo lobby, một mã code duy nhất sẽ được tạo ra và chia sẻ với những người chơi khác.
* Người chơi muốn tham gia chỉ cần nhập mã code vào giao diện của game để được kết nối với lobby tương ứng.
* Sau khi nhập mã code, hệ thống sẽ kiểm tra tính hợp lệ của mã và, nếu đúng, người chơi sẽ được đưa vào lobby.

Lợi ích:

* Tạo điều kiện cho việc tham gia nhanh chóng và thuận tiện mà không cần tìm kiếm phòng theo tên.
* Giảm thiểu khả năng nhập nhầm mã code và giúp tìm phòng chính xác hơn.
  + 1. Quay lại Lobby sau khi thoát game

Mô tả chức năng: Chức năng này cho phép người chơi quay lại lobby sau khi thoát game mà không phải nhập lại thông tin hoặc mã code.

Quy trình triển khai:

* Khi người chơi thoát game, thông tin về lobby và trạng thái của người chơi sẽ được lưu trữ trên server Unity.
* Sau khi người chơi quay lại game, hệ thống sẽ tự động khôi phục trạng thái của họ và đưa họ trở lại lobby mà họ đã tham gia trước đó, nếu vẫn còn hoạt động.
* Nếu lobby đã đóng, người chơi có thể được chuyển đến một lobby khác hoặc tạo mới một lobby.

Lợi ích:

* Cải thiện trải nghiệm người chơi khi không cần phải tìm lại phòng hoặc thông tin đã mất.
* Đảm bảo rằng người chơi có thể tiếp tục tham gia game mà không bị gián đoạn quá lâu.
  1. Luồng hoạt động của game
     1. Từ lúc vào game cho tới khi thoát

Khởi động game:

Khi người chơi vào game, giao diện chính của game sẽ hiển thị các lựa chọn như Chơi ngay, Cài đặt, Xem lịch sử, Thoát.

Nếu người chơi chọn Chơi ngay, hệ thống sẽ chuyển họ tới màn hình lựa chọn chế độ chơi (Chế độ đơn, đối kháng, v.v.).

Tạo phòng (Lobby):

Khi người chơi chọn Tạo phòng, game sẽ thực hiện các bước sau:

Tạo lobby: Sử dụng Lobby Service của Unity, một lobby mới sẽ được tạo ra. Mã phòng (room code) duy nhất sẽ được tạo và lưu trữ trên server. Mã này sẽ được cung cấp cho người chơi để chia sẻ với bạn bè hoặc đối thủ muốn tham gia.

Cài đặt lobby: Người chơi có thể cài đặt thêm một số tùy chọn như chế độ chơi, số lượng người chơi tối đa, hay loại trận đấu (PvP, Coop,...).

Chuyển đến Lobby: Người chơi sẽ được đưa đến lobby mới, nơi họ có thể chờ đợi các người chơi khác tham gia. Giao diện lobby sẽ hiển thị thông tin phòng, các lựa chọn trò chuyện và các nút để bắt đầu trận đấu.

Tham gia lobby:

Các người chơi khác có thể tham gia vào lobby này bằng cách nhập mã code phòng hoặc thông qua giao diện tìm kiếm phòng.

Sau khi đủ số lượng người chơi hoặc khi phòng đã sẵn sàng, game có thể tự động bắt đầu trận đấu.

Kết thúc game và thoát:

Sau khi trận đấu kết thúc, người chơi có thể chọn Quay lại lobby hoặc Thoát game.

Nếu người chơi chọn Thoát game, trạng thái game sẽ được lưu trữ, bao gồm việc đóng lobby và ngắt kết nối với các người chơi khác.

Nếu người chơi chọn Quay lại lobby, họ sẽ trở lại phòng chờ ban đầu (nếu phòng vẫn còn mở) và có thể tiếp tục tham gia hoặc tìm phòng mới.

* + 1. Quản lý trạng thái Lobby

Quản lý trạng thái của Lobby là một yếu tố quan trọng để đảm bảo rằng người chơi có thể tham gia phòng chơi một cách hiệu quả và không gặp phải sự cố trong quá trình kết nối. Trạng thái của lobby có thể thay đổi theo thời gian và phụ thuộc vào hành động của người chơi hoặc server.

Các trạng thái chính của Lobby:

* Lobby đang chờ (Waiting State):
* Đây là trạng thái mặc định của lobby sau khi được tạo.
* Trong trạng thái này, lobby chỉ có người tạo phòng và các người chơi khác có thể tham gia vào phòng nếu biết mã code hoặc tìm thấy phòng.
* Lobby ở trạng thái này cho phép người chơi trò chuyện và chuẩn bị cho trận đấu.
* Lobby đã đầy (Full State):
* Khi đủ số lượng người chơi tham gia vào lobby (theo cài đặt của phòng), lobby sẽ chuyển sang trạng thái "Full".
* Lúc này, người chơi mới không thể tham gia vào phòng này và sẽ cần tìm một lobby khác hoặc chờ một người rời phòng.
* Lobby sẵn sàng (Ready State):
* Sau khi tất cả người chơi đã tham gia và các cài đặt cần thiết đã hoàn tất (ví dụ: chọn chế độ chơi, cài đặt đối thủ), lobby sẽ chuyển sang trạng thái Ready.
* Khi trạng thái này được kích hoạt, trò chơi có thể bắt đầu tự động hoặc thông qua hành động của người tạo phòng (ví dụ: nhấn nút "Bắt đầu").
* Lobby đang thi đấu (In-Game State):
* Sau khi trận đấu bắt đầu, lobby sẽ chuyển sang trạng thái In-Game. Trong trạng thái này, game sẽ bắt đầu, và lobby sẽ được đóng lại hoặc thay đổi để không cho phép người chơi tham gia.
* Thông thường, các chức năng như chat hoặc thay đổi cài đặt sẽ bị vô hiệu hóa trong khi trận đấu đang diễn ra.
* Lobby đã kết thúc (Closed State):
* Sau khi trận đấu kết thúc và người chơi quyết định rời phòng, lobby sẽ được đóng. Trạng thái này cũng có thể xảy ra nếu người tạo phòng rời phòng hoặc hệ thống tự động đóng lobby.
* Các người chơi sẽ được đưa trở lại giao diện chính hoặc có thể tham gia các lobby khác.

Lợi ích của việc quản lý trạng thái Lobby:

* Đảm bảo người chơi luôn biết được trạng thái của phòng và không bị lạc trong quá trình kết nối.
* Giảm thiểu lỗi và trục trặc trong quá trình tham gia phòng chơi.
* Tăng tính tổ chức và dễ dàng quản lý các phòng chơi với nhiều người tham gia.

1. KIỂM THỬ VÀ TỐI ƯU HÓA
   1. Các loại kiểm thử
      1. Kiểm thử chức năng
      2. Kiểm thử hiệu năng
      3. Kiểm thử tính cân bằng gameplay
   2. Tối ưu hóa hệ thống
      1. Tối ưu hóa hiệu suất game
      2. Cân bằng game
2. TỔNG KẾT VÀ ĐỊNH HƯỚNG PHÁT TRIỂN
   1. Tổng kết dự án
   2. Kết quả đạt được
   3. Hạn chế và bài học kinh nghiệm
   4. Định hướng phát triển trong tương lai

KẾT LUẬN

Qua quá trình tìm hiểu và tham khảo, Nhóm đã xây dựng một website quản lý bán hàng nữ trang với những tính năng cơ bản giúp khách hàng tiếp cận được với của hàng. Sau khi hoàn thành đề tài, chúng em phần được củng cố thêm tư duy trong việc xây dựng dự án thực

Hướng phát triển: tiếp tục xây dựng hoàn chỉnh các tình năng của của Web hiện có. Phát triển thêm các tính năng mới và cuối cùng là triển khai dự án trên quy mô lớn.

BẢNG PHÂN CÔNG CÔNG VIỆC

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Họ và Tên | Công việc | Mức độ hoàn thành |
| 1 | Nguyễn Thị Phương Anh |  |  |
| 2 | Hoàng Thị Hiên |  |  |
| 3 | Vũ Bảo Lâm |  |  |
| 4 | Bùi Thị Thu Trang |  |  |
| 5 | Lương Thị Trang |  |  |

TÀI LIỆU THAM KHẢO

Internet

1. <https://startbootstrap.com/theme/sb-admin-2>