

HỌC VIỆN CÔNG NGHỆ BUỒU CHÍNH VIỄN THÔNG  
KHOA ĐA PHƯƠNG TIỆN

-----oo0-----



BÁO CÁO

LẬP TRÌNH GAME NÂNG CAO

ĐỀ TÀI: XÂY DỰNG GAME SPACESHIP SỬ DỤNG PHOTON  
ENGINE TRÊN NỀN TẢNG UNITY3D

Giảng viên hướng dẫn: TS.Phạm Vũ Minh Tú

Nhóm: 10

Thành viên nhóm

Mã sinh viên

- |                      |            |
|----------------------|------------|
| 1. Nguyễn Minh Quang | B19DCPT185 |
| 2. Bùi Thị Mai       | B19DCPT154 |
| 3. Phạm Văn Đang     | B19DCPT040 |
| 4. Nguyễn Văn Hưng   | B19DCPT115 |

Hà Nội – 2023

# MỤC LỤC

<b>MỤC LỤC.....</b>	<b>2</b>
<b>DANH MỤC HÌNH ẢNH.....</b>	<b>4</b>
<b>DANH MỤC BẢNG.....</b>	<b>7</b>
<b>DANH MỤC THUẬT NGỮ VIẾT TẮT.....</b>	<b>8</b>
<b>CHƯƠNG 1: CƠ SỞ LÝ THUYẾT .....</b>	<b>11</b>
I. Tìm hiểu về Multiplayer Game .....	11
1. Tổng quan về game và multiplayer game .....	11
2. Nguồn gốc hình thành .....	15
3. Phân loại Multiplayer game .....	17
4. Ưu điểm và nhược điểm của multiplayer game .....	20
5. Xu hướng chơi Multiplayer games .....	24
II. Tìm hiểu Game design document .....	26
1. GDD là gì?.....	26
2. Cấu trúc của 1 GDD .....	26
3. Beat chart.....	32
4. GDD one page .....	33
III. Các giao thức mạng .....	34
1. Khái niệm .....	34
2. Các giao thức kết nối mạng .....	35
3. Các loại giao thức kết nối mạng trong multiplayer game .....	36
c. WebSocket.....	43
4. Các đơn vị cung cấp giao thức mạng .....	47
IV. Tìm hiểu về Photon Engine .....	52
1. Khái niệm và tính năng .....	52
2. Ứng dụng của Photon Engine.....	53
3. Ưu nhược điểm của photon engine .....	54
4. Các ngôn ngữ lập trình và nền tảng hỗ trợ .....	54
5. Photon Unity Networking .....	55

<b>V. Tiêu kết chương 1 .....</b>	65
<b>CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG .....</b>	66
<b>I. Mô tả bài toán .....</b>	66
<b>II. Kiến trúc hệ thống.....</b>	67
<b>III. Phân tích thiết kế hệ thống.....</b>	69
1. Use case.....	69
2. Scenario .....	70
3. Biểu đồ lớp phân tích .....	77
4. Biểu đồ tuần tự .....	78
<b>IV. GDD game .....</b>	85
1. GDD .....	85
2. GDD one page .....	92
<b>V. Wireframe .....</b>	98
<b>CHƯƠNG 3: CÀI ĐẶT VÀ TRIỂN KHAI GAME .....</b>	105
<b>I. Cài đặt game.....</b>	105
<b>II. Triển khai game.....</b>	108
1. Giao diện Splash screen .....	108
2. Giao diện chọn chế độ .....	111
3. Giao diện chế độ chơi đơn.....	112
4. Giao diện chọn bản đồ.....	115
5. Giao diện cài đặt.....	116
6. Giao diện giới thiệu game .....	118
7. Giao diện chế độ nhiều người chơi .....	119
8. Giao diện tạo phòng .....	121
9. Giao diện phòng .....	123
10. Giao diện danh sách phòng .....	127
11. Giao diện trò chơi.....	129
<b>III. Tiêu kết chương 3 .....</b>	136
<b>DANH MỤC TÀI LIỆU THAM KHẢO.....</b>	137

## **DANH MỤC HÌNH ẢNH**

Hình 1.1. 1: Một số game nhiều người chơi phổ biến hiện nay .....	14
Hình 1.1. 2: Game Spacewar (phát hành năm 1962).....	15
Hình 1.1. 3 Game Pong (phát hành năm 1972) .....	16
Hình 1.1. 4: Game Doom.....	17
Hình 1.1. 5: Game Liên minh huyền thoại và game Fortnite .....	17
Hình 1.1. 6. Kết nối bạn bè bốn phương.....	21
Hình 1.1. 7. Cảm xúc của người chơi .....	22
Hình 1.1. 8. Game Liên minh huyền thoại.....	22
Hình 1.1. 9. Hack nhìn xuyên tường trong game bắn súng .....	23
Hình 1.1. 10. Sân thi đấu giải Liên minh huyền thoại .....	25
Hình 1.1. 11. Một vài game đa nền tảng nổi tiếng.....	25
Hình 1.3. 1: Cấu trúc 1 giao thức TCP/IP .....	38
Hình 1.3. 2: Truyền tải dữ liệu trong UDP .....	40
Hình 1.3. 3: Cấu trúc giao thức UDP .....	41
Hình 1.3. 4: Cấu trúc giao thức WebSockets.....	44
Hình 1.3. 5: Truyền dữ liệu trong giao thức HTTP .....	46
Hình 1.4. 1. Photon Dashboard.....	56
Hình 1.4. 2. Add PUN 2.....	57
Hình 1.4. 3. Import PUN 2.....	57
Hình 1.4. 4. Paste AppId đã copy vào để setup .....	58
Hình 1.4. 5. Kiểm tra xem đã kết nối.....	58
Hình 1.4. 6. Tham gia phòng trong Photon .....	59
Hình 1.4. 7. Add component Photon View cho đối tượng .....	62
Hình 1.4. 8. Gán Photon View bằng code .....	62
Hình 2.3. 1. Use case hệ thống game SpaceShip .....	69
Hình 2.3. 2: Biểu đồ lớp phân tích chế độ chơi đơn .....	77
Hình 2.3. 3:Biểu đồ lớp chế độ đa người chơi.....	77
Hình 2.3. 4: Biểu đồ lớp các thành phần trong game .....	78
Hình 2.3. 5: Biểu đồ tuần tự đăng nhập .....	79
Hình 2.3. 6: Biểu đồ tuần tự chọn chế độ chơi .....	79
Hình 2.3. 7: Biểu đồ tuần tự chọn chế độ .....	80
Hình 2.3. 8: Biểu đồ tuần tự chơi trong chế độ chơi đơn .....	80
Hình 2.3. 9: Biểu đồ tuần tự tạo phòng.....	81
Hình 2.3. 10: Biểu đồ tuần tự vào phòng ngẫu nhiên .....	81
Hình 2.3. 11: Biểu đồ tuần tự chọn tham gia phòng.....	82

Hình 2.3. 12: Biểu đồ tuần tự bắt đầu game .....	82
Hình 2.3. 13: Biểu đồ tuần tự phá hủy thiên thạch .....	83
Hình 2.3. 14: Biểu đồ tuần tự kết thúc game .....	84
Hình 2.3. 15: Biểu đồ tuần tự chọn map .....	84
Hình 2.3. 16: Biểu đồ tuần tự cài đặt .....	84
Hình 2.4. 1. Màn hình đăng nhập.....	89
Hình 2.4. 2. Sảnh chờ.....	89
Hình 2.4. 3. Tạo room .....	90
Hình 2.4. 4: Vào phòng .....	90
Hình 2.4. 5: Room List .....	91
Hình 2.4. 6: Phòng chơi .....	91
Hình 2.4. 7: Các model 3D trong game .....	92
Hình 2.4. 8: Ảnh phi thuyền.....	93
Hình 2.4. 9: Hình ảnh thiên thạch .....	93
Hình 2.5. 1: Wireframe màn hình đăng nhập .....	98
Hình 2.5. 2: Wireframe Đăng nhập và Thanh chờ.....	98
Hình 2.5. 3: Wireframe Chọn chế độ.....	99
Hình 2.5. 4: Wireframe Lobby của chế độ chơi đơn .....	99
Hình 2.5. 5: Wireframe Lobby của chế độ nhiều người chơi .....	100
Hình 2.5. 6: Wireframe Vào phòng .....	100
Hình 2.5. 7: Wireframe Tạo phòng .....	101
Hình 2.5. 8: Wireframe Danh sách phòng .....	101
Hình 2.5. 9: Wireframe Chọn map.....	102
Hình 2.5. 10: Wireframe About us.....	102
Hình 2.5. 11: Wireframe Setting .....	103
Hình 2.5. 12: Wireframe Giao diện màn chơi .....	103
Hình 2.5. 13: Wireframe Kết thúc game.....	104
Hình 3.1. 1: Photon Dashboard.....	105
Hình 3.1. 2: Tạo ứng dụng trong Photon .....	105
Hình 3.1. 3: Add PUN 2.....	106
Hình 3.1. 4. Import PUN 2.....	106
Hình 3.1. 5: Paste AppId đã copy vào để setup .....	107
Hình 3.1. 6: Kiểm tra xem đã kết nối.....	108
Hình 3.2. 1: Splash screen.....	108
Hình 3.2. 2: Giao diện màn đăng nhập .....	109
Hình 3.2. 3: Giao diện chọn chế độ chơi .....	111
Hình 3.2. 4: Giao diện chế độ 1 người chơi.....	112

Hình 3.2. 5: Giao diện chọn bản đồ .....	115
Hình 3.2. 6: Giao diện cài đặt .....	116
Hình 3.2. 7: Giao diện giới thiệu game.....	118
Hình 3.2. 8: Giao diện chế độ đa người chơi .....	119
Hình 3.2. 9: Giao diện tạo phòng.....	121
Hình 3.2. 10: Giao diện trong phòng chờ chơi game.....	123
Hình 3.2. 11: Giao diện danh sách phòng.....	127
Hình 3.2. 12: Giao diện chơi game .....	129
Hình 3.2. 13: Giao diện thông kê khi va chạm .....	129
Hình 3.2. 14: Giao diện kết thúc trò chơi .....	130

## **DANH MỤC BẢNG**

Bảng 2.3. 1: Kịch bản chức năng đăng nhập game.....	70
Bảng 2.3. 2: Kịch bản chức năng chọn chế độ chơi .....	70
Bảng 2.3. 3: Kịch bản chức năng chọn chế độ chơi đơn .....	71
Bảng 2.3. 4: Kịch bản chức năng tạo phòng .....	72
Bảng 2.3. 5: Kịch bản chức năng vào phòng ngẫu nhiên .....	72
Bảng 2.3. 6: Kịch bản chức năng chọn phòng và tham gia phòng .....	73
Bảng 2.3. 7: Kịch bản chức năng bắt đầu chơi game .....	73
Bảng 2.3. 8: Kịch bản chức năng bắn phá thiên thạch.....	74
Bảng 2.3. 9: Kịch bản chức năng phi thuyền va chạm với thiên thạch .....	74
Bảng 2.3. 10: Kịch bản kết thúc game .....	75
Bảng 2.3. 11: Kịch bản chọn map.....	76
Bảng 2.3. 12: Kịch bản cài đặt game .....	76
Bảng 2.4. 1: Bảng các nút điều khiển trong game .....	86
Bảng 2.4. 2: Cơ chế tính điểm trong game .....	86
Bảng 2.4. 3: Các đối tượng trong game .....	87
Bảng 2.4. 4: Màu sắc các phi thuyền xuất hiện trong game .....	88
Bảng 2.4. 5: Beat chart map 1 .....	95
Bảng 2.4. 6: Beat chart map 2 .....	96
Bảng 2.4. 7: Beat chart map 3 .....	97
Bảng 3.2. 1: Các button giao diện đăng nhập .....	109
Bảng 3.2. 2: Các button màn hình chọn chế độ chơi .....	111
Bảng 3.2. 3: Các button giao diện chế độ chơi đơn .....	113
Bảng 3.2. 4: Các button giao diện chọn bản đồ .....	115
Bảng 3.2. 5: Các button giao diện cài đặt .....	117
Bảng 3.2. 6: Các button giao diện giới thiệu game.....	118
Bảng 3.2. 7: Các button giao diện chế độ đa người chơi .....	119
Bảng 3.2. 8: Các text input giao diện tạo phòng .....	122
Bảng 3.2. 9: Các button giao diện tạo phòng.....	122
Bảng 3.2. 10: Các button giao diện phòng chờ game .....	124
Bảng 3.2. 11: Các button giao diện danh sách phòng.....	127
Bảng 3.2. 12: Các button trong giao diện chơi game .....	130

## **DANH MỤC THUẬT NGỮ VIẾT TẮT**

STT	Ký hiệu viết tắt	Chữ viết đầy đủ
1	LAN	Local Area Network
2	MMO	Massively Multiplayer Online
3	WAN	Wide Area Network
4	VPN	Virtual Private Network
5	PAN	Personal Area Network
6	WLAN	Wireless Local Area Network
7	MAN	Metropolitan Area Network
8	UDP	User Datagram Protocol
9	TCP/IP	Transmission Control Protocol/Internet Protocol
10	HTTP	Hypertext Transfer Protocol
11	URL	Uniform Resource Locator
12	FTP	File Transfer Protocol
13	SMTP	Simple Mail Transfer Protocol
14	DNS	Domain Name System
15	ACK	Acknowledgment Number
16	DHCP	Dynamic Host Configuration Protocol
17	SSL	Secure Sockets Layer
18	TLS	Transport Layer Security
19	ICMP	Internet Control Message Protocol
20	ARP	Address Resolution Protocol
21	ACK	Acknowledgment Number
22	SYN	Synchronize

23	FIN	Finish
24	PSH	Push
25	URG	Urgent
26	RST	Reset
27	DoS	Denial of Service
28	VPN	Virtual private Network
29	RPCs	Remote Procedure Calls
30	API	Application Programming Interface
31	DLC	Data Link Control
32	PC	Personal Computer
33	UI	User Interface
34	GDD	Game Design Document
35	AWS	Amazon Web Services
36	GCP	Google Cloud Platform
37	SDK	Software Development Kit
38	PUN	Photon Unity Networking
39	DLL	Dynamic Link Library
40	CCU	Concurrent Users

## LỜI CẢM ƠN

Trước tiên với tình cảm sâu sắc và chân thành nhất, cho phép nhóm em được bày tỏ lòng biết ơn đến thầy cô trong khoa Đa phương tiện đã tạo ra môn học này để bọn em học thêm được nhiều kiến thức bổ ích.

Với lòng biết ơn sâu sắc nhất, em xin gửi lời cảm ơn đến thầy Phạm Vũ Minh Tú đã truyền đạt vốn kiến thức quý báu cho chúng em trong suốt thời gian qua. Nhờ có những lời hướng dẫn và lời những góp ý sâu sắc của thầy nhóm em đã có thêm nhiều kiến thức về môn Lập trình Game nâng cao này.

Một lần nữa, em xin chân thành cảm ơn thầy Phạm Vũ Minh Tú – người đã trực tiếp giúp đỡ, quan tâm, hướng dẫn nhóm em hoàn thành tốt bài báo cáo này trong thời gian qua.

Do điều kiện thời gian và kiến thức còn hạn chế nên bài báo cáo nhóm em không tránh khỏi những hạn chế, thiếu sót, nhóm em rất mong nhận được những ý kiến đóng góp quý báu của thầy để những bài báo cáo sau này của nhóm em được tốt hơn

Cuối cùng nhóm em xin kính chúc thầy cô trong khoa Đa phương tiện có thật nhiều sức khỏe và luôn thành công trên nhiều lĩnh vực

Nhóm em xin chân thành cảm ơn!

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

## I. Tìm hiểu về Multiplayer Game

### 1. Tổng quan về game và multiplayer game

#### a. Game là gì?

Game (hay còn gọi là trò chơi điện tử) là một hoạt động giải trí trong đó người chơi tham gia vào một trải nghiệm tương tác với một mục tiêu cụ thể, thường là để đạt được một số điểm, chiến thắng hoặc vượt qua các cấp độ khó khăn hơn.

##### ❖ Đặc điểm của game

- Mục đích chơi game: Mục đích chơi game thường là để giải trí và thư giãn, hoặc để đạt được một mục tiêu nhất định.
- Quy tắc và hướng dẫn: Game có các quy tắc cụ thể để chơi và thường được cung cấp hướng dẫn để người chơi hiểu rõ cách thức chơi.
- Gameplay: Gameplay là cốt lõi của game, bao gồm các tính năng, cấp độ, thể loại và các yếu tố khác. Gameplay cần phải được thiết kế đơn giản, dễ hiểu và dễ chơi để thu hút sự quan tâm của người chơi. Gameplay cũng cần phải được thiết kế sao cho mang tính thử thách, phong phú và thú vị để người chơi có thể trải nghiệm và tận hưởng những giây phút thú vị trong game
- Cấu trúc và cấp độ: Game thường được chia thành các cấp độ khác nhau, với mức độ khó tăng dần. Người chơi cần hoàn thành các cấp độ này để tiếp tục chơi.
- Tính đa dạng và linh hoạt: Tính đa dạng và linh hoạt trong game là một đặc điểm quan trọng để tạo ra sự thú vị và lôi cuốn cho người chơi. Người chơi có nhiều lựa chọn về thể loại, chế độ chơi, tính năng và các nhiệm vụ để người chơi có thể trải nghiệm và tận hưởng những giây phút thú vị trong game.

- Đồ họa và âm thanh: Đồ họa là một yếu tố quan trọng của game, tạo nên một thế giới ảo đẹp mắt và sống động. Đồ họa càng chân thực và đẹp mắt thì sẽ tạo ra ấn tượng và trải nghiệm tốt hơn cho người chơi.
- Cộng đồng chơi game: Người chơi có thể tham gia vào cộng đồng chơi game để kết nối với những người chơi khác và thảo luận về các vấn đề liên quan đến game.
- Thời gian chơi game: Game có thể được chơi trong một khoảng thời gian ngắn hoặc kéo dài trong nhiều giờ đồng hồ tùy thuộc vào thể loại và mục đích chơi.

#### ❖ Tính chất của game

- Quy luật và thách thức: Game đưa ra quy luật và thách thức cho người chơi, bao gồm các mục tiêu và hành động cần thực hiện để hoàn thành trò chơi.
- Cung cấp trải nghiệm: Game cung cấp cho người chơi một trải nghiệm mới lạ, thú vị và giúp họ tạo ra cảm giác hứng thú khi tham gia.
- Sự phát triển kỹ năng: Game giúp người chơi phát triển các kỹ năng, như phản xạ, tư duy, kỹ năng xử lý thông tin và trực quan.
- Sự tương tác: Game thường có tính tương tác, cho phép người chơi tương tác với các đối tượng trong trò chơi, cũng như tương tác với những người chơi khác nếu là game multiplayer.
- Sự đa dạng: Các game có độ đa dạng khác nhau về thể loại, nền tảng, gameplay và cấp độ khó, tạo ra sự đa dạng cho người chơi trong việc lựa chọn và trải nghiệm trò chơi.
- Sự kết nối: Game có thể kết nối người chơi từ khắp nơi trên thế giới, tạo nên một cộng đồng game đa dạng và thu hút.
- Tính sáng tạo: Game cho phép người chơi tự sáng tạo và khám phá các giải pháp mới cho các thách thức trong game.

#### ❖ Phân loại

- **Theo nền tảng:** Game có thể được chơi trên nhiều nền tảng khác nhau, chẳng hạn như máy tính, điện thoại di động, máy chơi game, máy tính bảng và các nền tảng khác.
- **Theo thể loại:** Các thể loại game phổ biến bao gồm game hành động, game nhập vai, game chiến lược, game thể thao, game đua xe, game giải đố, game bắn súng, game kinh dị, game giáo dục, game mô phỏng, game âm nhạc, game cảm giác mạnh, game phiêu lưu...
- **Theo phong cách:** Game có thể được phân loại theo phong cách của chúng, bao gồm game 2D, game 3D, game pixel art, game anime...
- **Theo mức độ độc lập:** Game độc lập là game do các nhà phát triển độc lập phát triển mà không được phát hành bởi một công ty lớn. Trái ngược với đó, game AAA là các game được phát triển bởi các công ty lớn, có ngân sách lớn và thường được quảng bá rộng rãi.
- **Theo đối tượng chơi:** Game có thể được phân loại theo đối tượng chơi của chúng, bao gồm game cho trẻ em, game dành cho người lớn, game giáo dục, game đào tạo, game giải trí...

#### b. Multiplayer game

Multiplayer games được biết là trò chơi điện tử nhiều người chơi, tại đây game thủ sẽ bước chân vào thế giới game kết nối trực tuyến với nhiều người chơi tại thời gian thực. Multiplayer cho phép mọi người chiến đấu với các đối thủ hoặc hợp tác với nhau để đạt được mục tiêu cuối cùng.



*Hình 1.1. 1: Một số game nhiều người chơi phổ biến hiện nay*

❖ Đặc điểm của multiplayer game

- Sự tương tác giữa người chơi: Game multiplayer cho phép người chơi tương tác với nhau thông qua chơi đội, giao tiếp qua chat hoặc âm thanh, đối đầu với nhau trong trận đấu và cạnh tranh với nhau trong các hoạt động
- Độ khó tăng dần: Các game multiplayer thường có cấp độ khó tăng dần để đảm bảo người chơi phải thực hiện cố gắng hơn và có thể thăng tiến với thời gian chơi
- Sự đa dạng về gameplay: Game multiplayer có thể đa dạng về gameplay, cho phép người chơi lựa chọn trải nghiệm chơi game mà họ thích
- Sự cộng đồng và kết nối: Game multiplayer có thể kết nối người chơi từ khắp nơi trên thế giới, tạo nên sự đa dạng và hấp dẫn trong cộng đồng chơi game
- Cơ hội giao lưu và học hỏi: Game multiplayer cũng có thể mang lại cơ hội cho người chơi giao lưu, học hỏi kinh nghiệm và kỹ năng từ các người chơi khác trong cộng đồng game

- Độ thú vị và giải trí cao: Game multiplayer thường cho phép người chơi thử thách và cạnh tranh với những người chơi khác, giúp tăng tính thú vị và giải trí của người chơi
- Yêu cầu kết nối Internet: Game multiplayer yêu cầu kết nối internet và có thể yêu cầu máy tính hoặc thiết bị chơi game có cấu hình tương đối cao để đảm bảo chất lượng trải nghiệm chơi game

## 2. Nguồn gốc hình thành

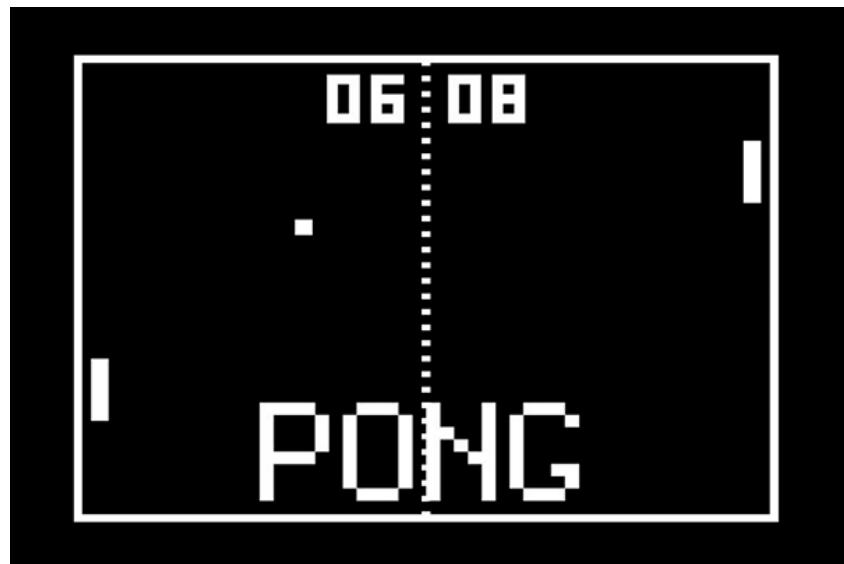
Multiplayer game đã xuất hiện được vài thập kỷ, nhưng nguồn gốc của chúng có thể bắt nguồn từ những ngày đầu của trò chơi điện tử và máy tính.

Một trong những ví dụ sớm nhất về trò chơi nhiều người chơi là trò chơi Spacewar, được tạo ra vào năm 1962 bởi một nhóm các nhà khoa học máy tính tại Viện Công nghệ Massachusetts (MIT). Spacewar! là một trò chơi hai người chơi được chơi trên máy tính lớn và có sự tham gia của hai phi thuyền chiến đấu với nhau trong môi trường không gian mô phỏng.



Hình 1.1. 2: Game Spacewar (phát hành năm 1962)

Trong những năm 1970 và 1980, trò chơi nhiều người chơi trở nên phổ biến hơn với sự ra đời của máy chơi game gia đình và máy tính cá nhân. Các trò chơi như Pong, được phát hành vào năm 1972, cho phép hai người chơi thi đấu với nhau trong một trò chơi đơn giản giống như quần vợt.



*Hình 1.1. 3 Game Pong (phát hành năm 1972)*

Khi công nghệ tiên tiến, trò chơi nhiều người chơi cũng vậy. Vào những năm 1990, trò chơi nhiều người chơi trực tuyến bắt đầu xuất hiện, cho phép người chơi kết nối internet và đấu với nhau trong thời gian thực. Các trò chơi như Doom và Quake trở nên phổ biến nhờ chế độ nhiều người chơi, cho phép người chơi cạnh tranh với nhau trong các trận chiến bắn súng góc nhìn thứ nhất có nhịp độ nhanh.



Hình 1.1. 4: Game Doom

Kể từ đó, trò chơi nhiều người chơi đã tiếp tục phát triển và trở nên phức tạp hơn, với các trò chơi như World of Warcraft, Fortnite và Liên minh huyền thoại quy tụ hàng triệu người chơi từ khắp nơi trên thế giới để chơi cùng nhau trong môi trường ảo rộng lớn.



Hình 1.1. 5: Game Liên minh huyền thoại và game Fortnite

### 3. Phân loại Multiplayer game

#### a. Dựa trên kết nối

##### ❖ Multiplayer games offline

Game Multiplayer offline là dạng game mà người chơi có thể chơi cùng lúc trên cùng một máy tính hoặc mạng LAN mà không yêu cầu kết nối

internet. Loại game này cho phép nhiều người chơi tham gia vào cùng một trận đấu hoặc cùng một chế độ chơi.

Các đặc điểm của game Multiplayer offline bao gồm:

- Tính tương tác trực tiếp: Người chơi có thể tương tác trực tiếp với nhau trên cùng một màn hình hoặc qua kết nối LAN. Điều này tạo ra cảm giác gần gũi, kết nối giữa các người chơi hơn.
- Đa dạng chế độ chơi: Game Multiplayer offline thường có nhiều chế độ chơi, cho phép người chơi thỏa sức khám phá và trải nghiệm. Ví dụ như chế độ đấu đội, chế độ đấu một mình, chế độ đua xe, chế độ bắn súng,...
- Không bị giới hạn bởi tốc độ internet: Với Game Multiplayer offline, người chơi không cần phải lo lắng về tốc độ internet, độ trễ kết nối hay vấn đề liên quan đến mạng. Điều này giúp trò chơi chạy mượt mà hơn và tránh được các vấn đề liên quan đến mạng.
- Tính hợp tác cao: Game Multiplayer offline thường có tính hợp tác cao, cho phép người chơi cùng hợp sức với nhau để vượt qua các thử thách trong trò chơi.
- Tính thách thức và cạnh tranh: Game Multiplayer offline cũng có tính thách thức và cạnh tranh khi các người chơi cùng tranh tài để giành chiến thắng.

Một số ví dụ về game Multiplayer offline phổ biến bao gồm: FIFA, Call of Duty, Counter-Strike, Mario Party, Age of Empires,...

#### ❖ Multiplayer games online

Multiplayer games online là dạng game mà người chơi kết nối với nhau thông qua internet, cho phép nhiều người chơi cùng tham gia vào cùng một trận đấu hoặc chế độ chơi. Loại game này được phổ biến rộng rãi trên các nền tảng như PC, console và thiết bị di động.

Các đặc điểm của Multiplayer games online bao gồm:

- **Tính tương tác giữa người chơi:** Multiplayer games online cho phép người chơi tương tác với nhau trên một nền tảng trực tuyến thông qua mạng internet. Người chơi có thể chơi cùng nhau, giao lưu và kết nối với nhau trong trò chơi.
- **Đa dạng chế độ chơi:** Multiplayer games online thường có nhiều chế độ chơi khác nhau, bao gồm chế độ đấu đội, chế độ đấu một mình, chế độ đua xe, chế độ bắn súng, chế độ chiến lược và nhiều hơn nữa.
- **Tốc độ và độ trễ kết nối:** Tốc độ và độ trễ kết nối đóng vai trò quan trọng trong Multiplayer games online. Điều này đảm bảo rằng trò chơi chạy mượt mà và không bị gián đoạn.
- **Tính hợp tác và cạnh tranh cao:** Multiplayer games online có tính hợp tác cao, cho phép người chơi hợp tác với nhau để đạt được mục tiêu chung. Ngoài ra, Multiplayer games online cũng có tính cạnh tranh cao khi các người chơi cùng tranh tài để giành chiến thắng.
- **Cộng đồng game online:** Multiplayer games online thường có cộng đồng game trực tuyến phát triển quanh nó. Các người chơi có thể giao lưu, trao đổi và tham gia các hoạt động trong cộng đồng game.

Một số ví dụ về Multiplayer games online phổ biến bao gồm: League of Legends, World of Warcraft, Fortnite, PUBG, Apex Legends, Overwatch,...

### b. Dựa trên thể loại game

- **Co-op games:** Co-op games là dạng game có tính chất hợp tác giữa người chơi, trong đó các người chơi cùng phối hợp để đạt được một mục tiêu chung. Co-op games có thể chơi trực tuyến hoặc offline, và thường được phát triển cho nhiều nền tảng như PC, console và thiết bị di động. Một số ví dụ về Co-op games phổ biến bao gồm: Left 4 Dead, Portal 2, Borderlands, Overcooked, Don't Starve Together,...

- **Competitive games:** Competitive games là loại game mà người chơi phải cạnh tranh với nhau để giành chiến thắng. Trong các trò chơi này, người chơi thường phải chơi trong một môi trường đấu đội hoặc đối đầu trực tiếp với nhau để đạt được mục tiêu chung của trò chơi. Một số ví dụ về competitive games phổ biến bao gồm: League of Legends, Overwatch, Counter-Strike: Global Offensive, Fortnite và Dota 2
- **Battle Royale games:** Battle Royale là một thể loại game nơi một nhóm lớn người chơi (thường là 100 người hoặc nhiều hơn) đấu tranh để sống sót và trở thành người chơi cuối cùng. Trong game Battle Royale, người chơi sẽ được thả xuống một khu vực đồ bộ trên một hòn đảo, sau đó tìm kiếm vũ khí, vật phẩm và thiết bị để chiến đấu và sống sót trong khi khu vực giảm dần  
Ví dụ về các trò chơi battle royale bao gồm PlayerUnknown's Battlegrounds, Apex Legends và Fortnite.
- **MMO (Trò chơi trực tuyến nhiều người chơi):** MMO là viết tắt của Massive Multiplayer Online, đây là thể loại game trực tuyến nhiều người chơi được phát triển để cho phép hàng nghìn người chơi tham gia vào một thế giới ảo, tương tác với nhau và tham gia vào các hoạt động và cuộc phiêu lưu. Ví dụ về MMO bao gồm World of Warcraft, Guild Wars 2 và Final Fantasy XIV.
- **Sports games:** Sports games là trò chơi nhiều người chơi mô phỏng các môn thể thao trong thế giới thực như bóng đá, bóng rổ và bóng đá,. Ví dụ về các trò chơi thể thao bao gồm FIFA, NBA 2K và Madden NFL.

#### **4. Ưu điểm và nhược điểm của multiplayer game**

##### **a. Ưu điểm**

- **Kết nối bạn bè**

Multiplayer game có số người chơi cực khủng mang bạn bước chân vào thế giới của các game thủ đa quốc gia cùng các nền văn hóa đa dạng. Chẳng hạn như các tựa game nức tiếng PUBG, Fortnite, ... đem đến với trận đấu với hơn 99 game thủ khác nhau, một con số tuyệt vời để giao lưu bạn bè bốn phương.



*Hình 1.1. 6. Kết nối bạn bè bốn phương*

Hơn hết, Multiplayer sở hữu lợi thế lớn là số lượng người chơi đông đảo nên cộng đồng game thủ “mọc lên như nấm”, người chơi có thể dễ dàng kết nối với những đồng đội cùng chí hướng một cách dễ dàng.

- **Gia tăng trải nghiệm**

Trong multiplayer game người chơi tha hồ thể hiện hết kỹ năng chơi game của mình, được so tài với nhiều người chơi từ những quốc gia khác nhau góp phần giúp người chơi chìm đắm vào những trải nghiệm tuyệt vời.



*Hình 1.1. 7. Cảm xúc của người chơi*

Và khi chơi game với nhiều người với nhiều trình độ khác nhau giúp bạn nâng cao hơn kỹ năng của bản thân, rút ngắn thời gian “nâng trình” của bản thân.

- **Chiến thuật phức tạp**

Đối với những game thiên về chiến thuật yêu cầu người chơi phải dành nhiều thời gian và tốn nhiều chất xám để nghĩ cho mình một bước đi thông minh. Một số game mang đậm tính chiến thuật nổi tiếng như LOL, Dota, Đế chế, ...



*Hình 1.1. 8. Game Liên minh huyền thoại*

Tính chiến thuật phức tạp trong Multiplayer cũng là ưu điểm sáng giá giúp người chơi rèn luyện tư duy và trí thông minh một cách đáng kể. Đây cũng được xem là một cách để game thủ thể hiện trình độ, bản lĩnh và tài trí của mình với bạn bè thế giới.

### b. Nhược điểm

- **Trình độ không đối xứng**

Trong Multiplayer games đây là một vấn đề nan giải, khi mà những tay gà mờ mới chơi sẽ gặp phải các người chơi có kỹ năng và cấp bậc cao hơn, việc bị ăn hành cũng không tránh khỏi

Nhưng giờ đây điều này cũng đã được hạn chế phần nào, khi giờ đây game đã có thể sắp xếp cho những người cùng cấp bậc gặp nhau và những người mới sẽ gặp những người mới và có cấp bậc thấp.

- **Gian lận**

Gian lận trong game là điều thường xuyên xảy ra trong game, những phần mềm gian lận giúp người chơi dễ dàng chiến thắng. Điều này làm mất cân bằng game và khiến những game thủ chân chính nản lòng và bỏ game



Hình 1.1. 9. Hack nhìn xuyên tường trong game bắn súng

- **Phụ thuộc vào internet**

Một vài vùng có kết nối mạng kém dẫn đến giật lag khi chơi game, làm gián đoạn trải nghiệm của người chơi.

- **Tốn thời gian và tiền bạc**

Nhiều game yêu cầu người chơi phải đầu tư thời gian vào chơi, nâng cấp kỹ năng, điều này làm ảnh hưởng đến một số người có thời gian hạn chế.

Bên cạnh đó nhiều game còn yêu cầu người chơi trả phí, nạp tiền để tiếp tục trải nghiệm hay mua đồ, gây mất cân bằng game bằng cách tăng phúc lợi cho người nạp tiền từ đó ép buộc người chơi phải đổ tiền vào.

## 5. Xu hướng chơi Multiplayer games

Trò chơi nhiều người chơi đã trở thành một xu hướng ngày càng phổ biến trong ngành công nghiệp trò chơi trong những năm gần đây. Một trong những xu hướng lớn nhất trong trò chơi nhiều người chơi là sự tăng của thể thao điện tử, là các cuộc thi được tổ chức trong đó các game thủ chuyên nghiệp thi đấu với nhau để giành giải thưởng và uy tín. Thể thao điện tử đã trở thành một hiện tượng toàn cầu, với hàng triệu khán giả theo dõi các sự kiện như giải Liên minh huyền thoại thế giới (League of Legends World Championship), giải Pubg thế giới (PUBG Global Championship), ... và rất nhiều giải khác.



Hình 1.1. 10. Sân thi đấu giải Liên minh huyền thoại

Một xu hướng khác của multiplayer game là sự gia tăng của các game đa nền tảng cho phép người dùng PC, hay thiết bị di động có thể cùng nhau chơi trong một thế giới game.



Tổng hợp 15 game đa nền tảng hấp dẫn nhất 2022

Hình 1.1. 11. Một vài game đa nền tảng nổi tiếng

Xu hướng của multiplayer game sẽ tiếp tục phát triển với các công nghệ và cải tiến mới giúp người chơi kết nối và chơi cùng nhau dễ dàng hơn bao giờ hết, bất kể họ ở đâu trên thế giới.

## II. Tìm hiểu Game design document

### 1. GDD là gì?

Game Design Document là một hoặc một bộ tài liệu mô tả hầu hết các thiết kế sẽ có trong game của bạn. Là nơi lưu trữ toàn bộ những khái niệm, định nghĩa, mô tả nhân vật, thuộc tính, màn chơi...

Mục đích chính của Game Design Document là để truyền đạt thông tin chi tiết về dự án của bạn cho chính bạn khi bạn làm việc với trò chơi của mình theo thời gian hoặc cho những người khác, chẳng hạn như các thành viên trong nhóm, nhà phát hành, các bên liên quan hoặc những người sẽ chơi trò chơi của bạn, như một phần của chiến dịch gây quỹ cộng đồng hoặc sản phẩm truy cập sớm. Nói đơn giản GDD là tài liệu để **thống nhất thiết kế** và là tiền đề cho việc lên kế hoạch sản xuất.

Tài liệu này sẽ được cập nhật, thay đổi và hoàn thiện suốt thời gian phát triển sản phẩm.

### 2. Cấu trúc của 1 GDD

- ❖ **Mô tả trò chơi:** Đây là phần giới thiệu tổng quan về trò chơi, bao gồm các yếu tố như thể loại, cốt truyện và mục tiêu của người chơi.
  - **Thể loại:** Là cách phân loại game theo đặc điểm chung của chúng.  
Các thể loại game hiện nay bao gồm:
    - Game hành động (Action game): Là những trò chơi tập trung vào một hoặc nhiều nhân vật di chuyển và đánh đấm, bắn súng hoặc giải đố.
    - Game nhập vai (Role-playing game - RPG): Là trò chơi cho phép người chơi tham gia vào một vai trò, điều khiển nhân vật và thực hiện các nhiệm vụ để phát triển nhân vật và tạo ra một câu chuyện.

- Game chiến thuật (Strategy game): Là trò chơi đòi hỏi người chơi sử dụng tư duy chiến thuật để đưa ra quyết định, tập trung vào quản lý tài nguyên, quân đội và xây dựng căn cứ.
- Game thể thao (Sports game): Là trò chơi mô phỏng các môn thể thao, cho phép người chơi thực hiện các hoạt động thể thao như bóng đá, bóng rổ, tennis, đua xe, vv.
- Game giải đố(Puzzle game): Là trò chơi về giải đố, cho phép người chơi phải sử dụng khả năng suy luận logic của bản thân để tìm ra đáp án phá đảo trò chơi.

### • Cốt truyện

Là tập hợp các sự kiện, hành động và nhân vật trong trò chơi. Cốt truyện mô tả chi tiết cụ thể về những gì xảy ra trong trò chơi bao gồm các tình tiết, môi trường. Cốt truyện được sắp xếp thành các thứ tự logic để tạo ra một câu chuyện có tính logic gây cảm động và lôi cuốn người chơi.

### • Mục tiêu của người chơi

Mục tiêu của người chơi là việc người chơi cần phải làm gì để phá đảo được con game này hay đạt được một thành tựu nhất định

### ❖ Gameplay:

Phần này mô tả cách thức chơi của trò chơi. Nó bao gồm các phần như độ khó, cơ chế chơi, hướng dẫn chơi và các chế độ chơi khác.

Gameplay là một yếu tố tối quan trọng ảnh hưởng đến việc người chơi có gắn bó với sản phẩm game của nhà phát hành hay không nó là một trong những cách giúp cho các game thủ tương tác với trò chơi và ngược lại trò chơi tương tác đến với người trải nghiệm. Ở đây chúng ta có thể hiểu nôm na rằng Gameplay là một cái gì đó tổng quát

bao gồm các nhiệm vụ, các tính năng trong trò chơi giúp game thủ trải nghiệm được trọn vẹn hơn.

Đây là một trong những yếu tố phụ không hề ép buộc cần có trong một sản phẩm game của nhà phát hành nhưng nó lại là điều kiện tối quan trọng thu hút người chơi.

### ❖ Characters:

Đây là phần giới thiệu về các nhân vật trong trò chơi, bao gồm cả nhân vật chính và phụ. Nó cũng bao gồm các tính năng, kỹ năng và sức mạnh của từng nhân vật.

Trong game multiplayer, characters (nhân vật) được sử dụng để tạo ra các trải nghiệm đa dạng và phong phú cho người chơi. Mỗi character có một bộ kỹ năng và tính năng riêng, nhằm đem lại cảm giác độc đáo và phù hợp với phong cách chơi của từng người chơi. Dưới đây là một số điểm cần lưu ý khi thiết kế các characters cho game multiplayer:

**Đa dạng hóa các characters:** Các characters cần phải đa dạng và thú vị để giữ cho người chơi quan tâm và tiếp tục chơi.

**Cân bằng các characters:** Các characters cần phải được thiết kế để đảm bảo rằng tất cả các người chơi đang chơi cùng một trò chơi multiplayer đều có cơ hội chiến thắng, và không bị ảnh hưởng bởi sự khác biệt về tính năng và kỹ năng giữa các characters.

**Tính tương tác của các characters:** Các characters có thể cung cấp cơ hội cho các người chơi để tương tác với nhau, ví dụ như hợp tác để vượt qua các thử thách, hoặc chiến đấu chống lại nhau.

**Tính độc đáo của các characters:** Các characters cần phải có tính độc đáo riêng để tạo ra cảm giác độc đáo cho người chơi và khác biệt

so với các trò chơi multiplayer khác. Tính độc đáo có thể đến từ bộ kỹ năng đặc biệt của character, cách sử dụng trang bị hoặc phong cách chơi.

Tùy chỉnh và đổi mới: Các characters nên được thiết kế để cho phép người chơi tùy chỉnh hoặc tạo ra các characters mới, đảm bảo tính động của trò chơi multiplayer.

#### ❖ **Worlds/Levels:**

Phần này mô tả các thế giới hoặc cấp độ của trò chơi. Nó bao gồm mô tả về bối cảnh, cách thức tiến hành, những khó khăn trong từng thế giới/cấp độ và cách thức hoàn thành chúng.

Trong game multiplayer, worlds hoặc levels được sử dụng để tạo ra các trải nghiệm chơi game đa dạng và hấp dẫn cho người chơi. Mỗi world hoặc level đều có một mục tiêu cụ thể để hoàn thành, như vượt qua các thử thách, chiến thắng trong trận đấu, hoặc hoàn thành các nhiệm vụ khác nhau. Dưới đây là một số điểm cần lưu ý khi thiết kế worlds/levels cho game multiplayer:

**Đa dạng hóa các worlds/levels:** Để giữ cho người chơi quan tâm và tiếp tục chơi, các worlds/levels cần phải đa dạng và thú vị..

**Đồng bộ hóa các worlds/levels:** Các worlds/levels cần phải được thiết kế để đảm bảo rằng tất cả các người chơi trong cùng một trận đấu đều có cùng một trải nghiệm chơi game..

**Độ khó của các worlds/levels:** Các worlds/levels cần được thiết kế để phù hợp với năng lực của các người chơi. Nếu một world/level quá khó hoặc quá dễ, nó có thể dẫn đến sự chán nản của người chơi.

**Tương tác giữa các người chơi:** Các worlds/levels có thể cung cấp cơ hội để các người chơi tương tác với nhau, ví dụ như chiến đấu chống lại nhau hoặc hợp tác để vượt qua các thử thách.

Tùy chỉnh và đổi mới: Các worlds/levels nên được thiết kế để cho phép người chơi tùy chỉnh hoặc tạo ra các worlds/levels mới, đảm bảo tính động của trò chơi multiplayer.

### ❖ User Interface:

Đây là phần mô tả giao diện người dùng của trò chơi, bao gồm các nút bấm, thanh trạng thái, màn hình chính và các tính năng khác. User Interface (UI) trong game multiplayer rất quan trọng để giúp người chơi tương tác và giao tiếp với game cũng như các người chơi khác. UI đóng vai trò rất quan trọng trong việc cung cấp thông tin cho người chơi, cho phép họ thực hiện các hành động và tương tác với môi trường và các đối thủ của họ.

Dưới đây là một số lời khuyên để thiết kế UI trong game multiplayer:

- Đảm bảo UI phù hợp với mục đích của game: UI trong game multiplayer phải được thiết kế sao cho phù hợp với mục đích của game, đồng thời giúp cho người chơi dễ dàng tìm thấy thông tin và tương tác với game.
- Cung cấp các tính năng tương tác: UI cần cung cấp các tính năng tương tác như chat, menu, danh sách bạn bè, hướng dẫn và cài đặt. Điều này sẽ giúp cho người chơi dễ dàng tương tác và tìm thấy thông tin mà họ cần.
- Đơn giản và dễ sử dụng: UI cần được thiết kế đơn giản và dễ sử dụng, giúp người chơi dễ dàng tìm thấy và sử dụng các tính năng. Điều này cũng giúp cho game trở nên hấp dẫn hơn với các người chơi mới.

Đảm bảo tính tương tác: UI cần phải đảm bảo tính tương tác giữa

các người chơi trong trò chơi. Điều này có thể đạt được bằng cách cung cấp các tính năng như chat, trao đổi thông tin, v.v.

- Tùy chỉnh UI: UI cần cho phép người chơi tùy chỉnh và thay đổi các thiết lập UI của mình, như cỡ chữ, màu sắc, v.v.

#### ❖ **Audio:**

Phần này mô tả các yếu tố âm thanh của trò chơi, bao gồm nhạc nền, hiệu ứng âm thanh và giọng nói.

**Audio** đóng vai trò quan trọng trong việc tạo ra trải nghiệm trò chơi tuyệt vời cho người chơi. Bao gồm các yếu tố âm thanh sau:

Nhạc nền: Nhạc nền là một yếu tố âm thanh quan trọng trong game, nó có thể giúp tăng cường cảm giác của người chơi và tạo ra một bầu không khí đặc biệt. Nhạc nền thường được thiết kế để phù hợp với tình huống và cảm xúc của game, ví dụ như nhạc nền hồi hộp trong trận chiến hoặc nhạc nền yên bình trong cảnh quan đẹp.

Hiệu ứng âm thanh: Hiệu ứng âm thanh là các âm thanh nhỏ được phát ra khi người chơi thực hiện hành động nào đó trong game, ví dụ như tiếng bước chân khi nhân vật di chuyển, tiếng súng khi bắn đạn hoặc tiếng bật lò xo khi nhân vật nhảy lên cao. Hiệu ứng âm thanh giúp người chơi cảm thấy rõ ràng và chân thật hơn về việc họ đang tham gia vào trò chơi.

Giọng nói: Giọng nói của các nhân vật trong game được sử dụng để tương tác với người chơi. Nó có thể là giọng nói của các nhân vật trong game hoặc là giọng nói của một nhân vật độc lập, chẳng hạn như giọng nói của nhân vật hướng dẫn hoặc giọng nói của nhà phát triển. Giọng nói cũng có thể được sử dụng để giải thích các chức năng hoặc quy trình trong game cho người chơi, tạo ra một trải nghiệm chơi game tốt hơn.

## ❖ Artwork

Đây là phần giới thiệu về các hình ảnh và đồ họa của trò chơi, bao gồm các mô hình 3D, hình ảnh 2D và các hiệu ứng khác.

Artwork trong game đóng vai trò quan trọng trong việc tạo ra trải nghiệm trò chơi tuyệt vời cho người chơi, bao gồm:

Tạo sự hấp dẫn cho game: Artwork được sử dụng để tạo ra các hình ảnh đẹp mắt, cuốn hút để thu hút sự chú ý của người chơi. Điều này có thể là một trong những yếu tố quan trọng để người chơi lựa chọn và tham gia trò chơi.

Tạo cảm xúc cho game: Artwork trong game có thể tạo ra các cảm xúc khác nhau cho người chơi, từ các hình ảnh đẹp mắt và lãng mạn đến các hình ảnh kinh dị và đáng sợ. Những hình ảnh này có thể là một phần quan trọng của truyền tải thông điệp và tạo ra sự khác biệt trong trò chơi.

Giúp xác định nhân vật và thế giới trong game: Artwork cũng được sử dụng để tạo ra các hình ảnh của các nhân vật và thế giới trong game, giúp người chơi dễ dàng nhận ra và phân biệt các nhân vật và vật phẩm khác nhau trong trò chơi.

Thúc đẩy việc mở rộng thương hiệu: Artwork trong game có thể được sử dụng để tạo ra các sản phẩm liên quan đến trò chơi, chẳng hạn như áo phông, sách màu, vật phẩm ghi chép và các sản phẩm đồ chơi khác. Việc tạo ra các sản phẩm liên quan đến trò chơi có thể giúp mở rộng thương hiệu và tăng khả năng tiếp cận của trò chơi đến đối tượng khách hàng mới.

## 3. Beat chart

"Beat chart" là một công cụ thiết kế game được sử dụng để mô tả cấu trúc tổng thể của trò chơi. Nó được sử dụng để xác định các sự kiện và trạng

thái chính của trò chơi, để giúp các nhà thiết kế game quản lý quá trình phát triển và đảm bảo rằng trò chơi có cấu trúc hợp lý và thu hút người chơi.

#### ❖ Các thành phần chính của 1 beat chart

- **Name:** Đây là tên của màn chơi mà nhà phát triển sẽ đặt
- **Progression:** sự phát triển và diễn biến dần dần của câu chuyện và các nhân vật trong game
- **Story beat:** đề cập đến một sự kiện hoặc thời điểm cụ thể trong câu chuyện của trò chơi có ý nghĩa hoặc tác động quan trọng đến câu chuyện. Đó là một điểm quan trọng trong cốt truyện giúp đưa câu chuyện về phía trước và thường liên quan đến sự thay đổi động cơ hoặc hoàn cảnh của nhân vật.
- **Gameplay:** Cách chơi của màn chơi
- **Enemies:** Quái vật, kẻ thù mà người chơi cần tiêu diệt trong màn chơi
- **Estimated time:** Thời gian của màn chơi, tùy thuộc vào thể loại trò chơi và độ khó của màn chơi để người làm game đưa ra thời gian phù hợp.
- **Color:** Màu sắc chính trong màn chơi
- **Music:** Ở đây bao gồm nhạc nền và hiệu ứng âm thanh của màn chơi

#### 4. GDD one page

**GDD one-page** là một bản thu nhỏ của một GDD hoàn chỉnh, đóng vai trò tóm tắt những nội dung chính và cốt lõi của một game.

Một GDD one-page gồm các phần:

- Tổng quan về trò chơi: Đây phải là phần giới thiệu ngắn gọn về trò chơi, bao gồm thể loại, bối cảnh.
- Cơ chế chơi trò chơi: Phần này sẽ phác thảo cơ chế chơi trò chơi cốt lõi, chẳng hạn như hành động, mục tiêu và thử thách của người chơi.

- Cốt truyện: Phần này sẽ cung cấp một cái nhìn tổng quan ngắn gọn về câu chuyện của trò chơi, bao gồm nhân vật chính, nhân vật phản diện chính và các điểm cốt truyện chính.
- Nhân vật: Phần này sẽ giới thiệu các nhân vật chính của trò chơi, cung cấp một mô tả ngắn gọn về động cơ và tính cách của họ.
- Phong cách của đồ họa: Phần này sẽ mô tả phong cách đồ họa trong game và tính thẩm mỹ của trò chơi, bao gồm mọi yếu tố hình ảnh độc đáo hoặc lựa chọn thiết kế.
- Đối tượng mục tiêu: Phần này sẽ phác thảo đối tượng mục tiêu dự định cho trò chơi, bao gồm độ tuổi, giới tính và bất kỳ thông tin nhân khẩu học có liên quan nào khác.
- Nền tảng: Phần này sẽ liệt kê các nền tảng mà trò chơi sẽ được phát hành, chẳng hạn như PC, bảng điều khiển hoặc thiết bị di động.

### **III. Các giao thức mạng**

#### **1. Khái niệm**

Mạng (network) là một tập hợp các thiết bị và phần mềm được kết nối với nhau để truyền tải thông tin giữa các thiết bị đó. Một mạng bao gồm các thành phần như máy tính, thiết bị định tuyến, công cụ kết nối, dây cáp, đường truyền, giao thức truyền tải dữ liệu và các thành phần khác.

Mục đích của mạng là để truyền tải dữ liệu và chia sẻ tài nguyên giữa các thiết bị trong mạng. Các loại mạng có thể khác nhau về quy mô và phạm vi, từ các mạng máy tính cá nhân nhỏ đến các mạng toàn cầu phức tạp được sử dụng bởi các tổ chức lớn và internet.

Có các loại mạng khác nhau bao gồm mạng máy tính, mạng di động, mạng LAN (Local Area Network), mạng WAN (Wide Area Network), mạng VPN (Virtual Private Network), và internet.

## 2. Các giao thức kết nối mạng

Giao thức kết nối mạng là một tập hợp các quy tắc và quy định được sử dụng để thiết lập, duy trì và chấm dứt một kết nối mạng giữa hai hoặc nhiều thiết bị trong mạng máy tính. Giao thức kết nối mạng được sử dụng để đảm bảo rằng các thiết bị trong mạng có thể giao tiếp và truyền dữ liệu với nhau một cách hiệu quả và đáng tin cậy.

Các giao thức kết nối mạng thường được phân thành hai loại chính: giao thức phân phối và giao thức truyền tải.

- Giao thức phân phối (routing protocol) được sử dụng để quản lý và phân phối các gói dữ liệu giữa các mạng và các thiết bị định tuyến (router) trong mạng. Giao thức phân phối sẽ xác định đường đi tốt nhất cho các gói dữ liệu để chúng có thể đến được đích một cách nhanh chóng và hiệu quả.
- Giao thức truyền tải (transport protocol) được sử dụng để quản lý việc truyền tải dữ liệu giữa các thiết bị trong mạng. Giao thức truyền tải sẽ đảm bảo rằng dữ liệu được truyền tải đến đích một cách chính xác và đáng tin cậy.

Một số giao thức kết nối mạng phổ biến bao gồm TCP/IP, HTTP, FTP, SMTP, DNS và DHCP. Các giao thức này đóng vai trò quan trọng trong việc kết nối và truyền tải dữ liệu giữa các thiết bị trong mạng.

Trong game multiplayer, các giao thức kết nối mạng được sử dụng để cho phép người chơi trong một trò chơi trực tuyến có thể giao tiếp và tương tác với nhau thông qua mạng Internet. Các giao thức này giúp đảm bảo tính năng lưu lượng mạng, độ trễ thấp và ổn định trong các trò chơi trực tuyến.

### **3. Các loại giao thức kết nối mạng trong multiplayer game**

#### **a. TCP/IP**

##### **❖ Khái niệm**

TCP/IP (Transmission Control Protocol/Internet Protocol - Giao thức điều khiển truyền nhận/ Giao thức liên mạng) là một bộ giao thức truyền thông được sử dụng để kết nối các thiết bị với internet và truyền dữ liệu qua internet.

TCP/IP bao gồm hai giao thức chính: TCP và IP. TCP cung cấp khả năng phân phối các gói dữ liệu theo thứ tự, đáng tin cậy giữa các thiết bị qua mạng. Nó chia một lượng lớn dữ liệu thành các gói nhỏ hơn, sau đó được truyền và tập hợp lại ở đầu nhận. TCP cũng cung cấp khả năng kiểm soát luồng và kiểm tra lỗi để đảm bảo dữ liệu được truyền chính xác và không bị thất thoát.

IP cung cấp định tuyến các gói dữ liệu giữa các thiết bị qua internet. Nó xử lý việc đánh địa chỉ của các thiết bị, phân mảnh và lắp ráp lại các gói khi chúng được truyền qua internet.

Trong multiplayer game, TCP/IP được sử dụng để truyền dữ liệu giữa máy khách và máy chủ trò chơi qua internet. Người lập trình sử dụng mã mạng TCP/IP để thiết lập và quản lý kết nối giữa máy khách và máy chủ trò chơi, cũng như để gửi và nhận gói dữ liệu chứa thông tin trạng thái trò chơi, vị trí người chơi, đầu vào của người dùng và các dữ liệu khác liên quan đến trò chơi.

##### **❖ Mục đích và tính năng của giao thức**

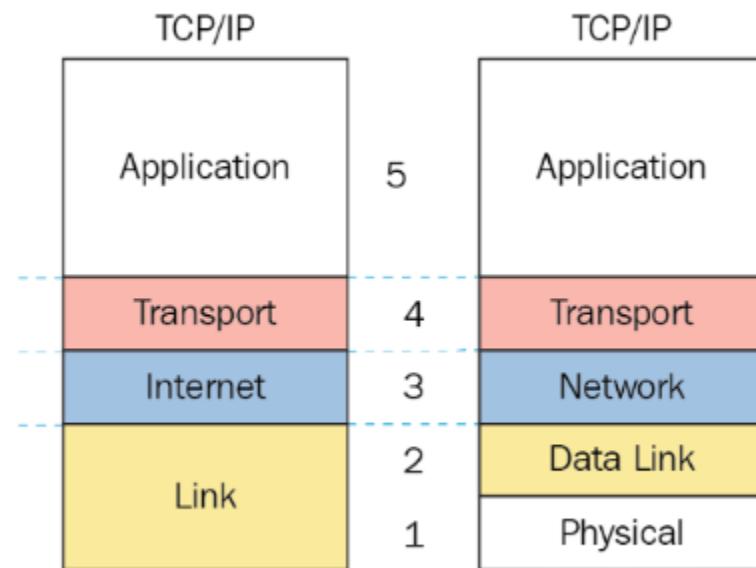
Bộ giao thức TCP/IP là một phần cơ bản trong cách vận hành của internet và nhiều mạng máy tính khác. Mục đích của nó là cung cấp một cách thức chuẩn hóa và đáng tin cậy để các thiết bị giao tiếp với nhau qua mạng, bất kể phần cứng hoặc phần mềm được sử dụng bởi những thiết bị.

Một số tính năng chính của giao thức TCP/IP bao gồm:

- **Định địa chỉ:** Giao thức IP cung cấp một cách định địa chỉ các thiết bị trên mạng được tiêu chuẩn hóa, cho phép các gói dữ liệu được định tuyến đến đích chính xác của chúng.
- **Độ tin cậy:** Giao thức TCP cung cấp khả năng gửi gói dữ liệu đáng tin cậy, đảm bảo rằng chúng được nhận theo đúng thứ tự và không có lỗi.
- **Tính linh hoạt:** TCP/IP hỗ trợ nhiều loại ứng dụng và thiết bị, làm cho nó trở thành một bộ giao thức đa năng và linh hoạt.
- **Khả năng mở rộng:** TCP/IP được thiết kế để hoạt động hiệu quả trên các mạng có quy mô khác nhau, từ mạng cục bộ nhỏ đến mạng internet toàn cầu.
- **Khả năng tương thích:** TCP/IP tương thích với nhiều loại hệ điều hành và phần cứng, cho phép các thiết bị có cấu hình khác nhau giao tiếp với nhau.
- **Bảo mật:** TCP/IP bao gồm các giao thức như SSL và TLS cung cấp liên lạc an toàn qua internet.

### ❖ Cấu trúc của giao thức

Một mô hình TCP/IP tiêu chuẩn bao gồm 4 lớp được chồng lên nhau, bắt đầu từ tầng thấp nhất là Tầng vật lý (Physical) → Tầng mạng (Network) → Tầng giao vận (Transport) và cuối cùng là Tầng ứng dụng (Application).



*Hình 1.3. 1: Cấu trúc 1 giao thức TCP/IP*

### ❖ Packet

Trong TCP/IP, dữ liệu được truyền dưới dạng các gói tin. Một gói thường bao gồm một tiêu đề và một trọng tải. Tiêu đề chứa thông tin như địa chỉ nguồn và đích, giao thức đang được sử dụng (ví dụ: TCP, UDP) và thông tin điều khiển khác. Tải trọng chứa dữ liệu thực tế được truyền đi.

Khi dữ liệu được gửi từ thiết bị này sang thiết bị khác qua mạng TCP/IP, dữ liệu sẽ được chia thành nhiều gói, với mỗi gói chứa một phần dữ liệu. Các gói này sau đó được gửi qua mạng và được tập hợp lại ở đầu nhận để tạo lại dữ liệu gốc.

Trong TCP, các gói còn được gọi là phân đoạn và chúng được sử dụng để thiết lập kết nối đáng tin cậy giữa hai điểm cuối. Các phân đoạn TCP chứa số thứ tự và xác nhận để đảm bảo phân phối đáng tin cậy và sắp xếp đúng thứ tự dữ liệu.

#### Cấu trúc của gói tin TCP bao gồm các trường sau:

- Port nguồn (Source Port): Đây là số hiệu cổng của ứng dụng nguồn, nó sẽ giúp cho ứng dụng đích nhận biết được gói tin đến từ đâu.

- Port đích (Destination Port): Đây là số hiệu cổng của ứng dụng đích, nó sẽ giúp cho ứng dụng đích biết được gói tin cần đến cho ai.
- Số thứ tự (Sequence Number): Đây là số thứ tự của gói tin, giúp cho các gói tin được sắp xếp theo đúng thứ tự khi truyền tải.
- Số ACK (Acknowledgment Number): Đây là số xác nhận của gói tin, nó cho biết gói tin đã được nhận đến đâu.
- Độ dài tiêu đề (Header Length): Đây là trường chứa độ dài của tiêu đề của gói tin.
- Byte trống (Reserved): Đây là trường dành trống cho mục đích sử dụng trong tương lai.
- Cờ (Flags): Đây là trường chứa các cờ điều khiển, bao gồm SYN, ACK, FIN, PSH, URG, và RST.
- Cửa sổ (Window): Đây là trường chứa kích thước của cửa sổ trượt, cho phép truyền tải dữ liệu theo chu kỳ.
- Kiểm tra định danh (Checksum): Đây là trường chứa giá trị checksum để kiểm tra tính toàn vẹn của gói tin.
- Con trỏ khẩn cấp (Urgent Pointer): Đây là trường chỉ ra vị trí của dữ liệu cần được ưu tiên truyền tải.
- Tùy chọn (Options): Đây là trường tùy chọn cho phép thêm thông tin phụ vào gói tin TCP.

#### ❖ **Tính bảo mật của giao thức**

TCP/IP được thiết kế có tính đến bảo mật và bao gồm một số giao thức cung cấp mã hóa, xác thực và tính toàn vẹn dữ liệu cho dữ liệu được truyền qua internet. Tuy nhiên, nó vẫn dễ dàng bị tấn công bởi nhiều kiểu tấn công khác nhau, chẳng hạn như tấn công từ chối dịch vụ (DoS) và đánh hơi gói tin, do đó, điều quan trọng là phải thực hiện các bước bổ sung để bảo vệ dữ liệu được truyền qua TCP/IP, chẳng hạn như sử dụng tường lửa và triển khai thực thi mã hóa an toàn

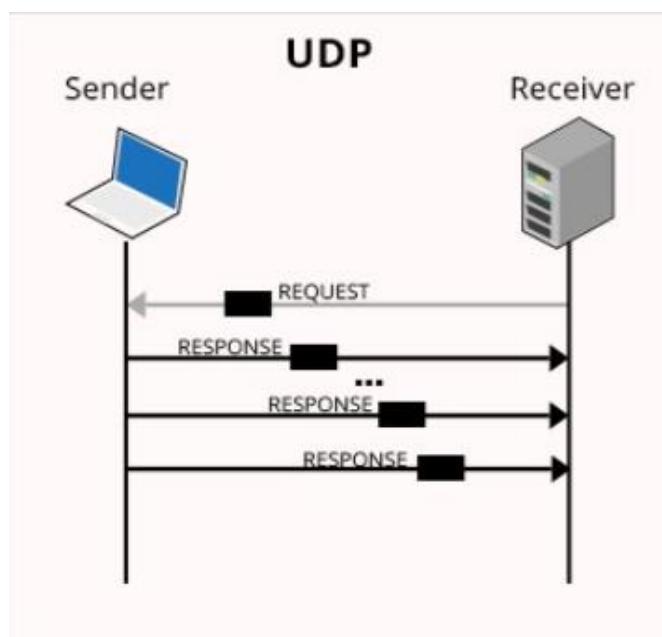
## b. UDP

### ❖ Khái niệm

User Datagram Protocol (UDP) thường được sử dụng để truyền tải các gói tin dữ liệu giữa các thiết bị trong mạng. UDP được ưa chuộng trong game multiplayer bởi vì nó có tính năng độ trễ thấp và tốc độ truyền tải nhanh, giúp đảm bảo sự liên tục và chính xác của các tác vụ trò chơi.

### ❖ Mục đích và tính năng của giao thức

Như đã nói trên, UDP được sử dụng trong game multiplayer vì nó cho phép truyền dữ liệu nhanh chóng và với độ trễ thấp hơn so với TCP. Điều này là quan trọng vì trong các trò chơi đa người chơi, các tín hiệu phải được truyền đi và đến các máy khách càng nhanh càng tốt để tránh độ trễ gây ra sự cố trong trò chơi.



Hình 1.3. 2: Truyền tải dữ liệu trong UDP

Một số tính năng quan trọng của UDP trong game multiplayer bao gồm:

- Tốc độ truyền dữ liệu nhanh hơn so với TCP.
- Độ trễ thấp hơn, giảm thiểu sự cố liên quan đến độ trễ trong trò chơi.
- Không yêu cầu thiết lập kết nối trước khi truyền dữ liệu, giảm thiểu độ trễ.

- Cho phép truyền dữ liệu nhanh chóng và không đảm bảo độ tin cậy, phù hợp với các trò chơi yêu cầu tốc độ và độ trễ thấp hơn độ tin cậy.
- Sử dụng cổng để xác định ứng dụng nhận dữ liệu, giúp đảm bảo dữ liệu được truyền đến đúng ứng dụng trong máy khách.

Tuy nhiên, do UDP không đảm bảo độ tin cậy và không có cơ chế xử lý lỗi dữ liệu, các lỗi như mất dữ liệu hoặc dữ liệu trùng lặp có thể xảy ra. Để giảm thiểu các sự cố này, các phần mềm game thường sử dụng các kỹ thuật và cơ chế bổ sung để đảm bảo tính đúng đắn của dữ liệu nhận được từ UDP.

### ❖ Cấu trúc của giao thức

Cấu trúc của gói tin UDP gồm 4 trường chính:

- Port nguồn (Source Port): Đây là số hiệu cổng của ứng dụng nguồn, nó sẽ giúp cho ứng dụng đích nhận biết được gói tin đến từ đâu.
- Port đích (Destination Port): Đây là số hiệu cổng của ứng dụng đích, nó sẽ giúp cho ứng dụng đích biết được gói tin cần đến cho ai.
- Độ dài gói tin (Length): Đây là trường chứa độ dài của toàn bộ gói tin, bao gồm cả tiêu đề và dữ liệu.
- Kiểm tra định danh (Checksum): Đây là trường chứa giá trị checksum để kiểm tra tính toàn vẹn của gói tin.



*Hình 1.3. 3: Cấu trúc giao thức UDP*

Các trường này được sắp xếp theo thứ tự từ trái sang phải trong gói tin UDP. Sau các trường này là dữ liệu thực sự được truyền tải trong gói tin.

=> Trong game multiplayer, UDP được sử dụng để truyền tải các thông tin như tọa độ của nhân vật, các hành động của người chơi, v.v. UDP có tính đơn giản và hiệu suất cao hơn TCP, tuy nhiên, nó không đảm bảo tính toàn vẹn dữ liệu và có thể góp phần làm giảm chất lượng trò chơi nếu không được sử dụng đúng cách. Trong trường hợp này, TCP thường được sử dụng thay thế cho UDP

### ❖ Packet

Packet của User Datagram Protocol (UDP) trong game multiplayer chứa thông tin để truyền từ máy chủ game đến các máy khách (clients) của người chơi hoặc ngược lại. Mỗi packet UDP bao gồm các trường sau:

- Port nguồn (Source Port): Đây là số cổng được sử dụng bởi máy gửi để gửi packet. Nó giúp máy nhận biết từ đâu packet được gửi đến.
- Port đích (Destination Port): Đây là số cổng được sử dụng bởi máy nhận để nhận packet. Nó cho phép các packet được gửi đến đúng ứng dụng trên máy khách.
- Độ dài (Length): Trường này xác định độ dài của packet tính bằng số byte.
- Checksum: Trường này chứa giá trị checksum (tổng kiểm tra), được sử dụng để kiểm tra tính chính xác của packet. Giá trị checksum được tính toán dựa trên nội dung của packet và được sử dụng để xác định xem package đã bị hỏng hay không.
- Dữ liệu (Data): Trường này chứa dữ liệu cần truyền đi.
- Packet UDP được gửi đi mà không yêu cầu thiết lập kết nối trước khi truyền dữ liệu. Khi packet được nhận, các ứng dụng trên máy khách có thể trích xuất dữ liệu từ packet và sử dụng chúng để hiển thị thông tin hoặc điều khiển các hoạt động trong trò chơi

### ❖ Tính bảo mật của giao thức

UDP là một giao thức có tính bảo mật kém không đảm bảo việc gửi và nhận các gói tin đúng thứ tự hoặc đảm bảo rằng chúng đã được gửi và nhận một cách an toàn và chính xác.

Do đó, nếu bạn muốn đảm bảo tính bảo mật của ứng dụng sử dụng giao thức UDP, bạn cần sử dụng các giải pháp bảo mật bổ sung như mã hóa và xác thực gói tin để đảm bảo rằng các thông tin được truyền qua UDP không bị xâm nhập hoặc thay đổi bởi người ngoài. Ví dụ, VPN (Virtual Private Network) thường sử dụng UDP để truyền dữ liệu nhưng cung cấp các giải pháp bảo mật bổ sung để đảm bảo an toàn và bảo mật cho dữ liệu truyền qua mạng.

## c. WebSocket

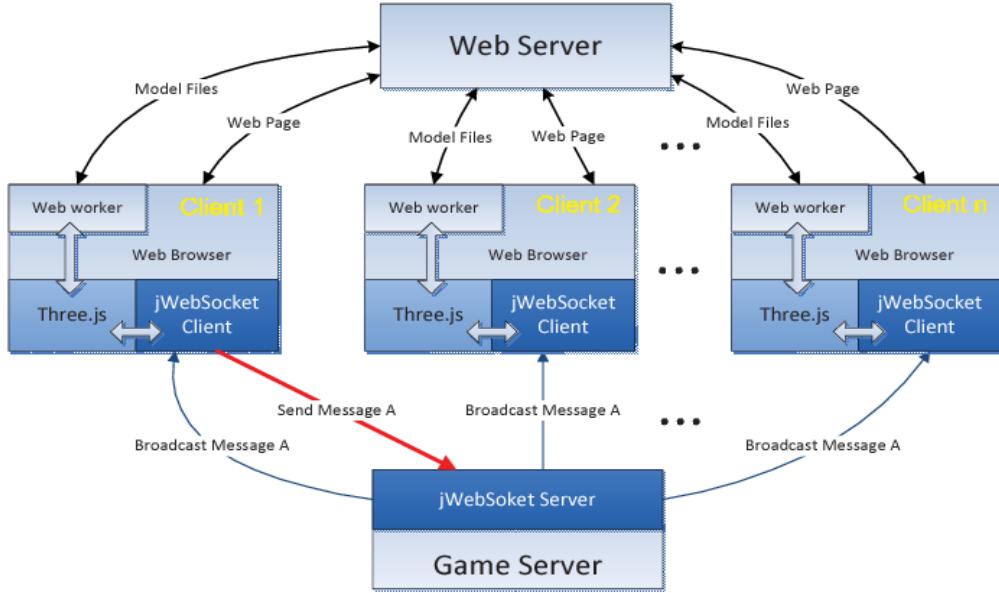
### ❖ Khái niệm

WebSockets là một công nghệ được sử dụng để tạo kết nối hai chiều giữa trình duyệt và máy chủ. Nó cho phép truyền tải dữ liệu thời gian thực trong thời gian thực qua một kết nối duy nhất, đồng thời giảm thiểu độ trễ và số lượng lưu lượng mạng cần thiết để truyền tải dữ liệu.

### ❖ Mục đích và tính năng của giao thức

Cấu trúc của WebSockets trong game multiplayer bao gồm ba thành phần chính: máy chủ WebSocket, trình duyệt, và giao thức WebSocket.

The architecture of our framework is shown in Figure 1.



Hình 1.3. 4: Cấu trúc giao thức WebSockets

Máy chủ WebSocket: Đây là phần của hệ thống game multiplayer nhận và xử lý các yêu cầu của trình duyệt thông qua giao thức WebSocket. Máy chủ này sẽ tạo và duy trì các kết nối WebSocket với trình duyệt của người chơi và xử lý các tương tác của họ trong trò chơi.

Trình duyệt: Trình duyệt sử dụng giao thức WebSocket để thiết lập kết nối với máy chủ WebSocket và truyền tải dữ liệu game qua kết nối đó. Người chơi có thể tương tác với trò chơi thông qua giao diện người dùng của trình duyệt và các tương tác này sẽ được gửi đến máy chủ thông qua kết nối WebSocket.

Giao thức WebSocket: Đây là một giao thức mở cho phép truyền tải dữ liệu theo hai chiều qua một kết nối TCP. Giao thức WebSocket cho phép truyền tải dữ liệu thời gian thực trong trò chơi multiplayer và giảm thiểu độ trễ.

Khi trò chơi được bắt đầu, trình duyệt của người chơi sẽ thiết lập kết nối WebSocket với máy chủ. Sau khi kết nối đã được thiết lập, trình duyệt có thể gửi dữ liệu đến máy chủ thông qua kết nối WebSocket. Máy chủ sẽ xử lý và đáp ứng các yêu cầu từ trình duyệt thông qua kết nối WebSocket.

Tóm lại, cấu trúc của WebSockets trong game multiplayer bao gồm máy chủ WebSocket, trình duyệt và giao thức WebSocket. Kết nối WebSocket cho phép truyền tải dữ liệu thời gian thực trong trò chơi và giảm thiểu độ trễ

### ❖ Packet

Packet của WebSocket trong game multiplayer bao gồm hai thành phần chính: header và payload.

- **Header:** Header của một packet WebSocket bao gồm các thông tin về loại dữ liệu truyền tải, độ dài của dữ liệu, mã hóa, xác thực, ... Header còn chứa các trường khác như địa chỉ nguồn và đích, phân đoạn, cờ, số thứ tự và checksum.
- **Payload:** Payload của một packet WebSocket chứa dữ liệu được truyền tải giữa trình duyệt và máy chủ. Payload có thể là bất kỳ loại dữ liệu nào cần truyền tải trong trò chơi multiplayer, chẳng hạn như các tương tác của người chơi, trạng thái của trò chơi, vị trí của các đối tượng trong trò chơi, và nhiều hơn nữa.

### ❖ Tính bảo mật của giao thức

WebSocket được sử dụng trong nhiều ứng dụng web, bao gồm cả trò chơi đa người chơi (multiplayer games). Tuy nhiên, việc đảm bảo tính bảo mật của WebSocket trong game multiplayer cũng là một vấn đề quan trọng cần được xem xét.

WebSocket có thể được sử dụng để truyền tải dữ liệu nhạy cảm, chẳng hạn như thông tin về tài khoản người dùng và các thông tin cá nhân khác.

## d. HTTP

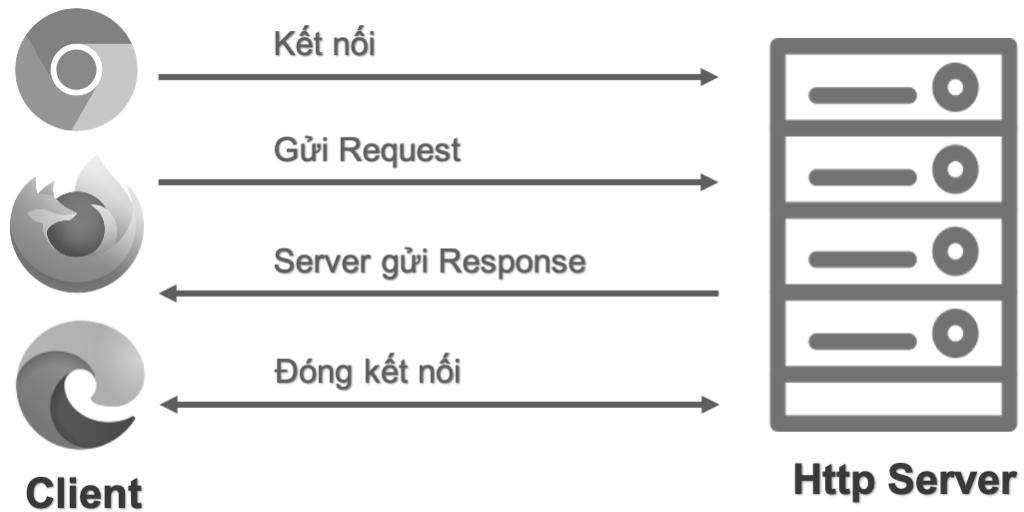
### ❖ Khái niệm

HTTP (Hypertext Transfer Protocol) là một giao thức truyền tải siêu văn bản được sử dụng để truyền tải dữ liệu qua mạng Internet. Tuy nhiên, HTTP

không phải là một giao thức thích hợp cho game multiplayer vì nó thiếu tính năng thời gian thực và độ trễ thấp.

Trong game multiplayer, thời gian đáp ứng (latency) và độ trễ (lag) là hai yếu tố quan trọng để đảm bảo trò chơi được diễn ra một cách mượt mà và thú vị. Việc sử dụng HTTP có thể gây ra độ trễ và thời gian đáp ứng chậm do các yếu tố như đóng gói dữ liệu và xác thực.

#### ❖ Truyền tải client-server trong giao thức HTTP



Hình 1.3. 5: Truyền dữ liệu trong giao thức HTTP

- Bước 1:** Mở kết nối TCP - Kết nối TCP (Giao thức HTTP dựa trên TCP) trên địa chỉ xác định bởi URL (Uniform Resource Locator) và cổng thường là 80, kết nối này được dùng để gửi các yêu cầu (request) và nhận phản hồi (response). Client có thể mở ra kết nối TCP mới hoặc sử dụng kết nối đang có, thậm chí nó tạo ra nhiều kết nối TCP cùng lúc đến server.
- Bước 2:** Gửi HTTP Message (request) - HTTP Message (request) chính là nội dung yêu cầu được client tạo ra và gửi đến server. HTTP Message có nội dung text (plain text) mà chúng ta có thể đọc được (người đọc được). Với phiên bản HTTP/2 thì nội dung HTTP Message được bao

bọc trong các frame, nó làm cho người không đọc được một cách trực tiếp - tuy nhiên về mặt ý nghĩa nội dung không đổi so với HTTP/1.1

- **Bước 3:** Đọc HTTP Message nhận được từ server (response) - Http Message (response) trả về từ server có cấu trúc tương tự Http Message (request)
- **Bước 4:** Đóng kết nối hoặc sử dụng lại cho các truy vấn khác

#### 4. Các đơn vị cung cấp giao thức mạng

##### ❖ Unity Networking

Unity Networking là một phần của Unity Engine cho phép game developer tạo ra các trò chơi multiplayer trên nền tảng desktop, mobile, và console. Unity Networking cung cấp cho developer các công cụ để tạo ra các kết nối giữa các máy tính hoặc thiết bị di động, trao đổi dữ liệu, và đồng bộ hóa trò chơi giữa các máy tính trong một mạng lưới.

Các thành phần chính của Unity Networking bao gồm:

- Network Manager: Là một component có thể được gắn vào GameObject trong trò chơi của bạn, cho phép bạn quản lý các kết nối mạng và phân phối dữ liệu giữa các client và server.
- Networked Objects: Là các đối tượng trong trò chơi của bạn được đồng bộ hóa giữa các client và server. Các networked objects có thể được đồng bộ hóa để đảm bảo rằng tất cả các client đang chơi trò chơi của bạn thấy cùng một trạng thái và hành động.
- Remote Procedure Calls (RPCs): Là một cơ chế để gọi các phương thức từ client đến server hoặc ngược lại. RPCs có thể được sử dụng để gửi các yêu cầu giữa các client và server để cập nhật trạng thái trò chơi, xử lý hành động của người chơi, v.v.
- Network Identity: Là một thành phần cho phép bạn gắn các ID định danh duy nhất cho các đối tượng trong trò chơi của bạn, cho phép các client biết đối tượng nào đang được đồng bộ hóa.

- Network Transform: Là một thành phần cho phép bạn đồng bộ hóa chuyển động và vị trí của các đối tượng giữa các client và server.

## ❖ Photon

Trong lập trình game multiplayer, Photon là một platform được sử dụng để phát triển các game online trực tuyến. Platform này cung cấp cho nhà phát triển các công cụ để kết nối, đồng bộ hóa và quản lý các kết nối mạng trong game, cho phép nhiều người chơi truy cập và tương tác với nhau trên cùng một trò chơi.

Photon hỗ trợ các giao thức mạng như TCP, UDP, WebSockets và cung cấp các tính năng như phân tích dữ liệu, đồng bộ hóa, chống giật, cân bằng tải và độ trễ. Nó cũng có các công cụ quản lý như bảng điều khiển quản lý, giao diện lập trình ứng dụng (API) để sử dụng và một thư viện Unity để tích hợp vào trò chơi.

Các ứng dụng của Photon trong lập trình game bao gồm:

- Điều khiển đồng bộ hóa: Photon giúp đồng bộ hóa trạng thái của game trên nhiều thiết bị và đảm bảo rằng các tín hiệu đến từ người chơi được xử lý một cách hợp lý. Ví dụ, khi một người chơi thực hiện một hành động, Photon đảm bảo rằng hành động đó được đồng bộ hóa và phản ánh trên tất cả các thiết bị khác.
- Kết nối mạng: Photon cung cấp một giao diện lập trình ứng dụng (API) để xử lý các kết nối mạng trong game, cho phép các người chơi kết nối với nhau thông qua các máy chủ hoặc peer-to-peer.
- Quản lý game: Photon cung cấp các công cụ để quản lý các trò chơi trực tuyến, cho phép nhà phát triển quản lý thông tin về người chơi, phòng chơi và các trạng thái khác của game.

Photon được sử dụng rộng rãi trong các trò chơi trực tuyến như Fortnite, Among Us và Minecraft để hỗ trợ việc kết nối và đồng bộ hóa giữa các người chơi trên toàn cầu.

## ❖ Unreal Engine

Unreal Engine là một hệ thống game engine được phát triển bởi Epic Games, và nó cũng cung cấp một số giao thức mạng để phát triển các trò chơi trực tuyến. Các giao thức mạng được hỗ trợ bao gồm:

- Unreal Networking: Đây là một giao thức mạng được xây dựng sẵn trong Unreal Engine và được thiết kế để hỗ trợ các trò chơi đa người chơi trực tuyến. Unreal Networking hỗ trợ các tính năng như cập nhật trạng thái game và đồng bộ hóa các hành động giữa các người chơi.
- Replication: Unreal Engine hỗ trợ replication cho các đối tượng trong game, giúp đồng bộ hóa các trạng thái giữa server và client. Các đối tượng được đồng bộ hóa này có thể bao gồm nhân vật, vật phẩm, vũ khí, v.v.
- RPC (Remote Procedure Call): RPC là một công nghệ mạng cho phép client gửi yêu cầu đến server và server phản hồi lại với kết quả. Unreal Engine hỗ trợ việc triển khai các hàm RPC trên các đối tượng trong game, cho phép client gọi các hàm này trên server.
- Steamworks: Unreal Engine cũng hỗ trợ Steamworks, platform của Valve Corporation để phát triển và quản lý các trò chơi trực tuyến trên Steam. Steamworks cung cấp cho các nhà phát triển một số công cụ và dịch vụ để giúp họ tạo ra các trò chơi trực tuyến.
- Socket.IO: Socket.IO là một thư viện JavaScript cho phép việc truyền thông tin giữa server và client thông qua giao thức WebSockets hoặc Polling. Unreal Engine hỗ trợ việc tích hợp Socket.IO vào các trò chơi trực tuyến.

Các giao thức mạng trong Unreal Engine được sử dụng để tạo ra các trò chơi trực tuyến với các tính năng đa người chơi như chơi đối kháng, chế độ chơi độc đối, chia sẻ thông tin và hỗ trợ cho việc tạo ra các mod và phiên bản tùy chỉnh của trò chơi

### ❖ Mirror

Mirror là một thư viện mạng miễn phí cho Unity, cho phép các nhà phát triển tạo game multiplayer với các tính năng như đồng bộ hóa và giao tiếp mạng. Nó hỗ trợ cả TCP và UDP.

Về cơ bản, mirror hoạt động bằng cách sao chép trạng thái của trò chơi từ máy chủ (server) đến các máy khách (client) thông qua một giao thức mạng đồng bộ hóa. Khi một sự kiện xảy ra trong trò chơi, như người chơi di chuyển hoặc tấn công, thông tin sẽ được gửi đến máy chủ và được xử lý tại đó. Sau đó, máy chủ sẽ gửi trạng thái mới nhất đến các máy khách để cập nhật trên màn hình của họ.

Trong thư viện Mirror, chỉ có một máy tính (máy chủ) tính toán và quản lý trạng thái của trò chơi, các máy tính khác (máy khách) sẽ nhận thông tin về trạng thái mới nhất từ máy chủ. Điều này giúp giảm độ trễ và đảm bảo rằng các máy tính trong mạng đồng bộ với nhau để tạo ra một trò chơi đồng nhất.

Tuy nhiên, Mirror cũng có nhược điểm khi tạo ra một mức độ phụ thuộc lớn vào máy chủ. Nếu máy chủ bị quá tải hoặc có vấn đề về kết nối, trò chơi có thể trở nên chậm hoặc không ổn định. Do đó, các lập trình viên game cần phải cân nhắc kỹ về cách triển khai mirror để đảm bảo hiệu quả và tính ổn định của trò chơi.

### ❖ Steamworks

Steamworks là một platform cung cấp bởi Valve Corporation để phát triển và quản lý các trò chơi trực tuyến trên Steam, một nền tảng phân phối trò chơi kỹ thuật số. Steamworks cung cấp cho các nhà phát triển một số công cụ và dịch vụ để giúp họ tạo ra các trò chơi trực tuyến, bao gồm:

- Quản lý người chơi và tài khoản: Steamworks cung cấp một hệ thống quản lý người chơi và tài khoản, cho phép nhà phát triển tạo và quản lý các tài khoản người chơi trên Steam.
- Hệ thống chơi đối kháng: Steamworks cung cấp một hệ thống chơi đối kháng cho các trò chơi trực tuyến, bao gồm khả năng tìm kiếm trận đấu, kết nối người chơi và hỗ trợ cho chế độ chơi độc đáo.
- Hệ thống cộng đồng: Steamworks cung cấp các công cụ và dịch vụ để tạo ra một cộng đồng trò chơi trực tuyến, bao gồm hệ thống lời nhắn, cửa hàng trong game, hệ thống chia sẻ thông tin và hỗ trợ cho việc tạo ra các mod và phiên bản tùy chỉnh của trò chơi.
- Quản lý DLC và nội dung tải về: Steamworks cung cấp các công cụ và dịch vụ để quản lý các bản cập nhật, DLC và nội dung tải về cho các trò chơi trực tuyến, bao gồm việc phân phối, quản lý và theo dõi các tài nguyên và phiên bản khác nhau của trò chơi.

Steamworks sử dụng một số giao thức mạng như TCP/IP, UDP và WebSockets để kết nối các người chơi với nhau và với các máy chủ trò chơi. Steamworks cũng hỗ trợ việc tạo ra các mạng LAN ảo cho các trò chơi trực tuyến, giúp các người chơi kết nối với nhau trong cùng một mạng LAN dù không cùng một địa chỉ vật lý

## ❖ SmartfoxServer

SmartFoxServer là một platform cung cấp giao thức mạng cho các trò chơi đa người chơi. Nó cung cấp các giao thức mạng như TCP và UDP để

cho các trò chơi đa người chơi có thể kết nối và truyền tải dữ liệu một cách ổn định và đáng tin cậy.

Với SmartFoxServer, các lập trình viên có thể dễ dàng thiết lập các kết nối mạng giữa các người chơi và phát triển các tính năng cho trò chơi như chơi đồng thời, chat, đồng bộ hóa dữ liệu và phân tích dữ liệu. SmartFoxServer hỗ trợ các giao thức mạng như socket, HTTP và WebSocket, cung cấp khả năng kết nối và truyền tải dữ liệu linh hoạt cho các trò chơi đa người chơi trên nhiều nền tảng, bao gồm PC, di động và web.

Ngoài ra, SmartFoxServer cũng cung cấp các tính năng quản lý dữ liệu để giúp đảm bảo tính đồng bộ và ổn định cho các trò chơi đa người chơi. Các tính năng này bao gồm đồng bộ hóa dữ liệu, quản lý tài nguyên, chống giật và bảo mật.

SmartFoxServer có khả năng mở rộng và hiệu suất cao, có thể xử lý hàng ngàn người dùng cùng lúc và hỗ trợ cơ chế phân tán để mở rộng hệ thống.

Tóm lại, SmartFoxServer là một đơn vị cung cấp giao thức mạng ổn định và hiệu quả cho các trò chơi đa người chơi, giúp cho các trò chơi đa người chơi có thể kết nối và truyền tải dữ liệu một cách ổn định và đáng tin cậy trên nhiều nền tảng khác nhau.

## IV. Tìm hiểu về Photon Engine

### 1. Khái niệm và tính năng

Photon Engine là một nền tảng phát triển trò chơi đa người chơi thời gian thực được cung cấp bởi Exit Games, một công ty có trụ sở tại Đức và Mỹ. Nền tảng này cung cấp các công cụ và dịch vụ để phát triển các trò chơi đa người chơi trực tuyến trên nhiều nền tảng, bao gồm web, điện thoại di động và máy tính.

Photon Engine bao gồm ba sản phẩm chính: Photon Cloud, Photon Server và Photon Unity Networking (PUN).

- Photon Cloud là một dịch vụ đám mây cho phép phát triển trò chơi đa người chơi trực tuyến với khả năng mở rộng cao.
- Photon Server là một máy chủ game đa nền tảng để tự lưu trữ các trò chơi đa người chơi của bạn. PUN là một plugin Unity cho phép các nhà phát triển sử dụng Photon Engine để phát triển các trò chơi đa người chơi trực tuyến trên Unity.

Với Photon Engine, nhà phát triển có thể tập trung vào việc phát triển nội dung và trải nghiệm người chơi, trong khi các vấn đề liên quan đến kết nối, đồng bộ hóa dữ liệu và khả năng mở rộng được giải quyết bởi nền tảng này. Nền tảng này đã được sử dụng để phát triển nhiều trò chơi đa người chơi trực tuyến phổ biến trên toàn thế giới.

## 2. Ứng dụng của Photon Engine

Photon Engine được ứng dụng rộng rãi trong nhiều lĩnh vực hiện nay

- Game đa người chơi trực tuyến: Photon Engine là một nền tảng phát triển game đa người chơi trực tuyến với khả năng mở rộng cao. Nền tảng này cung cấp các công cụ để xây dựng các trò chơi đa người chơi trực tuyến như game bắn súng, game thể thao, game chiến thuật, v.v.
- Thể thao điện tử: Photon Engine cũng được sử dụng để phát triển các trò chơi trực tuyến đa người chơi cho các giải đấu thể thao điện tử. Với khả năng mở rộng cao và độ ổn định cao, Photon Engine là một lựa chọn phù hợp cho các nhà phát triển muốn phát triển các trò chơi thể thao điện tử.
- Hội thảo trực tuyến và trò chuyện video: Photon Engine có thể được sử dụng để phát triển các ứng dụng hội thảo trực tuyến và trò chuyện video với nhiều người tham gia. Các ứng dụng như Skype, Zoom hay Microsoft Teams đều sử dụng các công nghệ tương tự như Photon Engine để đảm bảo độ trễ thấp và khả năng mở rộng.

- Thực tế ảo và thực tế tăng cường: Photon Engine cũng có thể được sử dụng để phát triển các ứng dụng thực tế ảo và thực tế tăng cường, cho phép người dùng tương tác với nhau trong môi trường ảo.
- Trong giáo dục và đào tạo: Photon Engine cũng có thể được sử dụng để phát triển các trò chơi giáo dục và đào tạo đa người chơi trực tuyến. Các ứng dụng như Edmodo và Kahoot! đều sử dụng các công nghệ tương tự như Photon Engine để đảm bảo tính tương tác và độ trễ thấp khi sử dụng trên các thiết bị khác nhau.

### **3. Ưu nhược điểm của photon engine**

#### **❖ Ưu điểm:**

- Thời gian phản hồi nhanh: Photon Engine có khả năng xử lý một lượng lớn người dùng cùng lúc, giúp giảm thiểu thời gian chờ đợi và tăng trải nghiệm người dùng.
- Hỗ trợ đa nền tảng tốt: Photon Engine hỗ trợ đa nền tảng tốt hơn Smartfox, bao gồm Windows, MacOS, iOS, Android, Xbox, PlayStation, Nintendo Switch và WebGL.
- Hỗ trợ nhiều ngôn ngữ lập trình: Photon Engine hỗ trợ nhiều ngôn ngữ lập trình như C++, C#, Java, JavaScript, Objective-C và Swift.

#### **❖ Nhược điểm:**

- Giới hạn tính năng: Photon Engine không có nhiều tính năng như Smartfox, điều này có thể gây khó khăn cho các game phát triển khi muốn tùy chỉnh các tính năng đặc biệt.
- Giá cả tăng khi scale: Khi tăng quy mô, giá cả của Photon Engine sẽ tăng lên, do đó nó có thể không phù hợp cho các tự doanh nhỏ và các nhà phát triển game indie.

### **4. Các ngôn ngữ lập trình và nền tảng hỗ trợ**

Photon Engine hỗ trợ nhiều ngôn ngữ lập trình và nền tảng khác nhau để phát triển các trò chơi đa người chơi trực tuyến.

- Ngôn ngữ lập trình: C# (.NET), C++, JavaScript, Objective-C, Swift, Java, Python.
- Nền tảng di động: iOS, Android, Windows Phone.
- Nền tảng máy tính: Windows, MacOS, Linux.
- Nền tảng trình duyệt: HTML5, WebGL, Unity Web Player.
- Nền tảng game engine: Unity, Unreal Engine, cocos2d-x.

Các ngôn ngữ lập trình và nền tảng này cho phép nhà phát triển tùy chỉnh và phát triển các trò chơi đa người chơi trực tuyến trên nhiều nền tảng khác nhau với khả năng mở rộng cao và độ trễ thấp.

## 5. Photon Unity Networking

### a. Tổng quan về Photon Unity NetWorking

Photon Unity Networking (PUN) là một plugin mạng được phát triển bởi Exit Games dành cho trò chơi được phát triển trên nền tảng Unity. PUN cho phép các nhà phát triển tạo ra các trò chơi đa người chơi trực tuyến trên nhiều nền tảng khác nhau, bao gồm PC, điện thoại di động và các thiết bị khác.

PUN sử dụng giao thức UDP để truyền tải dữ liệu trò chơi, đồng bộ hóa các tương tác giữa các người chơi và các đối tượng trên màn hình. Nó cũng hỗ trợ các tính năng như phân phối gói tin, xử lý độ trễ, xác thực người dùng và quản lý phòng chơi.

PUN cung cấp các API đơn giản để các nhà phát triển trò chơi có thể thêm tính năng mạng vào trò chơi của họ. Nó cũng cung cấp các tính năng như phát hiện mạng tự động, tự động tái kết nối và chống giật để giảm thiểu tác động của lỗi mạng đến trò chơi.

**Cấu trúc:** Thông thường PUN gồm 3 lớp

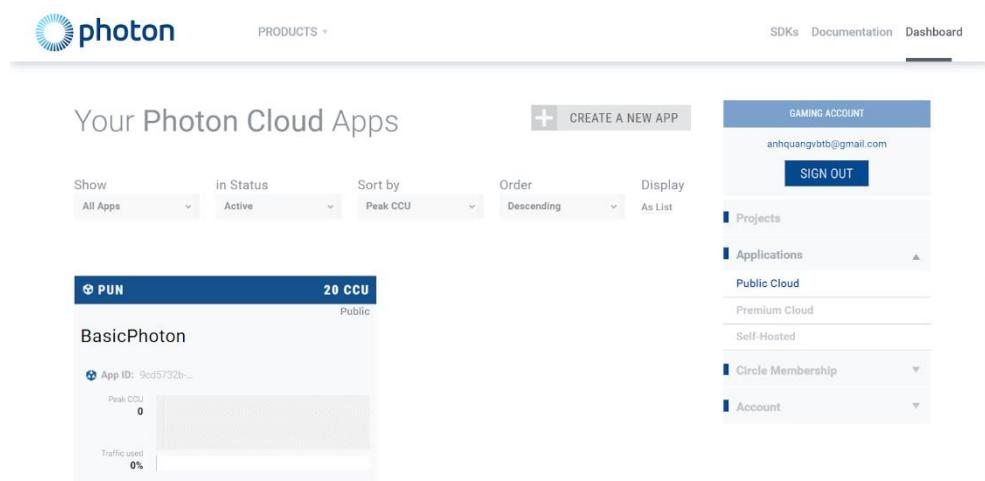
- Cấp độ cao nhất là mã PUN, thực hiện các tính năng dành riêng cho Unity như các đối tượng được nối mạng, RPC, v.v.

- Cấp độ thứ hai chứa logic để hoạt động với các máy chủ Photon, thực hiện mai mối, gọi lại, v.v. Đây là API thời gian thực. Điều này có thể được sử dụng riêng của nó rồi. Bạn sẽ nhận thấy rất nhiều chủ đề chồng chéo giữa PUN và API thời gian thực (còn gọi là API LoadBalancing) nhưng điều đó không sao cả.
- Mức thấp nhất được tạo thành từ các tệp DLL, có chứa de/serialization, các giao thức, v.v

## b. Cài đặt

Để sử dụng Photon Unity NetWorking để phát triển các trò chơi đa người chơi trực tuyến, cần thực hiện các bước sau:

**Bước 1: Đăng ký tài khoản:** Truy cập trang web của Photon Engine và đăng ký tài khoản để có thể sử dụng dịch vụ.



Hình 1.4. 1. Photon Dashboard

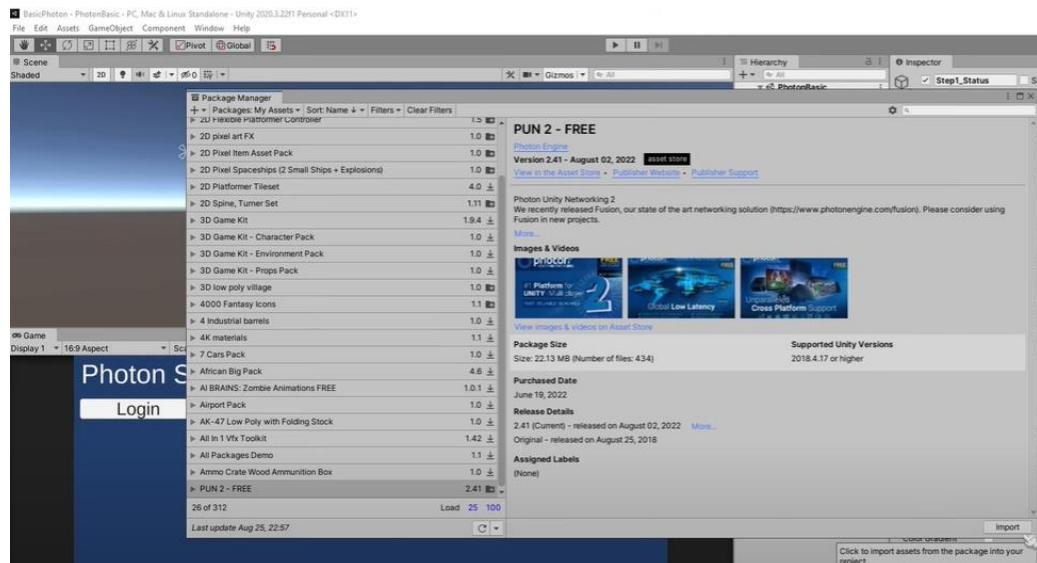
**Bước 2: Tạo ứng dụng:** Sau khi đăng nhập vào tài khoản, ta cần tạo một ứng dụng trên trang web của Photon Engine. Ứng dụng này sẽ được sử dụng để kết nối các người chơi lại với nhau.

**Bước 3: Tải SDK:** Photon Engine cung cấp các SDK cho các ngôn ngữ lập trình khác nhau. Ta cần tải SDK phù hợp với ngôn ngữ lập trình bạn đang sử dụng. Vd: Unity



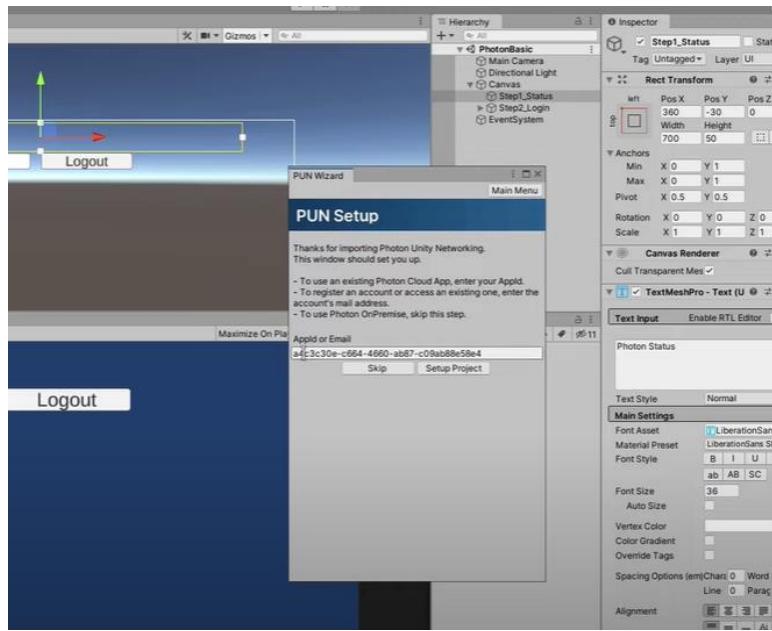
*Hình 1.4. 2. Add PUN 2*

**Bước 4: Thêm SDK vào dự án:** Sau khi tải SDK, ta cần thêm các thư viện của SDK vào dự án của mình.



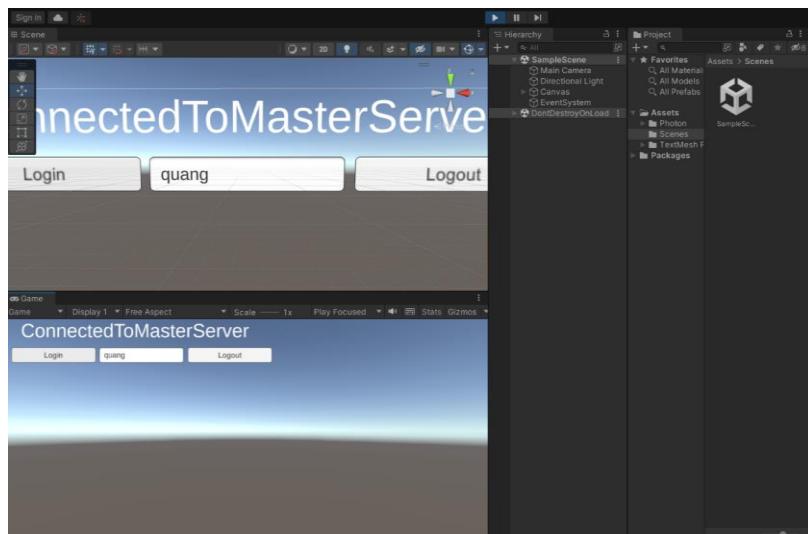
*Hình 1.4. 3. Import PUN 2*

**Bước 5: Kết nối với Photon Server:** Ta cần kết nối ứng dụng của mình với Photon Server để có thể gửi và nhận dữ liệu. Các thông tin kết nối này được cung cấp khi tạo ứng dụng.



Hình 1.4. 4. Paste AppId đã copy vào để setup

**Bước 6: Xây dựng trò chơi:** Sau khi kết nối thành công, có thể bắt đầu xây dựng trò chơi của mình. Các chức năng như xử lý đăng nhập, kết nối, gửi và nhận dữ liệu từ các người chơi khác đều được cung cấp bởi SDK của Photon Engine.



Hình 1.4. 5. Kiểm tra xem đã kết nối

### c. Photon cloud

Về cơ bản thì Photon cloud là 1 loạt các máy chủ photon(photon server) chạy trên đó. Photon cloud có thể miễn phí hoặc cần phải trả phí nếu chúng ta muốn có nhiều hơn CCU(số lượng người dùng đồng thời trong server).

Đây là những gì sẽ diễn ra bên trong Photon cloud

- Mọi người kết nối với "Máy chủ định danh" trước. Nó kiểm tra ứng dụng nào (với AppId) và khu vực mà khách hàng muốn sử dụng. Sau đó, nó chuyển tiếp máy khách đến Máy chủ chính.
- Máy chủ chính là trung tâm cho một loạt các máy chủ khu vực. Nó biết tất cả các phòng cho khu vực này. Bất cứ khi nào một phòng (trận đấu / trò chơi) được tạo hoặc tham gia, máy khách sẽ được chuyển tiếp đến một trong các máy khác, được gọi là "Game Server".

Khi người chơi kết nối cùng 1 máy chủ thì chỉ người chơi nào có cùng AppId sẽ gặp được nhau. AppId ở đây chính là 1 đoạn mã được cấp cho ta khi tạo 1 app trên photon.

Room (phòng) là nơi những người chơi chơi game với nhau trong một phòng, mọi người đều nhận được bất cứ thứ gì người khác gửi (trừ khi bạn gửi tin nhắn cho những người chơi cụ thể)

Lobby (sảnh) tồn tại trên máy chủ để liệt kê các phòng trong trò chơi. Nó không cho phép người chơi giao tiếp với nhau.

#### d. Kết nối với Photon Cloud và tham gia room

```
public void Connect()
{
    // we check if we are connected or not, we join if we are , else we initiate the connection
    if (PhotonNetwork.IsConnected)
    {
        // #Critical we need at this point to attempt joining a Random Room. If it fails,
        PhotonNetwork.JoinRandomRoom();
    }
    else
    {
        // #Critical, we must first and foremost connect to Photon Online Server.
        PhotonNetwork.ConnectUsingSettings();
        PhotonNetwork.GameVersion = gameVersion;
    }
}
```

Hình 1.4. 6. Tham gia phòng trong Photon

Để kết nối đầu tiên chúng ta sẽ check xem đã được kết nối chưa bằng Photon.isConnected. Nếu đã được kết nối thì ta sẽ tiến hành tham gia vào 1

phòng nào đó bắt kỳ bằng PhotonNetwork.JoinRandomRoom(), ngược lại thì sẽ tiến hành bắt đầu kết nối bằng PhotonNetwork.ConnectUsingSettings()

Và PUN cũng cung cấp cho chúng ta một số hàm callback, để sử dụng chúng ta cần implement chúng:

- **IConnectionCallbacks:** liên quan đến kết nối.
- **IN RoomCallbacks:** sự kiện xảy ra trong phòng.
- **ILobbyCallbacks:** sự kiện liên quan đến tiền sảnh.
- **IMatchmakingCallbacks:** sự kiện liên quan đến việc tìm kiếm và kết nối với các phòng trên Photon Server
- **IONEventCallback:** cho bất kỳ sự kiện nào đã nhận.  
**IWebRpcCallback:** nhận phản hồi hoạt động của WebRPC.
- **IPunInstantiateMagicCallback:** sự kiện liên quan đến các tiền tố PUN được khởi tạo.
- **IPunObservable:** liên quan đến tuần tự hóa PhotonView.
- **IPunOwnershipCallbacks:** chuyển quyền sở hữu PUN.
- Hoặc thay vì phải implement chúng ta có thể sử dụng **MonoBehaviourPunCallbacks** thay vì **MonoBehaviour**

### e. Photon View

PhotonView là một thành phần trong Photon Unity Networking (PUN) của Photon Engine, được sử dụng để đồng bộ hóa các đối tượng trên mạng trong các trò chơi đa người chơi. Mỗi đối tượng trên mạng cần phải có một PhotonView để có thể được đồng bộ hóa và truyền tải giữa các máy khác nhau trong một phòng chơi (room).

Một PhotonView chứa thông tin về đối tượng trên mạng, bao gồm ID của nó, cách thức đồng bộ hóa (sử dụng các phương thức Serialize hoặc SerializeStream của PUN), và quyền sở hữu của nó (xác định đối tượng do máy khách hay máy chủ điều khiển). Nếu một đối tượng được đánh dấu với

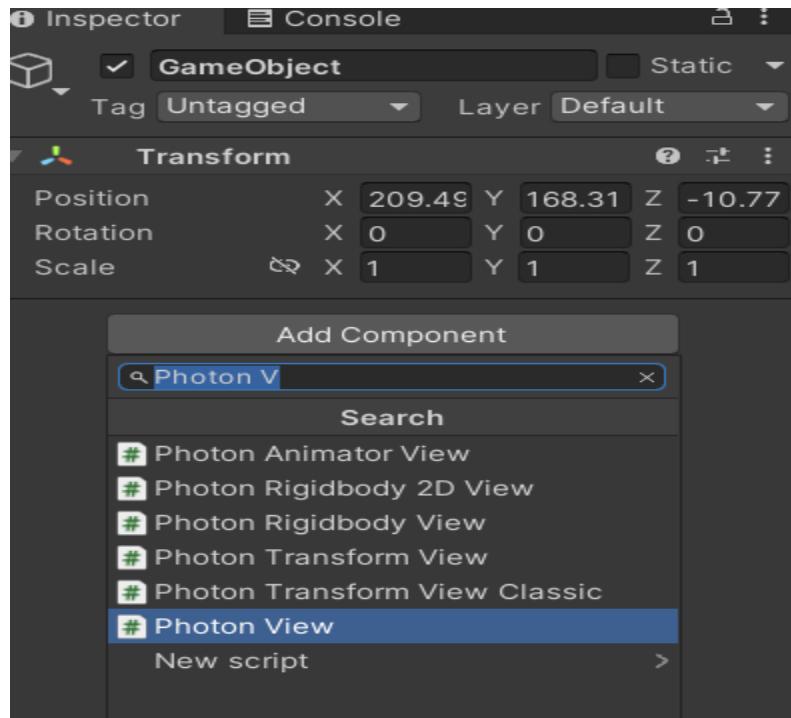
một PhotonView, PUN sẽ tự động xử lý việc đồng bộ hóa và truyền tải giữa các máy trong cùng một phòng chơi.

Một đối tượng có thể có nhiều PhotonView, ví dụ như một nhân vật trong trò chơi có thể có một PhotonView cho hình dạng của nhân vật và một PhotonView khác cho việc di chuyển của nhân vật. Khi một đối tượng được tạo trên một máy, PUN sẽ tự động tạo một PhotonView tương ứng và thêm vào đối tượng đó.

Việc sử dụng PhotonView trong PUN là cực kỳ quan trọng để đảm bảo rằng các đối tượng trong trò chơi được đồng bộ hóa đúng cách và truyền tải đúng dữ liệu giữa các máy trong cùng một phòng chơi.

PhotonView Creator là một tính năng trong Photon Unity Networking (PUN) cho phép tạo và quản lý PhotonView trên mạng trong Unity Editor. Điều này giúp quản lý các PhotonView một cách dễ dàng hơn và giảm thiểu lỗi đồng bộ hóa trên mạng. Khi sử dụng tính năng này, có thể tạo các PhotonView mới cho đối tượng trong game, hoặc chỉnh sửa các thuộc tính của PhotonView đang tồn tại. Tính năng này cũng cho phép đồng bộ hóa các thuộc tính của PhotonView giữa các máy khác nhau trong phòng chơi.

Để tạo PhotonView thì chỉ cần khi tạo ra một gameobject chúng ta add cho nó 1 component là PhotoView là được



Hình 1.4. 7. Add component Photon View cho đối tượng

Chúng ta cũng có thể tạo nó bằng code:

```

using Photon.Pun;
using UnityEngine;

public class MyScript : MonoBehaviourPunCallbacks
{
    void Start()
    {
        // Tạo một PhotonView với ViewID = 1 và tùy chọn "Ownership Transfer" cho đối tượng
        PhotonView photonView = PhotonView.Get(1);
        photonView.TransferOwnership(PhotonNetwork.LocalPlayer);

        // Gán PhotonView cho GameObject
        gameObject.AddComponent<PhotonView>();
        gameObject.GetComponent<PhotonView>().ViewID = photonView.ViewID;
    }
}

```

Hình 1.4. 8. Gán Photon View bằng code

PhotonView Owner (chủ sở hữu Photon View): Trong PUN, quyền sở hữu của đối tượng được quản lý bởi component PhotonView. Một khi một người chơi có quyền sở hữu đối tượng, họ có thể gọi phương thức TransferOwnership() của PhotonView để chuyển quyền sở hữu sang người

chơi khác. Người chơi mới sẽ có quyền kiểm soát đối tượng đó và có thể thay đổi trạng thái của đối tượng đó.

PhotonView Controller: Thường chủ sở hữu sẽ là người có quyền kiểm soát trừ trường hợp chủ sở hữu không tồn tại hoặc bị mất kết nối mạng.

## f. Đồng bộ hóa

Trong PUN việc xử lý đồng bộ hóa rất dễ dàng chúng ta chỉ cần tạo 1 PhotonView cho một đối tượng là có thể đồng bộ hóa vị trí, hướng, các thông số khác với các bản sao ở các máy khách khác.

Có 4 tùy chọn để đồng bộ hóa dữ liệu:

- Off: Đồng bộ hóa không xảy ra, không có gì được gửi hoặc nhận
- Reliable Delta Compressed: Dữ liệu được đảm bảo nhận được với cơ chế tối ưu hóa nội bộ sẽ gửi null nếu dữ liệu không thay đổi.
- Unreliable: Dữ liệu được nhận theo thứ tự nhưng một số cập nhật có thể bị mất.
- Unreliable OnChange: Dữ liệu được nhận theo thứ tự nhưng một số cập nhật có thể bị mất. Nếu các bản cập nhật lặp lại thông tin cuối cùng, PhotonView sẽ tạm dừng gửi các bản cập nhật cho đến khi có thay đổi tiếp theo.

**Đồng bộ hóa thuộc tính:** Điều này được thực hiện 'qua máy chủ' theo mặc định, nghĩa là:

Theo mặc định, việc đặt thuộc tính cho nhân vật cho thuộc tính của nhân vật hoặc phòng sẽ không có hiệu lực ngay lập tức đối với ứng dụng khách của người gửi/người thiết lập (tác nhân đặt thuộc tính) khi được tham gia vào một phòng trực tuyến. Thay vào đó, máy khách người gửi/thiết lập (tác nhân đặt thuộc tính) sẽ đợi sự kiện máy chủ PropertyChanged áp dụng/đặt thay đổi cục bộ. Vì vậy, cần đợi cho đến khi lệnh gọi lại OnPlayerPropertiesUpdate hoặc OnRoom Properties Update được kích hoạt cho máy khách cục bộ để truy cập chúng. Lý do

đằng sau điều này là các thuộc tính có thể dễ dàng không đồng bộ hóa nếu chúng ta đặt chúng cục bộ trước rồi gửi yêu cầu thực hiện điều đó trên máy chủ và cho các tác nhân khác trong phòng. Cái sau có thể không thành công và có thể kết thúc với các thuộc tính của máy khách người gửi/người thiết lập (tác nhân đặt các thuộc tính) khác cục bộ với những gì trên máy chủ hoặc trên các máy khách khác. Nếu muốn có hành vi cũ (đặt thuộc tính cục bộ trước khi gửi yêu cầu đến máy chủ để đồng bộ hóa chúng), hãy đặt roomOptions.BroadcastPropsChangeToAll thành false trước khi tạo phòng. Nhưng ta không nên làm điều này.

- Máy khách vẫn có thể lưu trữ các thuộc tính của trình phát cục bộ bên ngoài phòng. Những tài sản đó sẽ được gửi khi vào phòng. Ngoài ra, việc đặt thuộc tính ở chế độ ngoại tuyến diễn ra ngay lập tức.
- Ngoài ra, theo mặc định, các thuộc tính của nhân vật cục bộ không bị xóa giữa các phòng

### **g. Cơ chế chống hack game**

Thông thường, Photon đề xuất phương án: Yêu cầu tất cả người chơi tìm và kiểm tra gian lận trong game và báo cáo những gian lận đó. Cấm/chặn người dùng khỏi trò chơi, những người đã bị báo cáo quá nhiều lần (và cả những người đã báo cáo tin tức quá nhiều lần).

Ngoài ra nhà phát triển có thể tìm kiếm những tool chống gian lận từ Asset store.

Ngoài ra có thể sử dụng Photon Bolt vì Bolt có một số tính năng tích hợp để giảm khả năng gian lận về tổng thể, ngay cả khi không có máy chủ. Nó xác định trạng thái và tất cả các thông báo, điều này làm cho thông tin tùy ý có thể được đưa vào ít hơn.

## V. Tiêu kết chương 1

Tóm lại để có thể nghiên cứu và làm ra được một game đa người chơi chúng ta cần phải nắm vững và tìm hiểu kỹ rất nhiều kiến thức như kiến thức về multiplayer game, game design document và các phương thức kết nối mạng. Từ những kiến thức đó giúp chúng ta bước đầu vạch ra được những kế hoạch tiếp theo trong phát triển game.

## CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG

### I. Mô tả bài toán

Trong game Spaceship, người chơi sẽ được đảm nhận vai trò của phi công đang điều khiển một chiếc phi thuyền trong không gian. Mục tiêu chính của người chơi là bắn phá và tiêu diệt các thiên thạch bay đến từ khắp nơi để kiếm điểm số càng cao càng tốt.

Người chơi sẽ đối mặt với các thiên thạch có kích thước và tốc độ di chuyển khác nhau. Họ sẽ sử dụng các phím hoặc bộ điều khiển để điều khiển phi thuyền di chuyển trong không gian và nhắm bắn vào các thiên thạch. Kỹ năng và phản ứng nhanh của người chơi sẽ được thử thách khi cố gắng tránh va chạm với các thiên thạch và đồng thời bắn chúng.

Khi người chơi bắn trúng một thiên thạch, nó sẽ phá hủy thành mảnh nhỏ và người chơi sẽ nhận được điểm số tương ứng. Điểm số có thể được tính dựa trên kích thước, tốc độ hoặc loại thiên thạch bị tiêu diệt. Mục tiêu của người chơi là cố gắng bắn hạ càng nhiều thiên thạch càng tốt để tích lũy điểm số cao nhất.

Tuy nhiên, nếu phi thuyền va chạm vào bất kỳ thiên thạch nào, người chơi sẽ mất một lượt chơi. Người chơi có tối đa 3 lượt chơi để cố gắng đạt được số điểm cao nhất có thể. Khi mất hết số lượt chơi, trò chơi kết thúc và điểm số cuối cùng được tính toán.

Trong quá trình chơi, điểm số của người chơi sẽ được hiển thị trên bảng xếp hạng, cho phép người chơi so sánh thành tích của mình với những người chơi khác. Điều này tạo thêm tính cạnh tranh và thúc đẩy người chơi cố gắng đạt được kết quả tốt nhất.

Tóm lại, game spaceship yêu cầu người chơi phải có khả năng điều khiển phi thuyền, bắn phá và tiêu diệt các thiên thạch để kiếm điểm số cao. Kỹ năng, phản ứng nhanh và khả năng tránh va chạm là những yếu tố quan trọng trong việc vượt qua các thử thách trong game này.

## II. Kiến trúc hệ thống

Game SpaceShip được xây dựng dựa trên mô hình client-server, có hai phần chính: máy chủ (server) và máy khách (client). Dưới đây là mô tả chi tiết về mô hình này:

❖ Máy chủ (Server):

- Máy chủ đóng vai trò là trung tâm điều khiển của trò chơi. Nó có nhiệm vụ quản lý và xử lý các sự kiện của game
- Nó xử lý các yêu cầu và lệnh từ các máy khách và cập nhật trạng thái game dựa trên hành động của người chơi.
- Máy chủ quản lý các phòng trong trò chơi, nơi người chơi có thể tham gia và tương tác với nhau. Khi một người chơi muốn tạo phòng mới, máy chủ tạo một phòng mới với một ID duy nhất và gán người chơi làm chủ phòng. Người chơi khác có thể tìm và tham gia vào các phòng đã được tạo sẵn.
- Máy chủ cũng có nhiệm vụ xác thực và kiểm soát dữ liệu của người chơi, như thông tin tài khoản và điểm số.

❖ Máy khách (Client):

- Máy khách là phần mềm chạy trên thiết bị của người chơi, cho phép họ tham gia vào trò chơi Spaceship.
- Máy khách gửi yêu cầu và lệnh đến máy chủ để tương tác và tham gia vào trò chơi.
- Nó nhận dữ liệu từ máy chủ, bao gồm trạng thái game và các cập nhật về các tàu vũ trụ và thiên thạch.
- Máy khách hiển thị thông tin game cho người chơi, bao gồm đồ họa, âm thanh và giao diện người dùng.

❖ Giao tiếp:

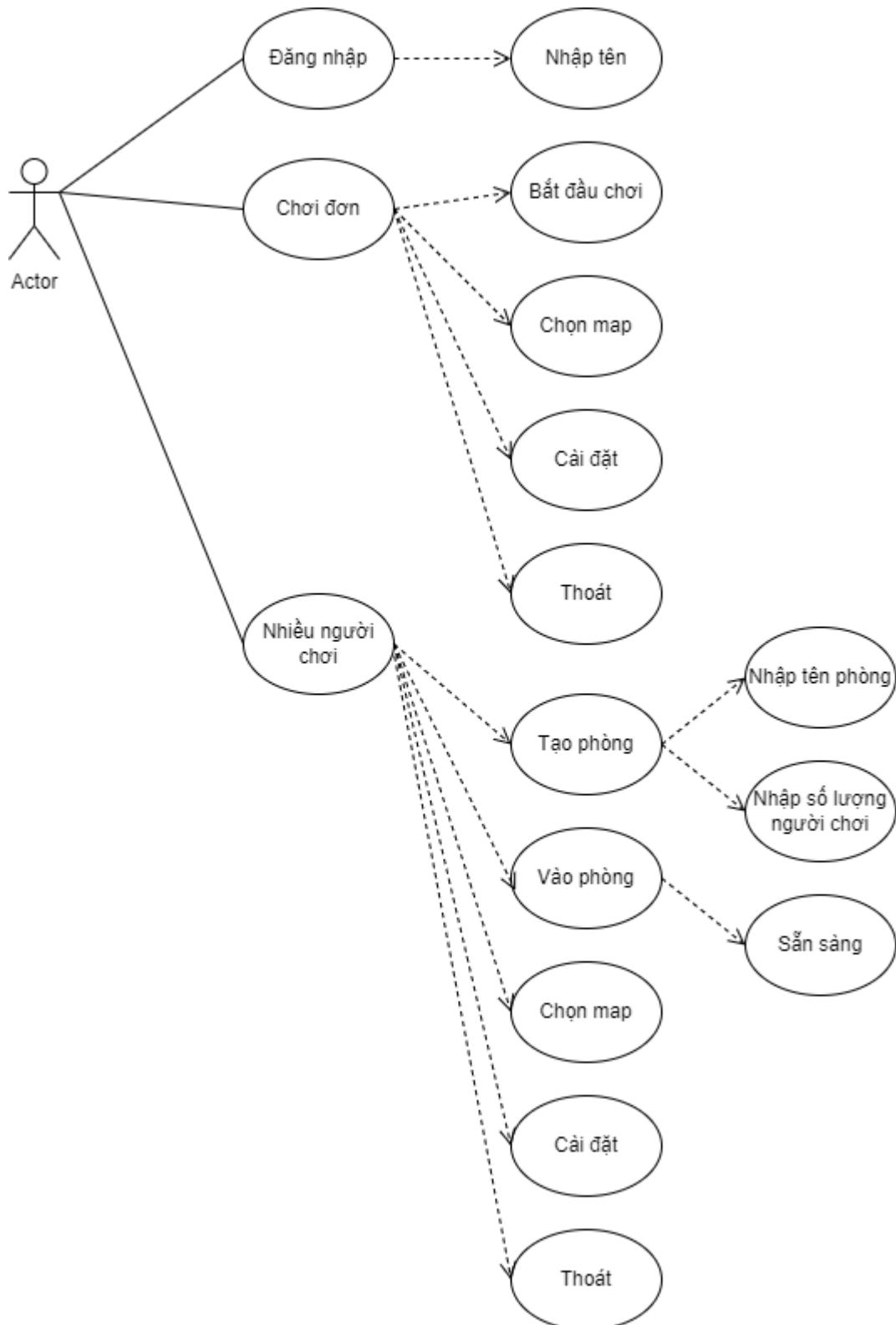
- Máy khách và máy chủ giao tiếp thông qua mạng, thường là mạng Internet.

- Máy khách gửi yêu cầu và lệnh đến máy chủ thông qua giao thức mạng, như TCP/IP hoặc UDP.
- Máy chủ xử lý yêu cầu từ máy khách, cập nhật trạng thái game và gửi lại dữ liệu phản hồi.
- Máy khách nhận dữ liệu từ máy chủ và cập nhật hiển thị trên giao diện người dùng của trò chơi Spaceship.

Mô hình Client-Server trong game Spaceship cho phép nhiều người chơi kết nối và tương tác với nhau thông qua máy chủ chung. Máy chủ là trung tâm điều khiển và cung cấp sự đồng bộ hóa giữa các máy khách, đảm bảo mọi người chơi có trải nghiệm chung và phản hồi nhanh nhạy trong trò chơi.

### III. Phân tích thiết kế hệ thống

#### 1. Use case



Hình 2.3. 1. Use case hệ thống game SpaceShip

## 2. Scenario

- Đăng nhập vào game

Tên	Đăng nhập vào game
Tác nhân	Người chơi
Mục tiêu	Đăng nhập vào game
Yêu cầu liên quan	Không
Tiền điều kiện	Không
Mô tả	1. Người chơi click vào button “Đăng nhập”
Hậu điều kiện	Người chơi được đưa đến scene chọn chế độ chơi
Biến thể	Không
Ngoại lệ	Không

Bảng 2.3. 1: Kịch bản chức năng đăng nhập game

- Chọn chế độ chơi

Tên	Chọn chế độ chơi
Tác nhân	Người chơi
Mục tiêu	Chọn chế độ chơi game
Yêu cầu liên quan	Đăng nhập thành công
Tiền điều kiện	Đăng nhập
Mô tả	1. Người chơi chọn 1 trong 2 chế độ “chơi đơn” hoặc “nhiều người chơi”
Hậu điều kiện	Người chơi được đưa đến lobby của chế độ tương ứng
Biến thể	Không
Ngoại lệ	Không

Bảng 2.3. 2: Kịch bản chức năng chọn chế độ chơi

- Chơi với chế độ chơi đơn

Tên	Chơi với chế độ chơi đơn
Tác nhân	Người chơi
Mục tiêu	Chơi ở chế độ chơi đơn
Yêu cầu liên quan	Đang trong lobby chơi đơn
Tiền điều kiện	Chọn chế độ chơi đơn
Mô tả	<ol style="list-style-type: none"> <li>1. Người chơi click vào button chơi trong lobby</li> <li>2. Trò chơi bắt đầu</li> </ol>
Hậu điều kiện	Người dùng được đưa vào scene chơi game
Biến thể	Không
Ngoại lệ	Không

Bảng 2.3. 3: Kịch bản chức năng chọn chế độ chơi đơn

- Tạo phòng (trong chế độ nhiều người chơi)

Tên	Tạo phòng
Tác nhân	Người chơi
Mục tiêu	Tạo phòng
Yêu cầu liên quan	Trong lobby của chế độ nhiều người chơi
Tiền điều kiện	Chọn chế độ nhiều người chơi
Mô tả	<ol style="list-style-type: none"> <li>1. Người chơi click vào button “Tạo phòng” trong cửa sổ lobby</li> <li>2. Người chơi được đưa đến scene tạo phòng</li> <li>3. Người chơi cần nhập tên phòng và số lượng người chơi trong phòng</li> <li>4. Người chơi click button “Tạo” để tiến hành tạo phòng</li> <li>5. Sau khi tạo thành công người chơi sẽ được đưa đến phòng</li> </ol>

	chờ
Hậu điều kiện	Người chơi được đưa vào phòng chờ
Biến thể	Không
Ngoại lệ	Không

Bảng 2.3. 4: Kịch bản chức năng tạo phòng

- Vào phòng ngẫu nhiên

Tên	Vào phòng ngẫu nhiên
Tác nhân	Người chơi
Mục tiêu	Vào phòng
Yêu cầu liên quan	Trong lobby của chế độ nhiều người chơi
Tiền điều kiện	Không
Mô tả	<ol style="list-style-type: none"> <li>Người dùng click vào button “Vào phòng”</li> <li>Người chơi sẽ được đưa vào phòng chờ của 1 phòng đã được tạo.</li> </ol>
Hậu điều kiện	Người chơi có thẻ sẵn sàng tham gia game
Biến thể	Không
Ngoại lệ	Nếu không có phòng, thì người chơi sẽ là chủ của một phòng được tạo ra

Bảng 2.3. 5: Kịch bản chức năng vào phòng ngẫu nhiên

- Chọn phòng và tham gia phòng (trong chế độ nhiều người chơi)

Tên	Chọn phòng
Tác nhân	Người chơi
Mục tiêu	Chọn phòng

Yêu cầu liên quan	Trong lobby của chế độ nhiều người chơi
Tiền điều kiện	Không
Mô tả	<ol style="list-style-type: none"> <li>Người chơi click vào button “Danh sách phòng”</li> <li>Một list các danh sách phòng xuất hiện</li> <li>Nếu muốn tham gia phòng nào người chơi chỉ cần click “Tham gia” để tham gia phòng đó</li> </ol>
Hậu điều kiện	Người chơi tham gia phòng
Biến thể	Không
Ngoại lệ	Nếu phòng đã đầy thì người chơi không tham gia được

Bảng 2.3. 6: Kịch bản chức năng chọn phòng và tham gia phòng

- Bắt đầu game (trong chế độ nhiều người chơi)

Tên	Bắt đầu game
Tác nhân	Người chơi
Mục tiêu	Bắt đầu game
Yêu cầu liên quan	Các người chơi trong phòng chờ đã sẵn sàng
Tiền điều kiện	Tham gia phòng
Mô tả	<ol style="list-style-type: none"> <li>Người chơi chủ phòng click vào button “Bắt đầu” trong scene</li> <li>Trò chơi bắt đầu</li> </ol>
Hậu điều kiện	Trò chơi bắt đầu
Biến thể	Không
Ngoại lệ	Khi những người chơi khác chưa sẵn sàng thì chủ phòng không thể nhấn click “Bắt đầu”

Bảng 2.3. 7: Kịch bản chức năng bắt đầu chơi game

- Phá hủy thiên thạch

Tên	Phá hủy thiên thạch
Tác nhân	Người chơi
Mục tiêu	Phá hủy thiên thạch
Yêu cầu liên quan	Không
Tiền điều kiện	Trò chơi được bắt đầu
Mô tả	<ol style="list-style-type: none"> <li>Người chơi di chuyển phi thuyền hướng đến thiên thạch và nhả dấu cách.</li> <li>Thiên thạch sẽ bị vỡ ra 3 phần nhỏ, người chơi cần tiếp tục phá hủy tiếp</li> </ol>
Hậu điều kiện	Người chơi sẽ được cộng điểm
Biến thể	Không
Ngoại lệ	Không

Bảng 2.3. 8: Kịch bản chức năng bắn phá thiên thạch

- Phi thuyền va chạm với thiên thạch

Tên	Phi thuyền va chạm với thiên thạch
Tác nhân	Người chơi
Mục tiêu	Không
Yêu cầu liên quan	Không
Tiền điều kiện	Trò chơi được bắt đầu
Mô tả	<ol style="list-style-type: none"> <li>Khi điều khiển phi thuyền người chơi vô tình để va chạm với các viên thiên thạch</li> </ol>
Hậu điều kiện	Số mạng sẽ bị giảm -1
Biến thể	Không
Ngoại lệ	Không

Bảng 2.3. 9: Kịch bản chức năng phi thuyền va chạm với thiên thạch

- Kết thúc game

Tên	Kết thúc game
Tác nhân	Người chơi
Mục tiêu	Trò chơi kết thúc
Yêu cầu liên quan	Không
Tiền điều kiện	Không
Mô tả	<ol style="list-style-type: none"> <li>1. Tất cả người chơi đều chết</li> <li>2. Trò chơi kết thúc</li> </ol>
Hậu điều kiện	Thông báo điểm và đưa người chơi thoát ra khỏi phòng chơi
Biến thể	Không
Ngoại lệ	Không

Bảng 2.3. 10: Kích bản kết thúc game

- Chọn map

Tên	Chọn map
Tác nhân	Người chơi
Mục tiêu	Chọn khung cảnh trong game
Tiền điều kiện	Chọn chế độ người chơi
Mô tả	<ol style="list-style-type: none"> <li>1. Người dùng chọn chế độ chơi đơn hoặc nhiều người chơi</li> <li>2. Người chơi click button “Danh sách bản đồ”</li> <li>3. Người chơi được đưa đến 1 list các map</li> <li>4. Người chơi ấn vào button đầu các map trong list để chọn map</li> </ol>
Hậu điều kiện	Giao diện màn chơi được thay đổi
Biến thể	Không
Ngoại lệ	Không

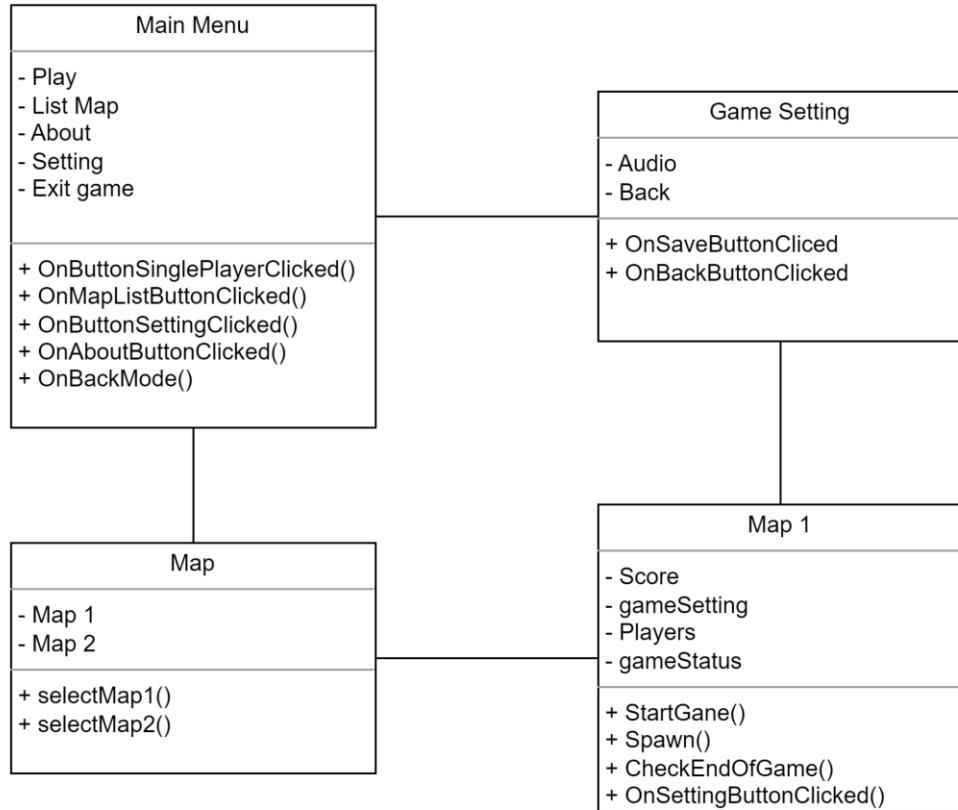
Bảng 2.3. 11: Kịch bản chọn map

- Cài đặt

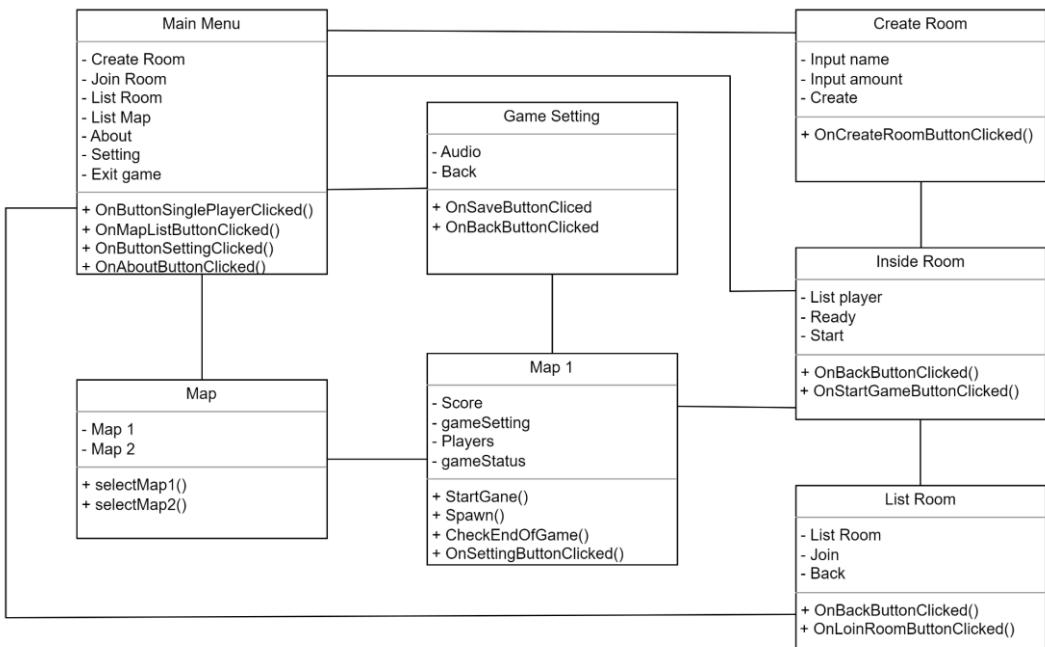
Tên	Cài đặt
Tác nhân	Người chơi
Mục tiêu	Chỉnh sửa thay đổi trong game
Tiền điều kiện	Đã đăng nhập các chế độ chơi game
Mô tả	<ul style="list-style-type: none"> <li>5. Người dùng chọn chế độ chơi đơn hoặc nhiều người chơi</li> <li>6. Người chơi click button “Cài đặt”</li> <li>7. Người chơi được đưa đến 1 trang điều khiển cài đặt trò chơi</li> </ul>
Hậu điều kiện	Thay đổi cài đặt của game như âm thanh,...
Biến thể	Không
Ngoại lệ	Không

Bảng 2.3. 12: Kịch bản cài đặt game

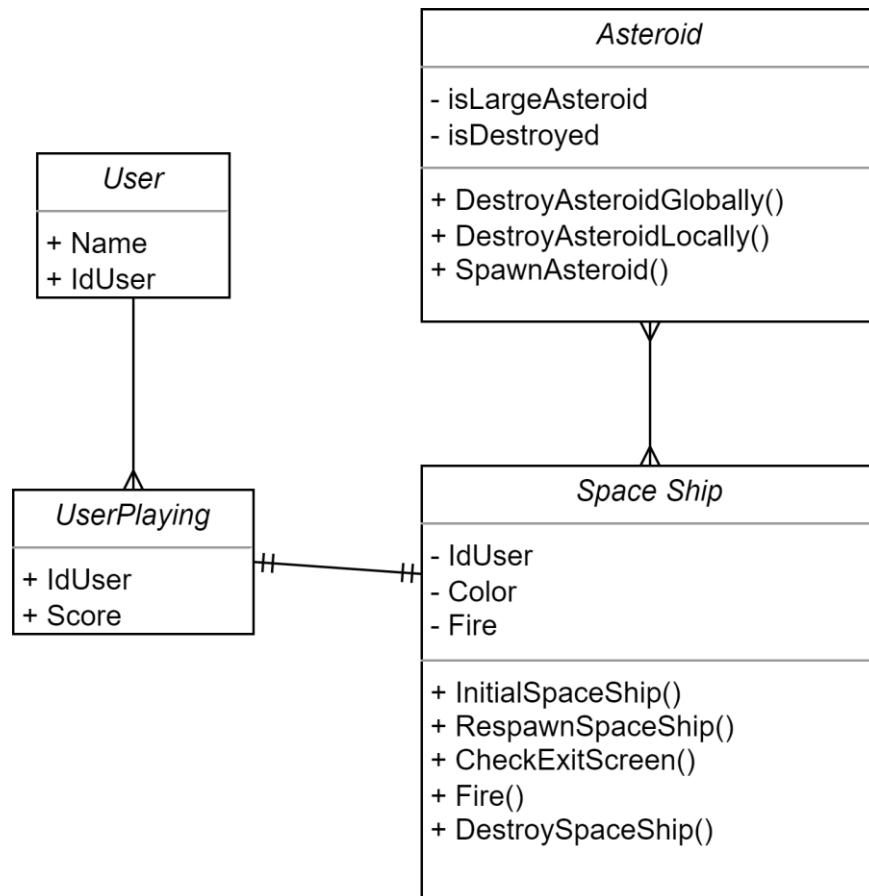
### 3. Biểu đồ lớp phân tích



Hình 2.3. 2: Biểu đồ lớp phân tích chế độ chơi đơn



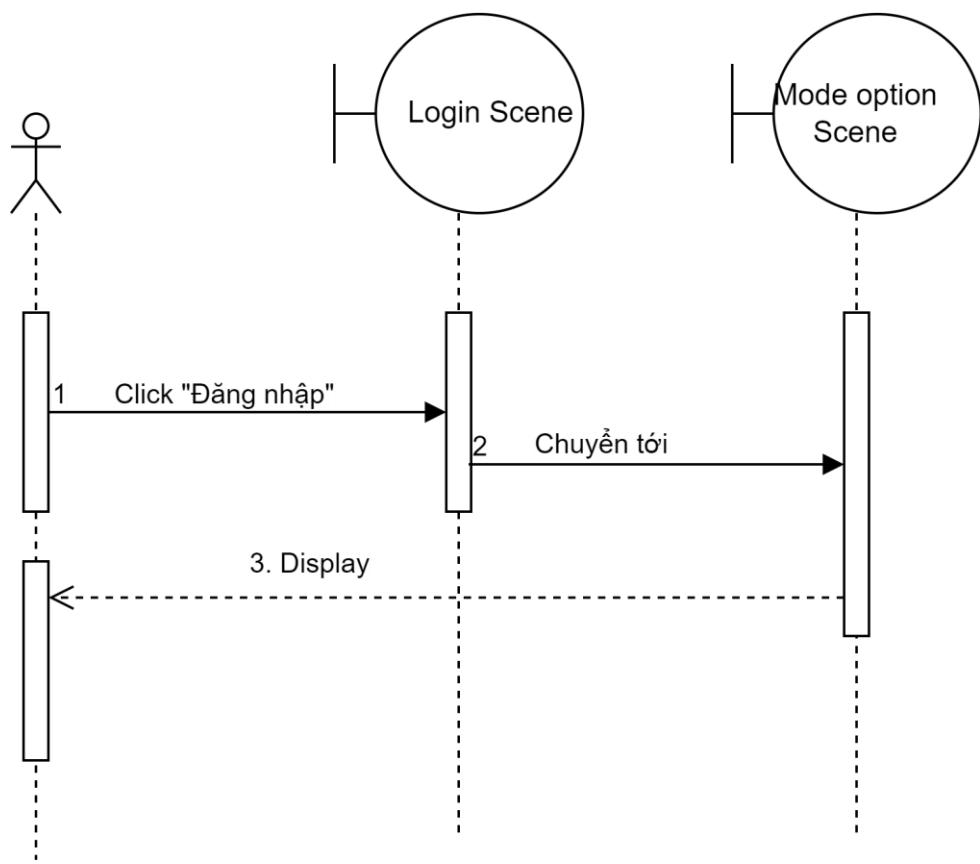
Hình 2.3. 3: Biểu đồ lớp ché đố đa người chơi



Hình 2.3. 4: Biểu đồ lớp các thành phần trong game

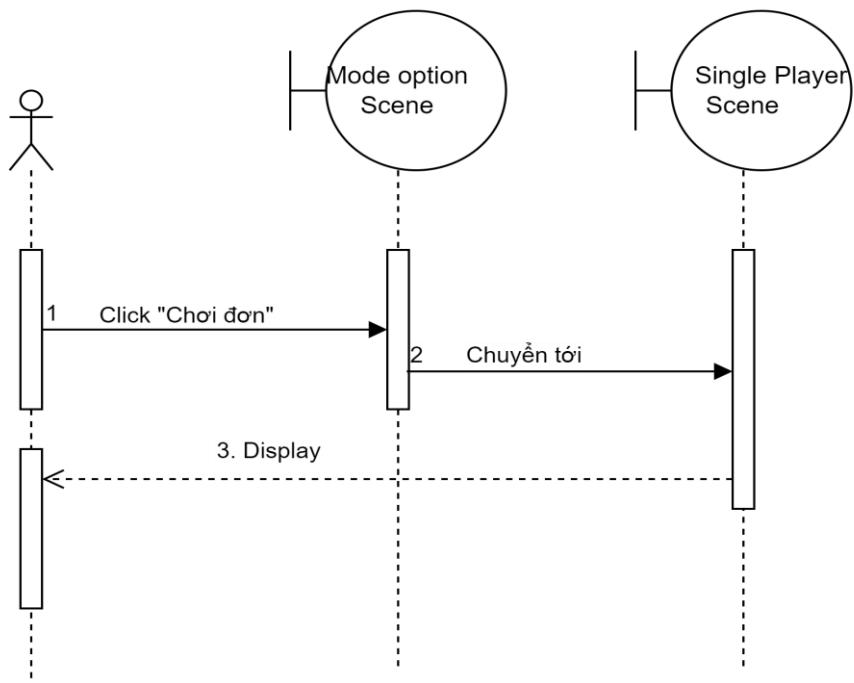
#### 4. Biểu đồ tuần tự

- Biểu đồ tuần tự đăng nhập

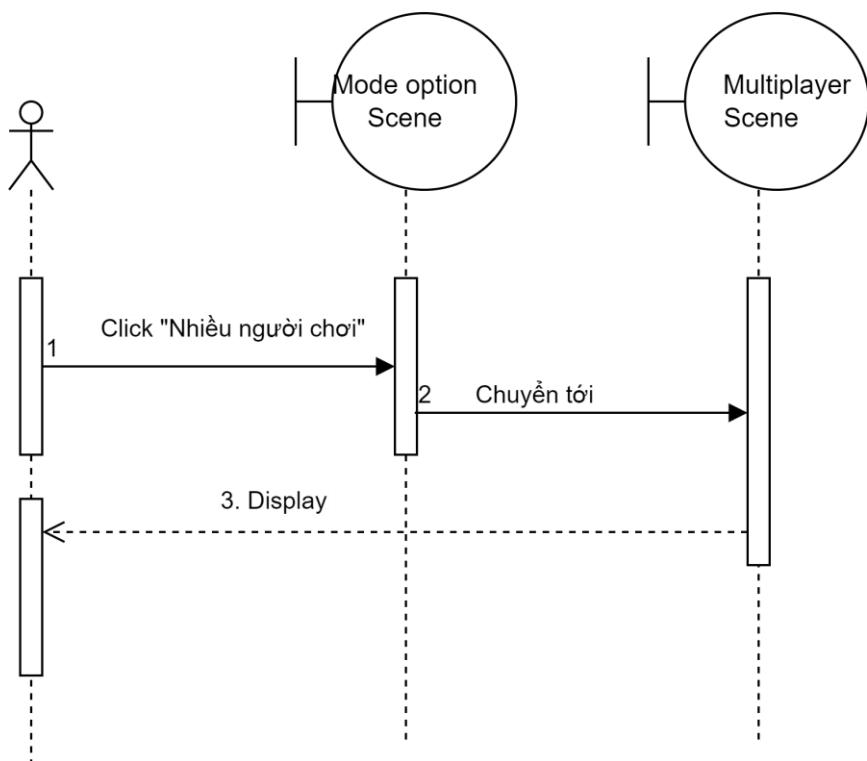


Hình 2.3. 5: Biểu đồ tuần tự đăng nhập

- Biểu đồ tuần tự chọn chế độ chơi

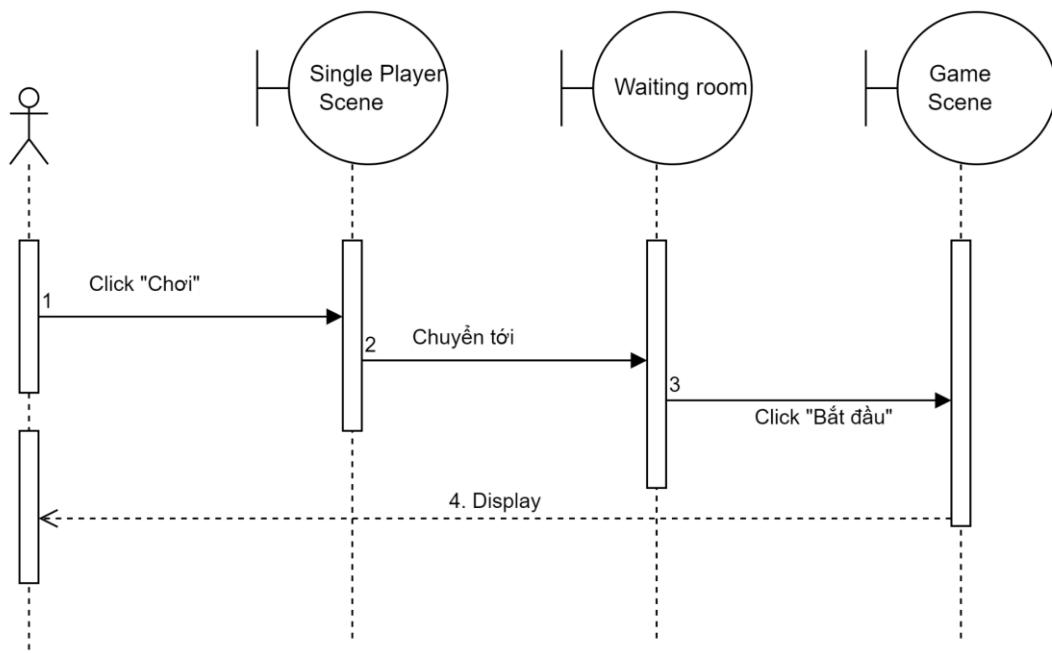


Hình 2.3. 6: Biểu đồ tuần tự chọn chế độ chơi



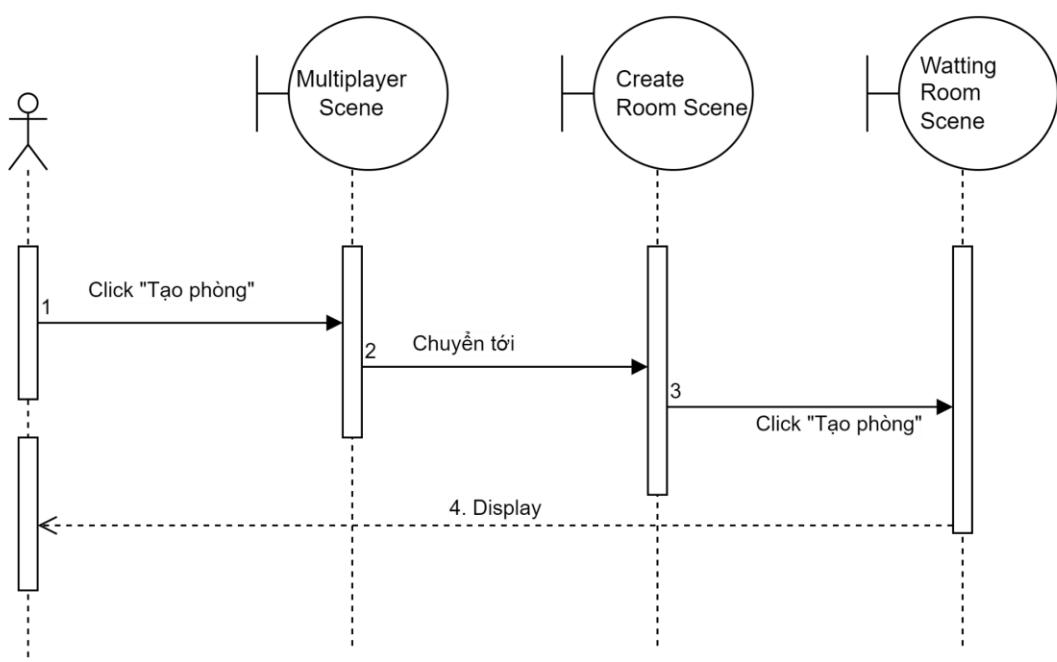
Hình 2.3. 7: Biểu đồ tuần tự chọn chế độ

- Biểu đồ tuần tự chơi game trong chế độ chơi đơn



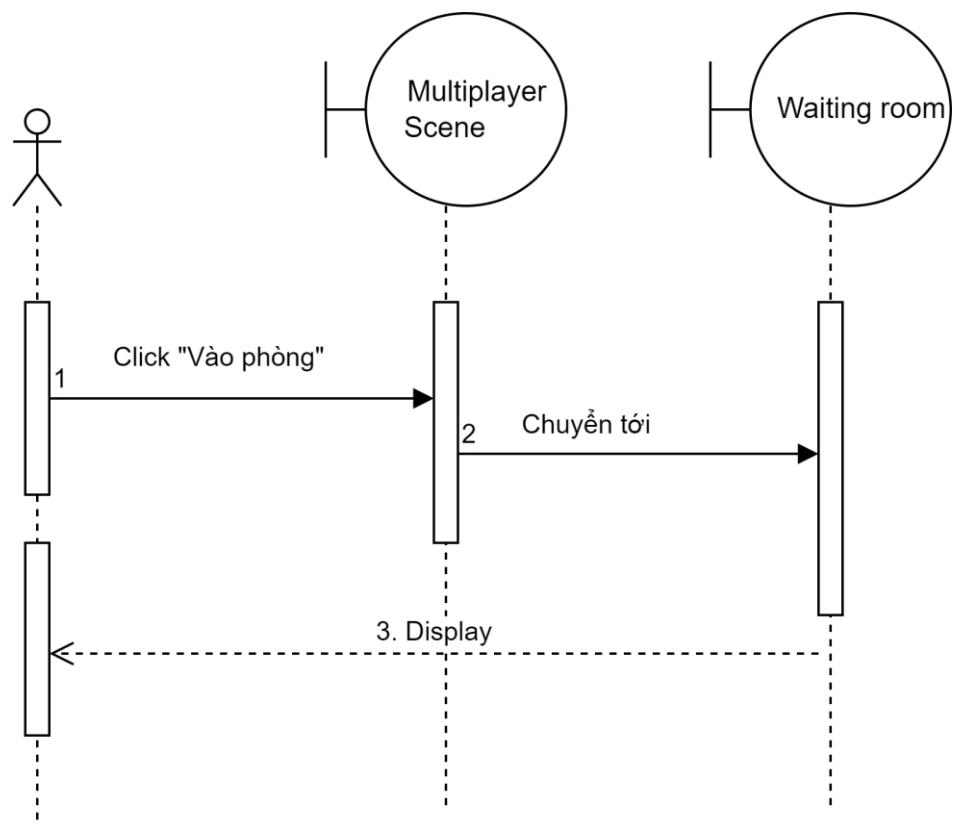
Hình 2.3. 8: Biểu đồ tuần tự chơi trong chế độ chơi đơn

- Biểu đồ tuần tự tạo phòng



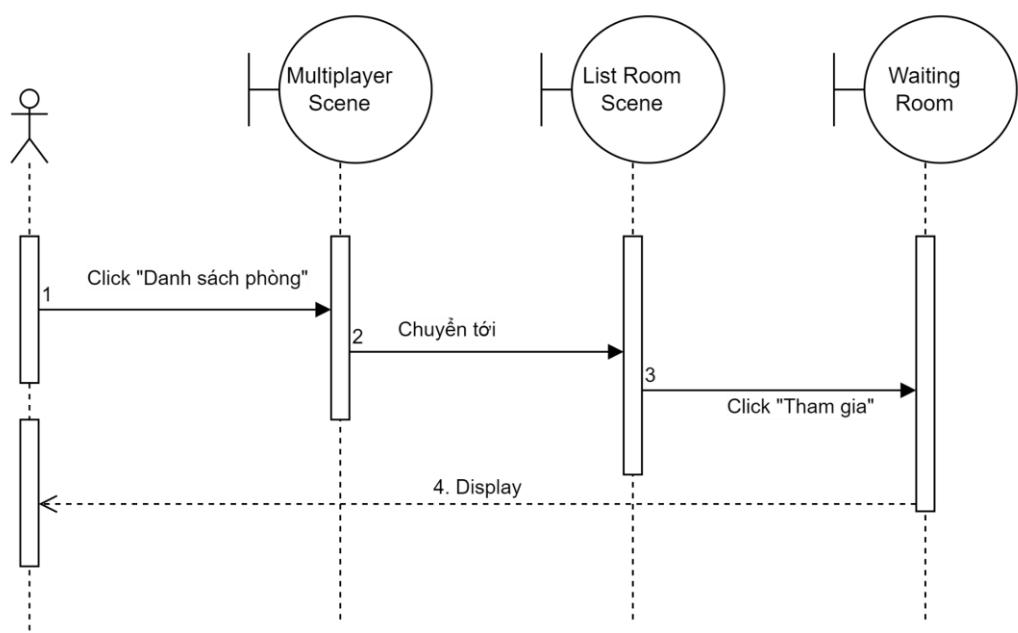
Hình 2.3. 9: Biểu đồ tuần tự tạo phòng

- Biểu đồ tuần tự vào phòng ngẫu nhiên



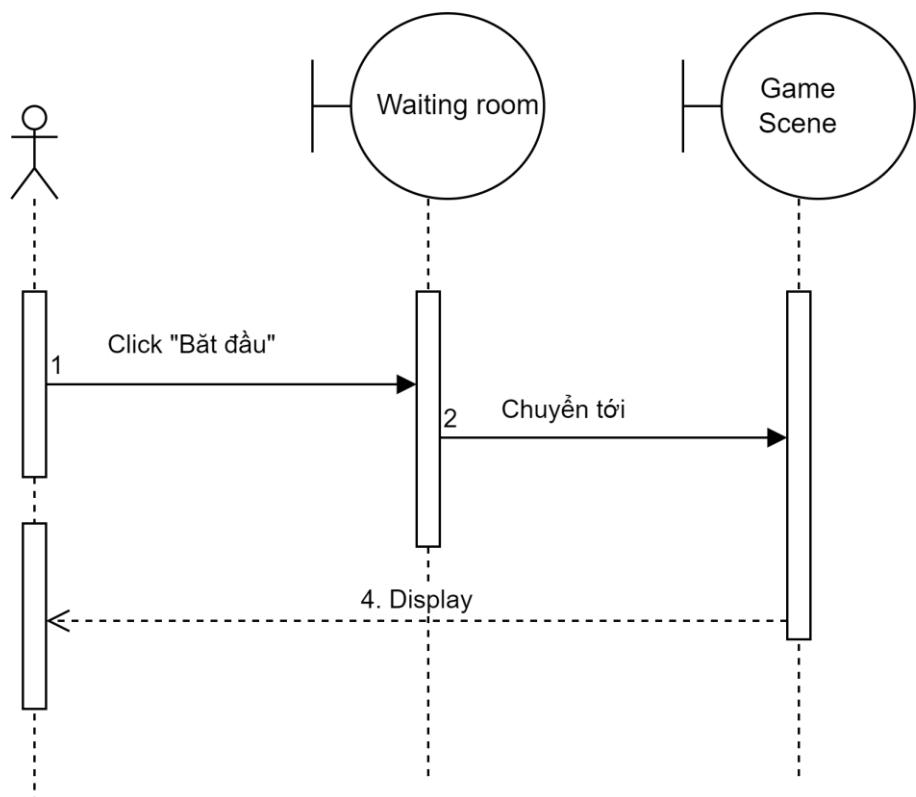
Hình 2.3. 10: Biểu đồ tuần tự vào phòng ngẫu nhiên

- Biểu đồ tuần tự chọn tham gia phòng



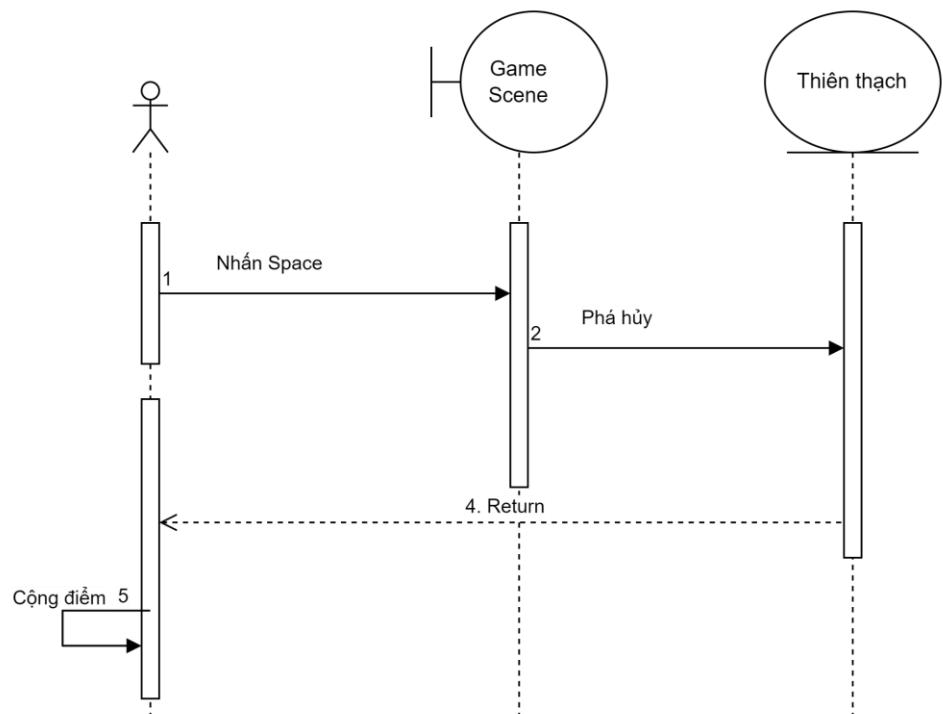
*Hình 2.3. 11: Biểu đồ tuần tự chọn tham gia phòng*

- Biểu đồ tuần tự bắt đầu game



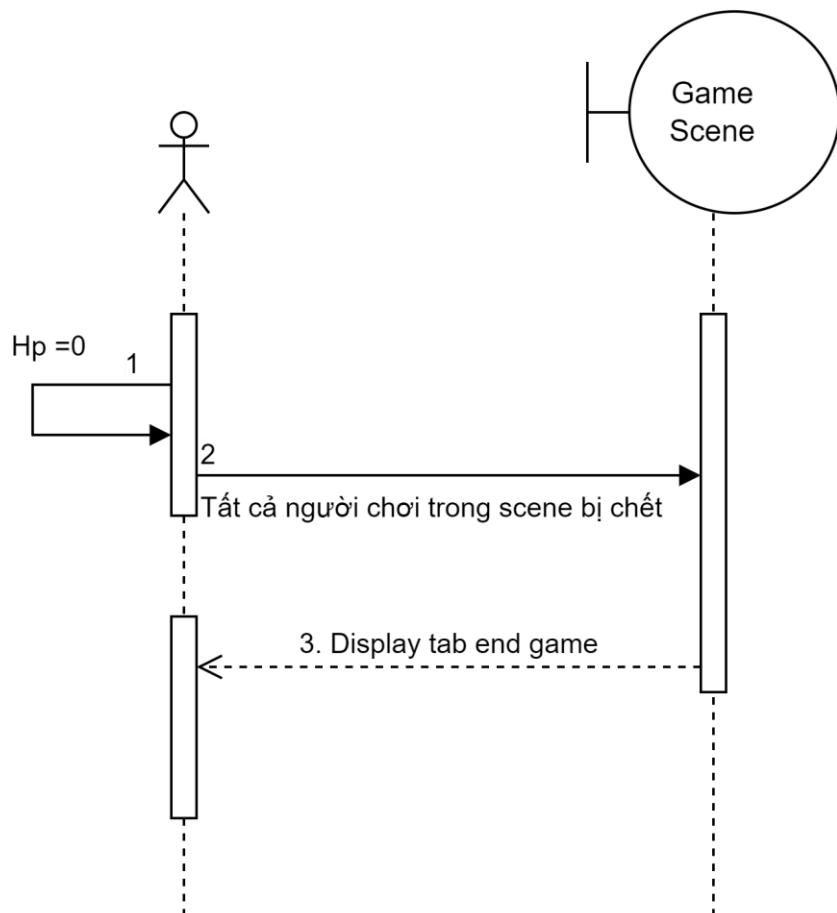
*Hình 2.3. 12: Biểu đồ tuần tự bắt đầu game*

- Biểu đồ tuần tự phá hủy thiên thạch



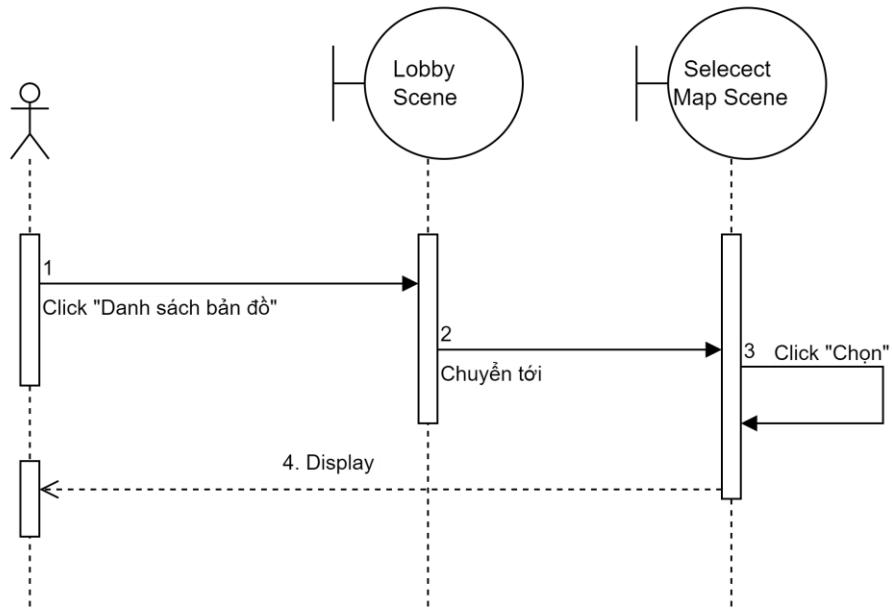
Hình 2.3. 13: Biểu đồ tuần tự phá hủy thiên thạch

- Biểu đồ tuần tự kết thúc game



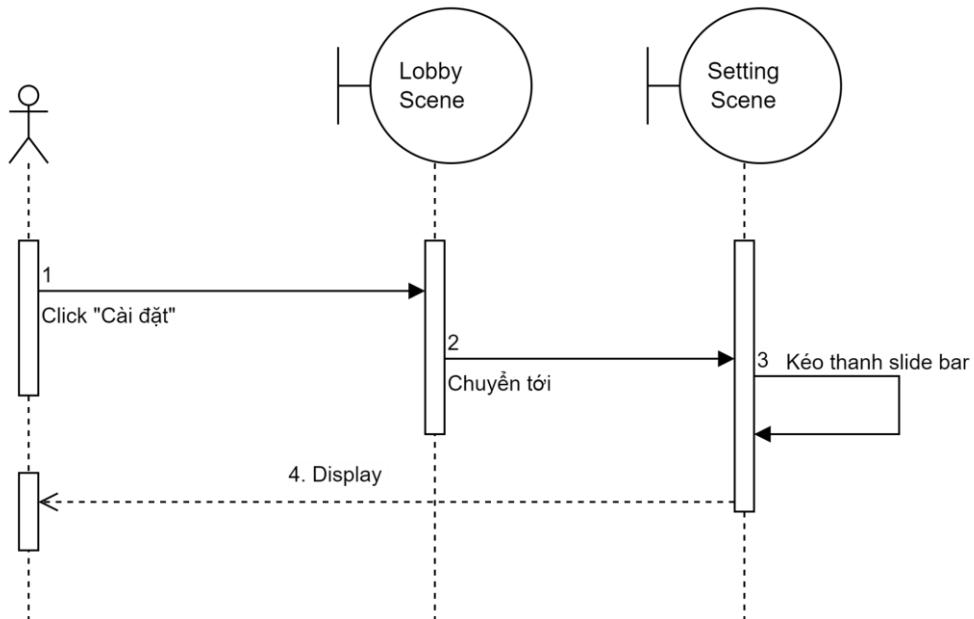
Hình 2.3. 14: Biểu đồ tuần tự kết thúc game

- Biểu đồ tuần tự chọn map



Hình 2.3. 15: Biểu đồ tuần tự chọn map

- Biểu đồ tuần tự cài đặt



Hình 2.3. 16: Biểu đồ tuần tự cài đặt

## IV. GDD game

### 1. GDD

#### ❖ Mô tả trò chơi

**Thể loại:** Game hành động

**Cốt truyện:** Năm 2500 khi mà con người đã hoàn toàn nắm giữ và phát triển được những công nghệ tiên tiến thì con người đã đẩy mạnh ra con đường khai phá vũ trụ rộng lớn. Nhưng trong những cuộc khai phá ra những vùng đất mới trong vũ trụ thì không hề dễ dàng vì con người gặp phải những mảng thiên thạch trôi nổi ngáng đường. Để kích thích và tăng cường tốc độ khai phá thì con người đã tổ chức cuộc thi phá thiên thạch, họ đã sử dụng những chiếc phi thuyền vũ trụ được điều khiển từ xa và con người chỉ cần ngồi trong 1 cabin ảo để điều khiển phi thuyền đó để bắn phá các thiên thạch để lấy điểm. Và người nào phá hủy được nhiều thiên thạch nhất sẽ dành chiến thắng.

**Mục tiêu của trò chơi:** Người chơi khi tham gia trò chơi cần phải điều khiển phi thuyền bắn phá các thiên thạch để lấy được càng nhiều điểm số cho mình càng tốt.

#### ❖ Gameplay

Spaceship là một game có độ khó khá cao nếu người chơi không biết di chuyển chiếc phi thuyền của mình.

**Cơ chế của trò chơi:** Người chơi sẽ phải điều khiển một phi thuyền để né tránh các thiên thạch bay lơ lửng trong vũ trụ và đồng thời phải bắn vỡ các thiên thạch đó để lấy điểm cạnh tranh với những người chơi khác. Người chơi khi vào game chỉ được cung cấp tối đa 3 mạng và khi va chạm với thiên thạch sẽ khiến phi thuyền của người chơi nổ và mất mạng. Sau khi dùng hết số lượt hồi sinh thì người chơi không được hồi sinh nữa và chờ xem liệu người chơi còn lại có vượt qua được số điểm của bản thân hay không.

- Cách thức di chuyển

- Điều khiển:

Button	Action
Up Arrow	Di chuyển lên
Right Arrow	Di chuyển phải
Down Arrow	Di chuyển xuống
Left Arrow	Di chuyển trái
Space	Bắn
Esc	Thoát game

Bảng 2.4. 1: Bảng các nút điều khiển trong game

- Vận tốc di chuyển: Ban đầu vào game phi thuyền sẽ đứng im, khi người chơi giữ nút up arrow trên bàn phím vận tốc sẽ tăng dần với tốc độ là 2f. Khi tốc độ vượt quá 200f thì mặc định tốc độ sẽ được đặt lại là 200f.
- Cơ chế tính điểm:

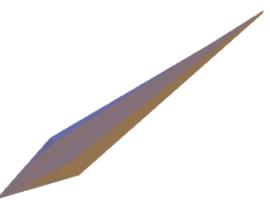
Enemy	Point
Thiên thạch to	2
Thiên thạch nhỏ	1

Bảng 2.4. 2: Cơ chế tính điểm trong game

## ❖ Characters

Các đối tượng trong game:

	Tên	Mô tả
	Thiên thạch to	1 đối tượng trong game, lơ lửng trong không gian trò chơi

	Thiên thạch nhỏ	Khi các phi thuyền bắn ra những viên đạn sẽ gây nổ những viên thiên thạch nhỏ và đối với những viên thiên thạch lớn chúng sẽ bị tách ra thành những viên nhỏ.
	Phi thuyền	Đại diện cho 1 người chơi trong game, mỗi người chơi khi tham gia trò chơi sẽ được phân biệt với nhau bởi các màu riêng biệt và các phi thuyền trong trò chơi đều giống nhau.
	Đạn	Các phi thuyền có thể cạnh tranh với nhau bằng cách bắn những viên đạn về phía họ để đẩy người chơi khác.

Bảng 2.4. 3: Các đối tượng trong game

- Cơ chế tạo màu cho phi thuyền: Trong một file script đã lưu một hàm GetColor() và tham số đầu vào của hàm này là một số thứ tự chính là số thứ tự của người chơi đó trong listEntry. trong hàm GetColor() sử dụng switch...case để return lại màu tương ứng cho các số thứ tự truyền vào.

<b>Stt</b>	<b>Color</b>
0	red
1	green
2	blue
3	yellow
4	cyan
5	grey
6	magenta
7	white

*Bảng 2.4. 4: Màu sắc các phi thuyền xuất hiện trong game*

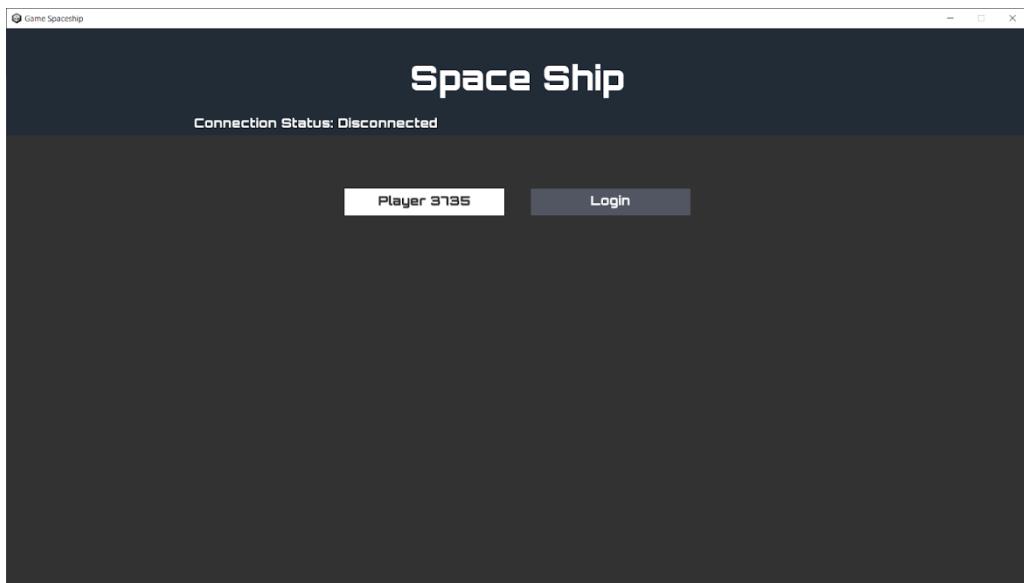
#### ❖ **Worlds/Levels**

Bối cảnh trong game là ở trong một chiề̂u không gian ngoài vũ trụ ở đó luôn có những viên thiên thạch bay lơ lửng với những quỹ đạo khác nhau.

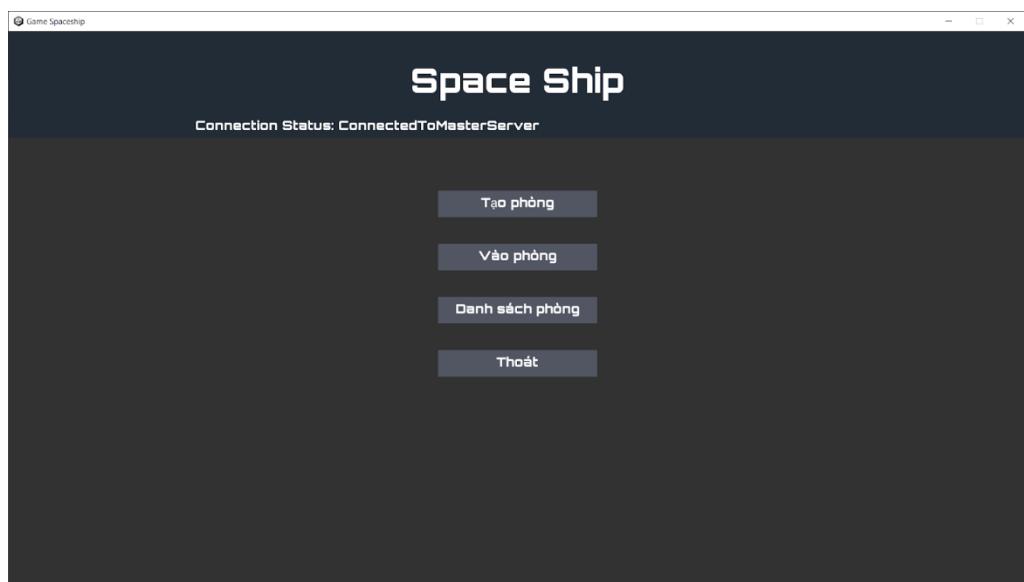
Người chơi khi tham gia chơi cần phải phá hủy những thiên thạch đang lao về phía phi thuyền của mình để lấy điểm và phải hạn chế va chạm với thiên thạch

#### ❖ **User Interface**

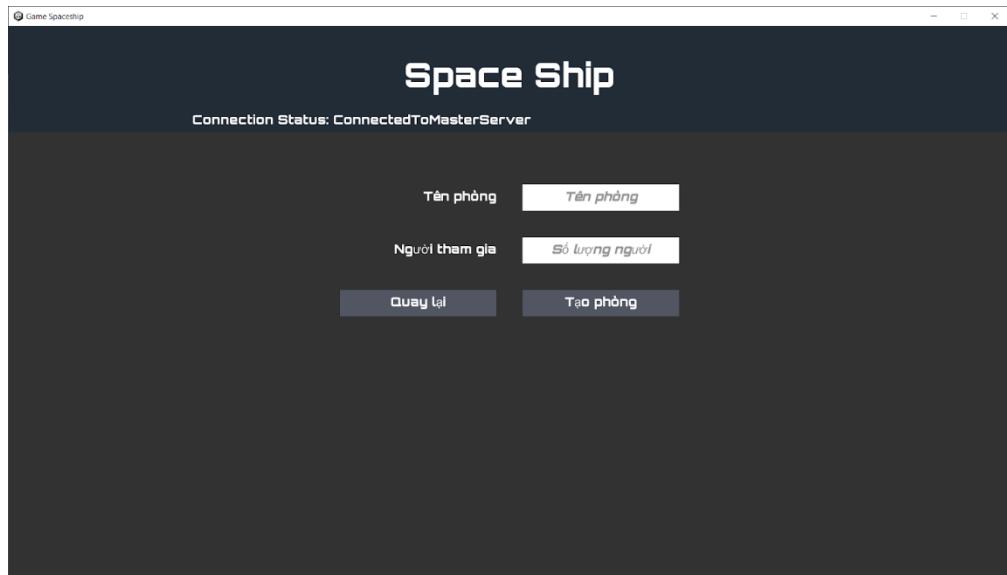
Trò chơi được thiết kế với UI đơn giản giúp người chơi có thể dễ dàng sử dụng



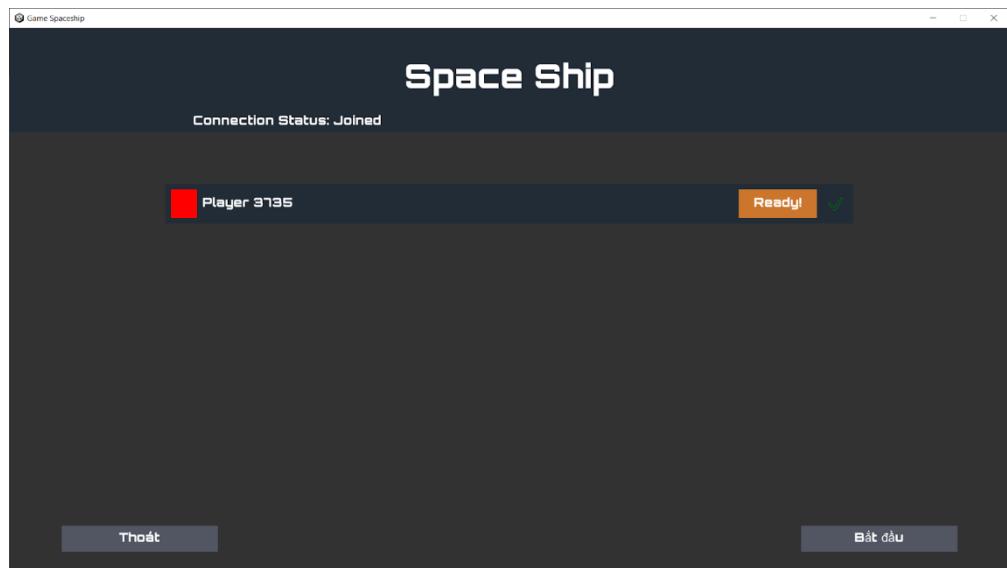
Hình 2.4. 1.Màn hình đăng nhập



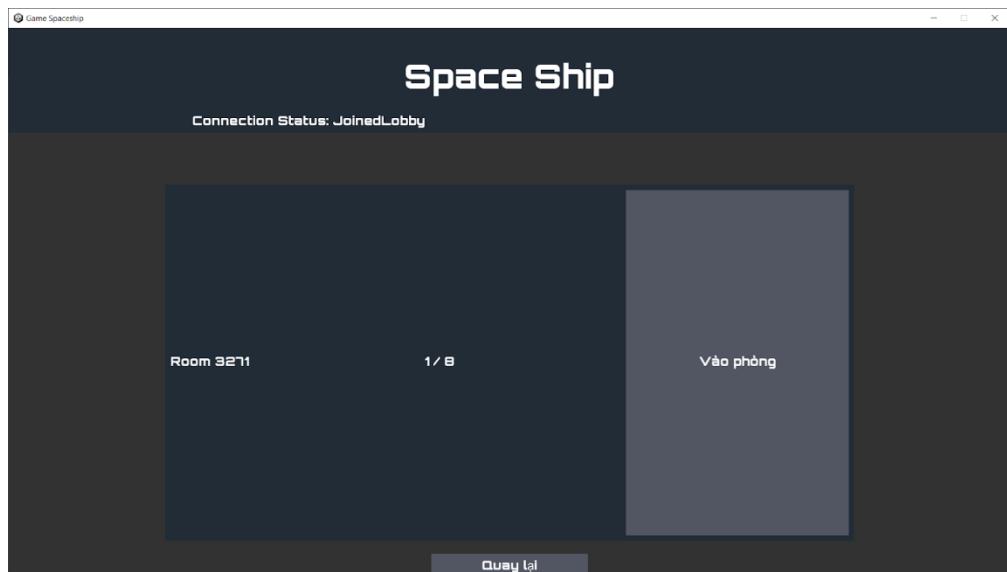
Hình 2.4. 2.Sảnh chờ



Hình 2.4. 3.Tạo room



Hình 2.4. 4: Vào phòng



Hình 2.4. 5: Room List



Hình 2.4. 6: Phòng chơi

## ❖ Audio

Spaceship được thiết kế với âm thanh nền sống động

Hiệu ứng âm thanh của trò chơi bao gồm:

- Hiệu ứng âm thanh khi phi thuyền bắn đạn
- Hiệu ứng âm thanh khi phi thuyền bị nổ

## ❖ Artwork

Đồ họa của game là những hình khối 3D



Hình 2.4. 7: Các model 3D trong game

## 2. GDD one page

**2.1 Tổng quan:** Spaceship là một game về thể loại hành động, người chơi sẽ điều khiển một phi thuyền trong không gian để phá hủy những thiên thạch vũ trụ bay lơ lửng

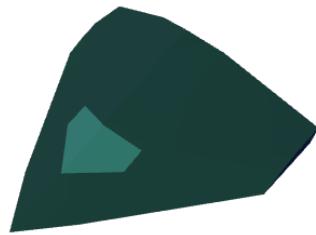
**2.2 Cơ chế trò chơi:** Người chơi sẽ phải điều khiển một phi thuyền để né tránh các thiên thạch bay lơ lửng trong vũ trụ và đồng thời phải bắn vỡ các thiên thạch đó để lấy điểm cạnh tranh với những người chơi khác. Người chơi khi vào game chỉ được cung cấp tối đa 3 mạng và khi va chạm với thiên thạch sẽ khiến phi thuyền của người chơi nổ và mất mạng. Sau khi dùng hết số lượt hồi sinh thì người chơi không được hồi sinh nữa và chờ xem liệu người chơi còn lại có vượt qua được số điểm của bản thân hay không.

**2.3 Cốt truyện:** Năm 2500 khi mà con người đã hoàn toàn nắm giữ và phát triển được những công nghệ tiên tiến thì con người đã đẩy mạnh ra con đường khai phá vũ trụ rộng lớn. Nhưng trong những cuộc khai phá ra những vùng đất mới trong vũ trụ thì không hề dễ dàng vì con người gặp phải những mảng thiên thạch trôi nổi ngáng đường. Để kích thích và tăng cường tốc độ khai phá thì con người đã tổ chức cuộc thi phá thiên thạch, họ đã sử dụng những chiếc phi thuyền vũ trụ được điều khiển từ xa và con người chỉ cần ngồi trong 1 ca bin ảo để điều khiển phi thuyền

đó để bắn phá các thiên thạch để lấy điểm. Và người nào phá hủy được nhiều thiên thạch nhất sẽ dành chiến thắng.

#### **2.4 Nhân vật:**

- Phi thuyền



*Hình 2.4. 8: Ảnh phi thuyền*

- Thiên thạch



*Hình 2.4. 9: Hình ảnh thiên thạch*

**2.5. Phong cách đồ họa:** Đồ họa trong game là đồ họa hình khối 3D

**2.6. Đối tượng mục tiêu:** Trò chơi dành cho lứa tuổi 12+

**2.7. Nền tảng:** trò chơi bước đầu hướng tới phát triển trên PC

### **3. Beat chart**

	<b>Bối cảnh</b>
<b>Name</b>	Ở trong không gian của thiên hà 01
<b>Progression</b>	Con người cần khai phá vũ trụ nên đã tổ chức một cuộc thi phá hủy thiên thạch để kích thích và đẩy nhanh việc khai phá. Và con người đã dùng những chiếc phi thuyền điều khiển từ xa để phá hủy những thiên thạch đó, nhiệm vụ của những người tham gia cuộc thi là phá hủy càng nhiều thiên thạch dành cho mình nhiều điểm số nhất
<b>Story beat</b>	Tham gia vào chiều không gian vũ trụ, làm quen với việc điều khiển phi thuyền
<b>Gameplay</b>	Di chuyển, bắn đạn
<b>Enemies</b>	Những viên thiên thạch
<b>Estimated time</b>	Không giới hạn
<b>Color</b>	Màu tím than, xám đen

<b>Music</b>	Nhạc nền  Hiệu ứng âm thanh: Tiếng bắn đạn, tiếng nổ của tàu khi bị phá hủy
--------------	---

Bảng 2.4. 5: Beat chart map 1

	<b>Bối cảnh</b>
<b>Name</b>	Ở trong không gian thiên hà 02
<b>Progression</b>	Sau khi cuộc tấn công, khai phá thiên hà 01 đã hoàn tất con người đã nhanh chóng tiến đến thiên hà số 02. Ở thiên hà này có nhiều hành tinh có những khoáng vật mới lạ và có ích cho con người nên họ càng thôi thúc việc khai phá.
<b>Story beat</b>	Không giống với thiên hà 01, thiên hà 02 này người chơi cần phải vừa tiêu diệt thiên thạch vừa phải thu thập những tài nguyên trôi nổi trong thiên hà này.
<b>Gameplay</b>	Di chuyển, bắn đạn, thu thập những tài nguyên vũ trụ
<b>Enemies</b>	Những viên thiên thạch

<b>Estimated time</b>	Không giới hạn
<b>Color</b>	Màu tím than, xám đen
<b>Music</b>	Nhạc nền  Hiệu ứng âm thanh: Tiếng bắn đạn, tiếng nổ của tàu khi bị phá hủy, âm thanh thu thập thành công tài nguyên.

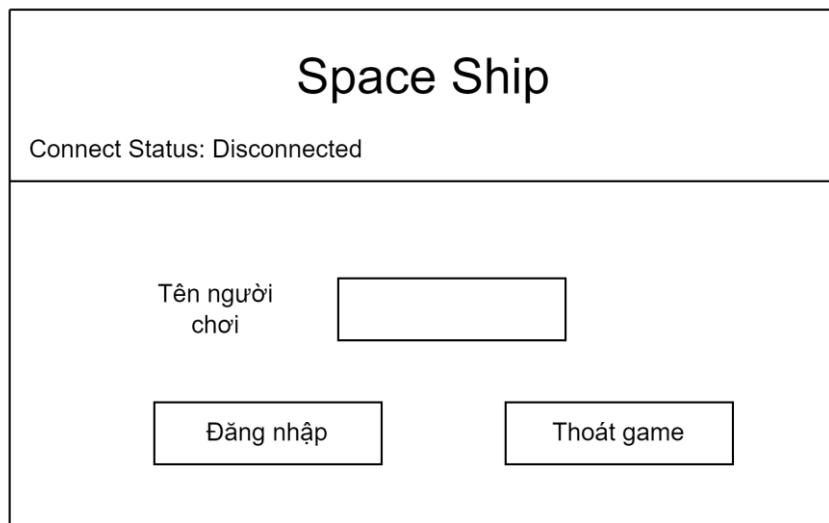
Bảng 2.4. 6: Beat chart map 2

	<b>Bối cảnh</b>
<b>Name</b>	Ở trong không gian của thiên hà 03
<b>Progression</b>	Việc thu thập tài nguyên ở thiên hà số 02 đã đến giới hạn, con người cần tìm ra một thiên hà mới để có thể tìm kiếm tài nguyên mới. Và trong một lần vô tình đi vào một chiều không gian khác, một phi thuyền đã phát hiện ra đường đến một thiên hà. Sau đó thiên hà này được đặt tên là thiên hà 03.
<b>Story beat</b>	Trong một cuộc thi đang được tổ chức, mọi chuyện đang diễn ra tốt đẹp thì bỗng nhiên có một vài phi thuyền tham gia bị nổ. Sau đó con người đã xác nhận

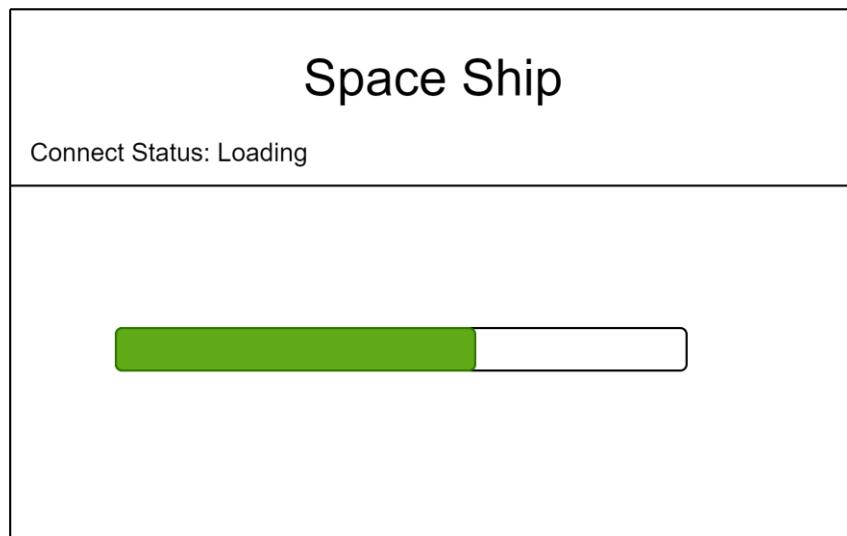
	<p>những phi thuyền đó đã bị bắn bởi những tên lửa không có từ trái đất. Điều này đã làm chấn động ngành công nghiệp khai thác vũ trụ. Từ đó bên cạnh việc cạnh tranh với những người chơi khác thì con người còn phải đối mặt với những sinh vật đến từ ngoài Trái Đất cạnh tranh trong công cuộc khai phá vũ trụ.</p>
<b>Gameplay</b>	Di chuyển, bắn đạn, thu thập tài nguyên, tiêu diệt những phi thuyền ngoài Vũ trụ
<b>Enemies</b>	Những viên thiên thạch, phi thuyền của sinh vật ngoài Vũ trụ
<b>Estimated time</b>	Không giới hạn
<b>Color</b>	Màu tím than, xám đen, đỏ
<b>Music</b>	Nhạc nền  Hiệu ứng âm thanh: Tiếng bắn đạn, tiếng nổ của tàu khi bị phá hủy, âm thanh thu thập thành công tài nguyên

Bảng 2.4. 7: Beat chart map 3

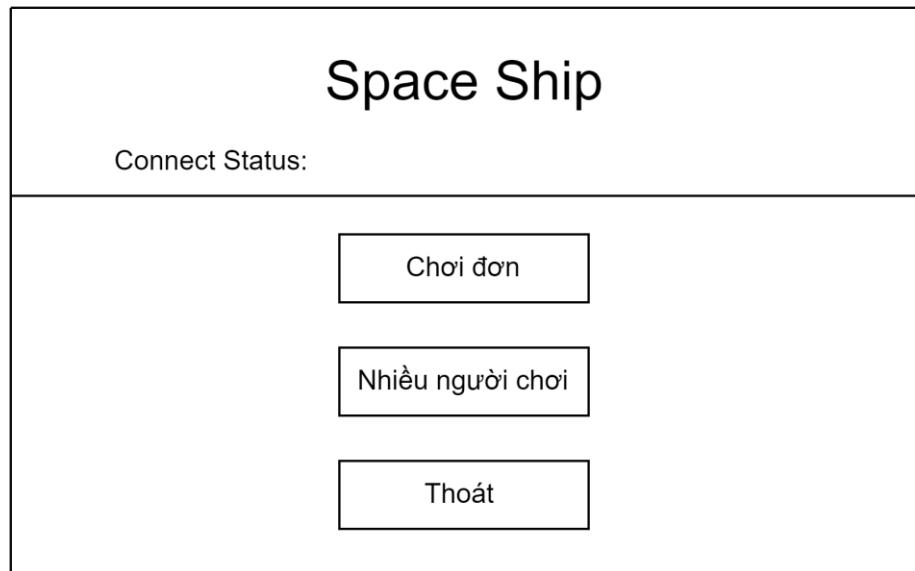
## V. Wireframe



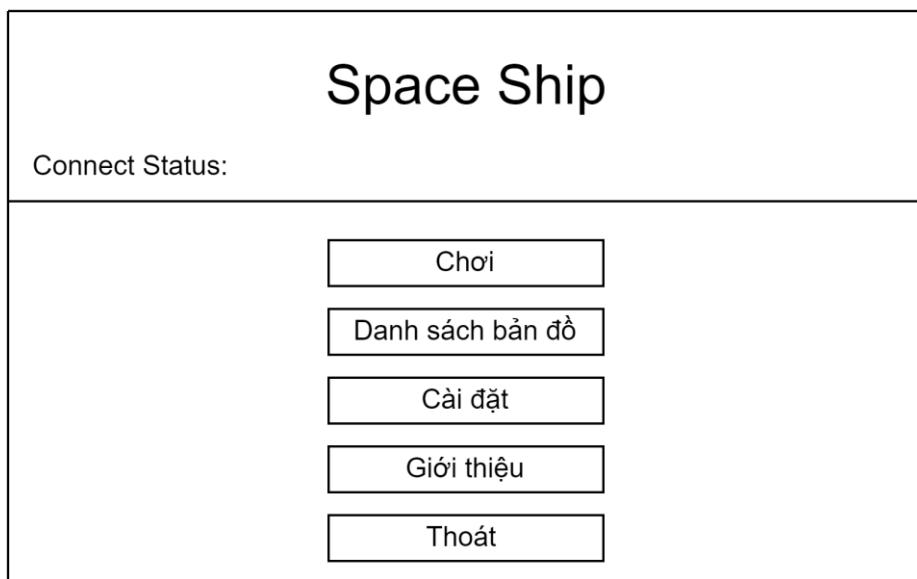
Hình 2.5. 1: Wireframe màn hình đăng nhập



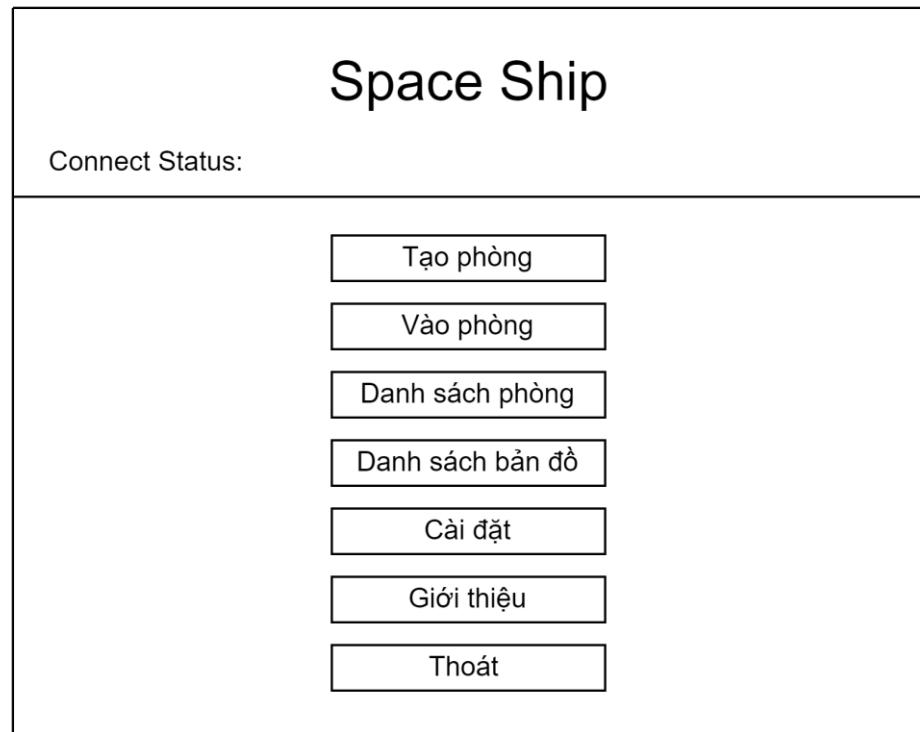
Hình 2.5. 2: Wireframe Đăng nhập và Thanh chờ



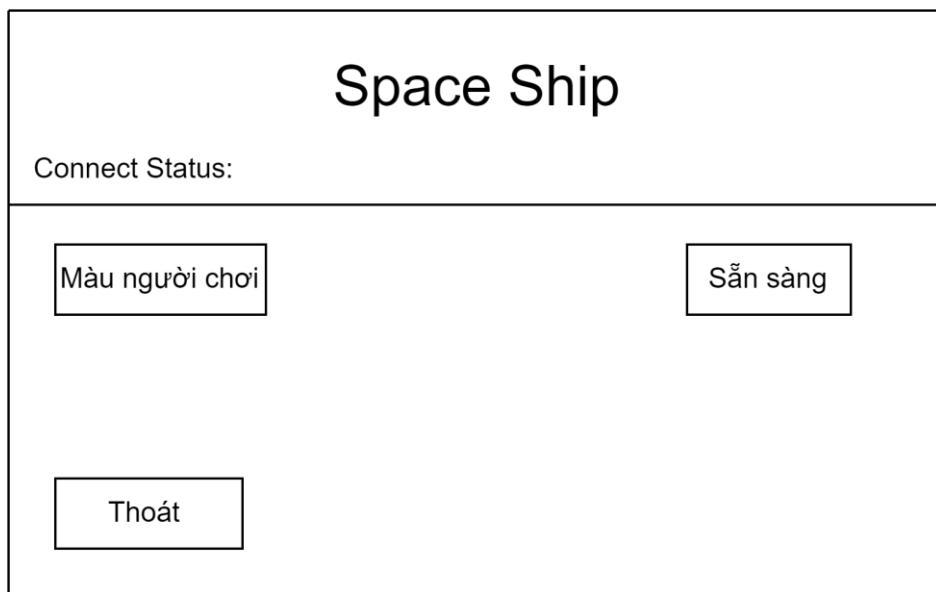
Hình 2.5. 3: Wireframe Chọn chế độ



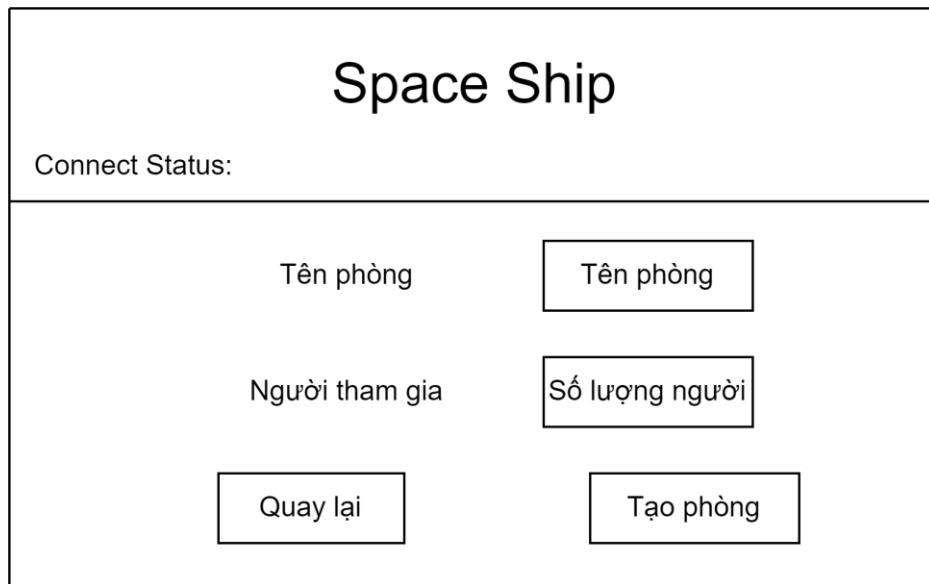
Hình 2.5. 4: Wireframe Lobby của chế độ chơi đơn



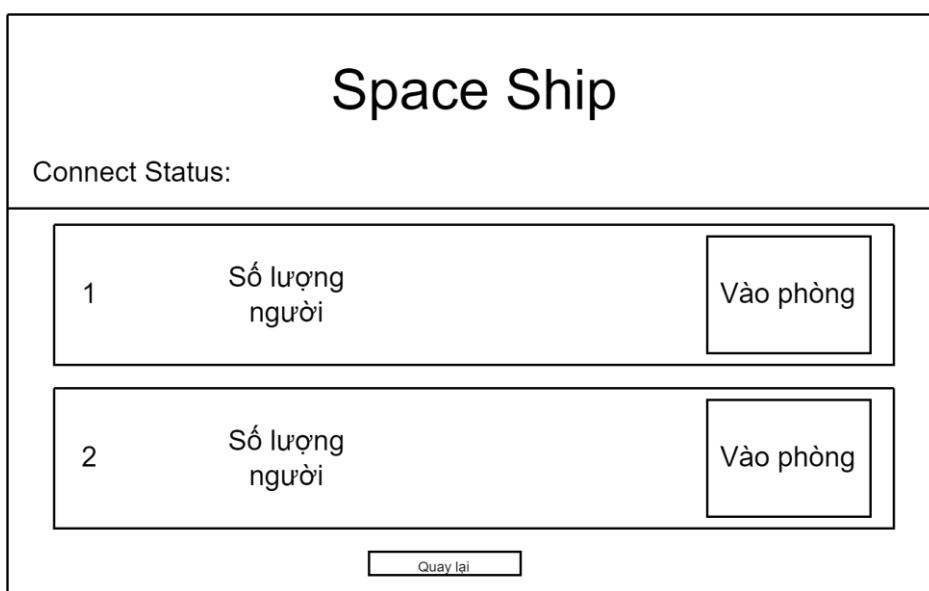
Hình 2.5. 5: Wireframe Lobby của chế độ nhiều người chơi



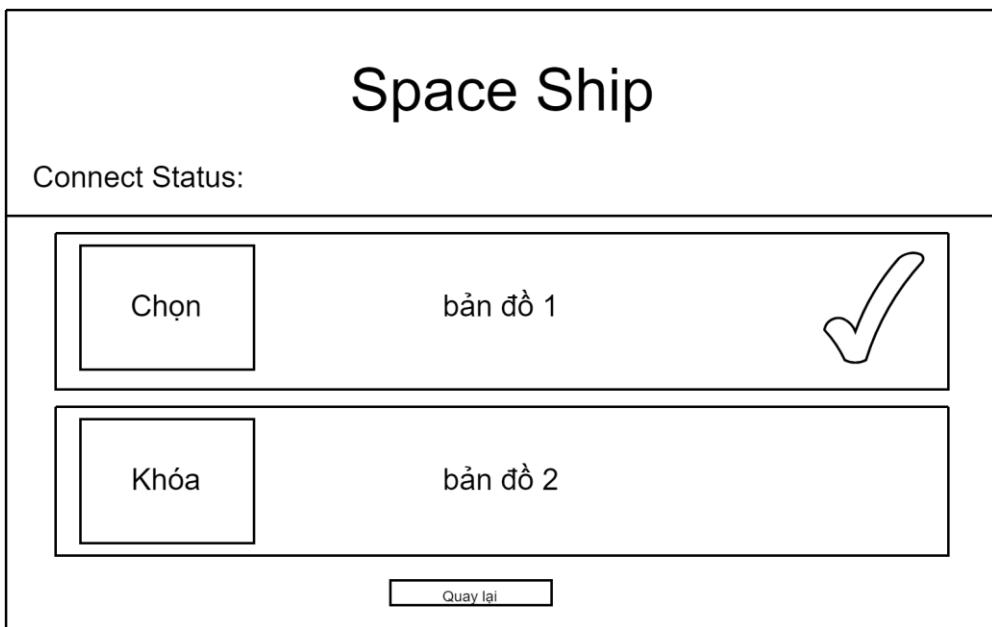
Hình 2.5. 6: Wireframe Vào phòng



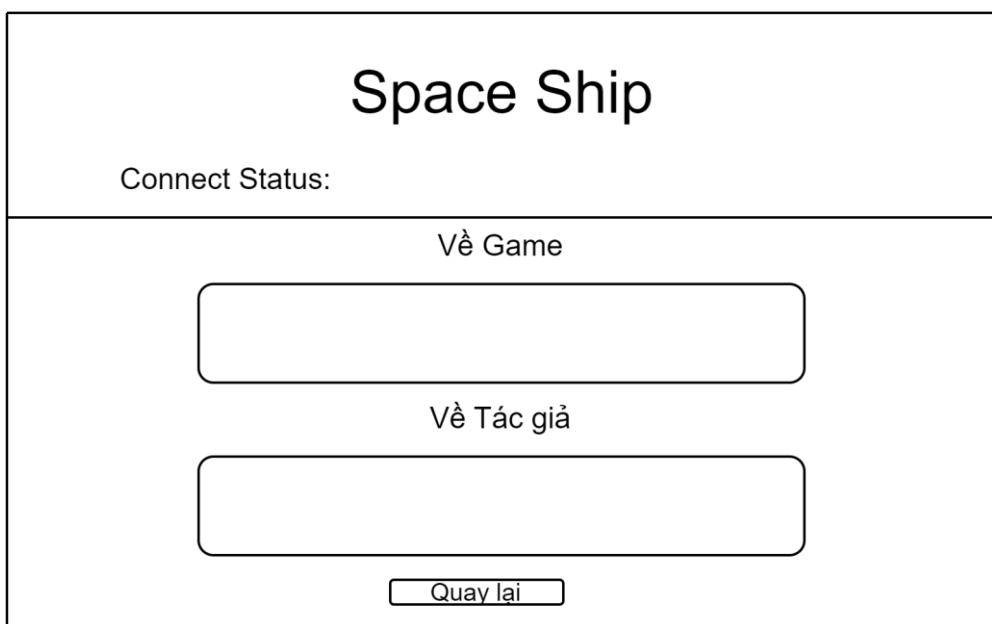
Hình 2.5. 7: Wireframe Tạo phòng



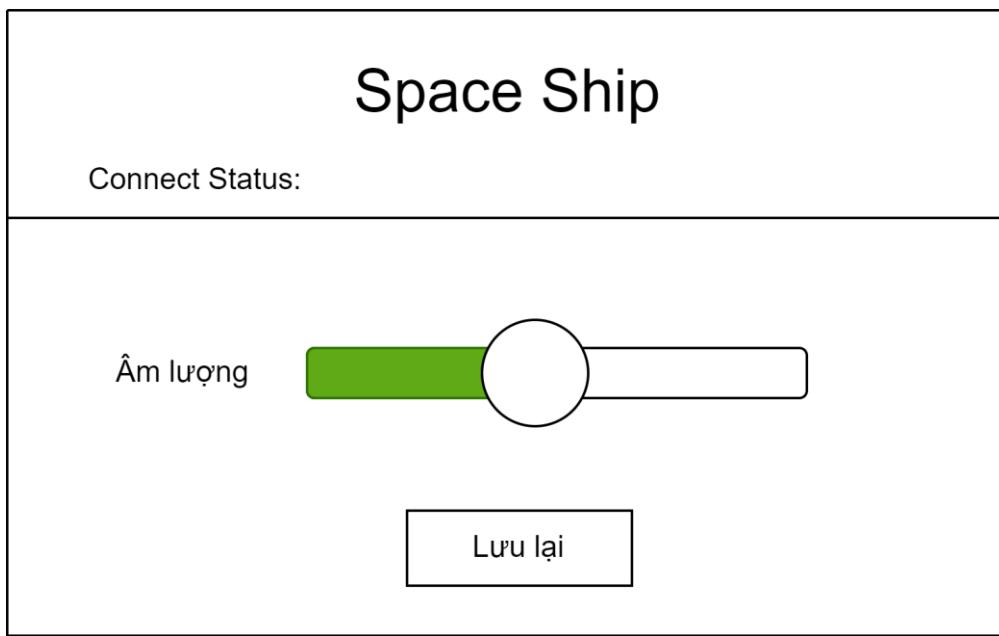
Hình 2.5. 8: Wireframe Danh sách phòng



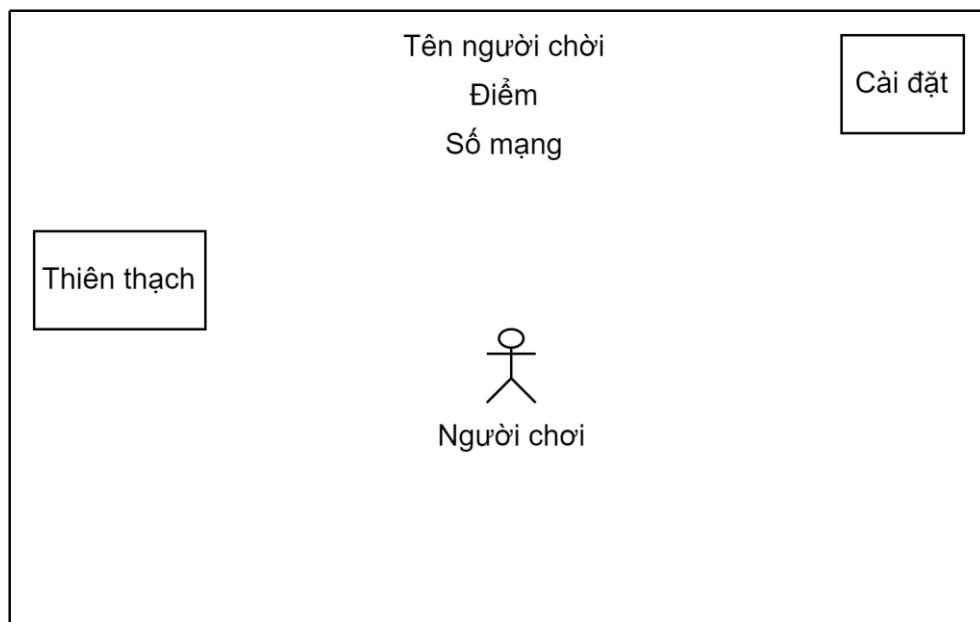
Hình 2.5. 9: Wireframe Chọn map



Hình 2.5. 10: Wireframe About us



Hình 2.5. 11: Wireframe Setting



Hình 2.5. 12: Wireframe Giao diện màn chơi



*Hình 2.5. 13: Wireframe Kết thúc game*

## VI. Tiêu kết chương 2

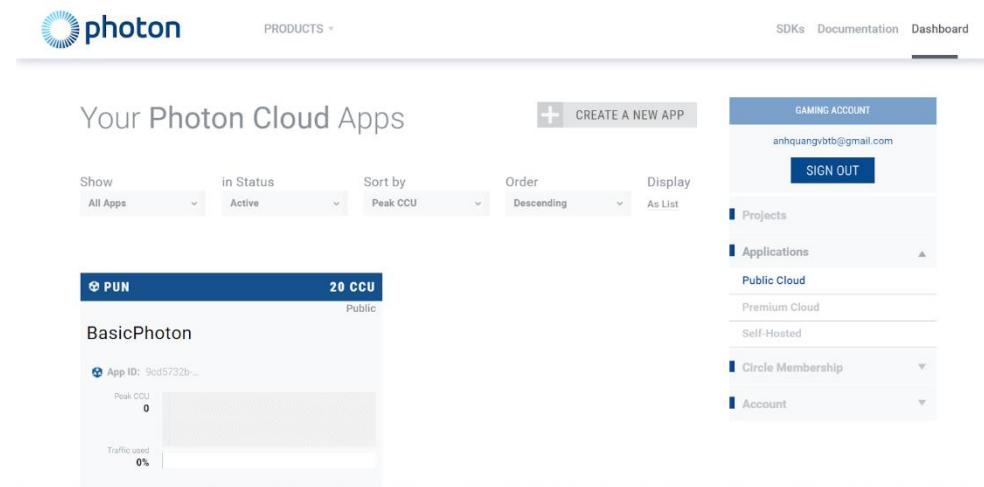
Phân tích thiết kế là một giai đoạn rất quan trọng trong quá trình làm game. Đây có thể coi như là bước đệm để các nhà phát triển sâu hơn trong dự án của mình. Bên cạnh đó nhờ có phần phân tích thiết kế thì bước phát triển game ở phần sau này sẽ có thể giảm bớt đi được phần nào khó khăn.

# CHƯƠNG 3: CÀI ĐẶT VÀ TRIỂN KHAI GAME

## I. Cài đặt game

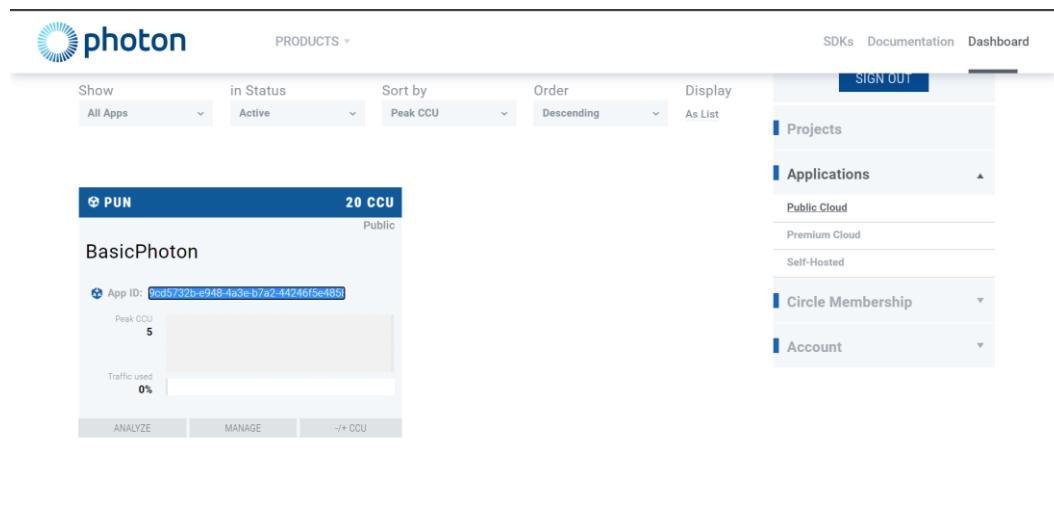
### Import và cài đặt PUN 2

**Bước 1: Đăng ký tài khoản:** Truy cập trang web của Photon Engine và đăng ký tài khoản để có thể sử dụng dịch vụ.



Hình 3.1. 1: Photon Dashboard

**Bước 2: Tạo ứng dụng:** Sau khi đăng nhập vào tài khoản, ta cần tạo một ứng dụng trên trang web của Photon Engine. Ứng dụng này sẽ được sử dụng để kết nối các người chơi lại với nhau. Sau đó ta cần phải copy App ID để phục vụ cho việc kết nối sau này



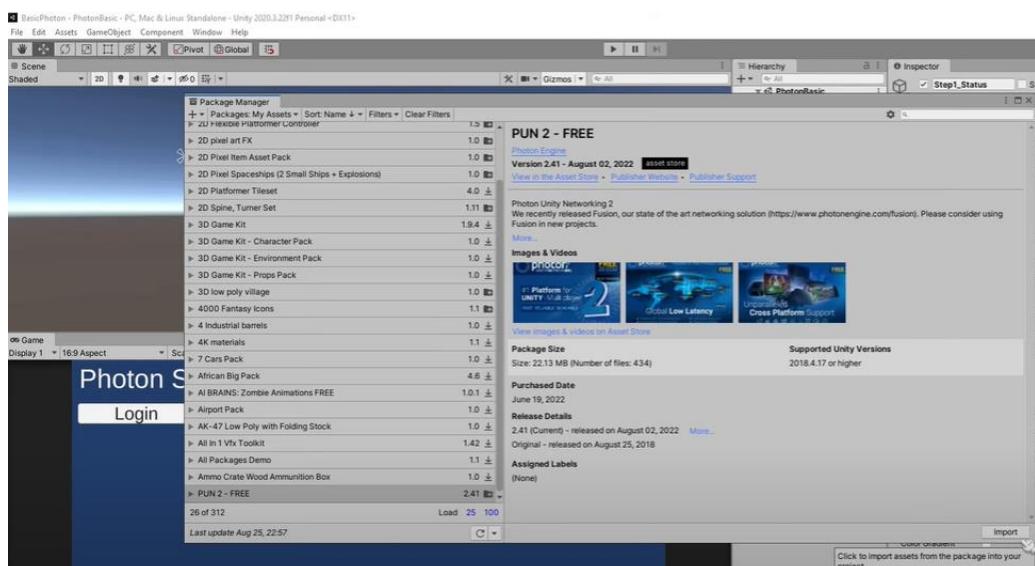
Hình 3.1. 2: Tạo ứng dụng trong Photon

**Bước 3: Tải SDK:** Photon Engine cung cấp các SDK cho các ngôn ngữ lập trình khác nhau. Ta cần tải SDK phù hợp với ngôn ngữ lập trình bạn đang sử dụng. Vd: Unity



Hình 3.1. 3: Add PUN 2

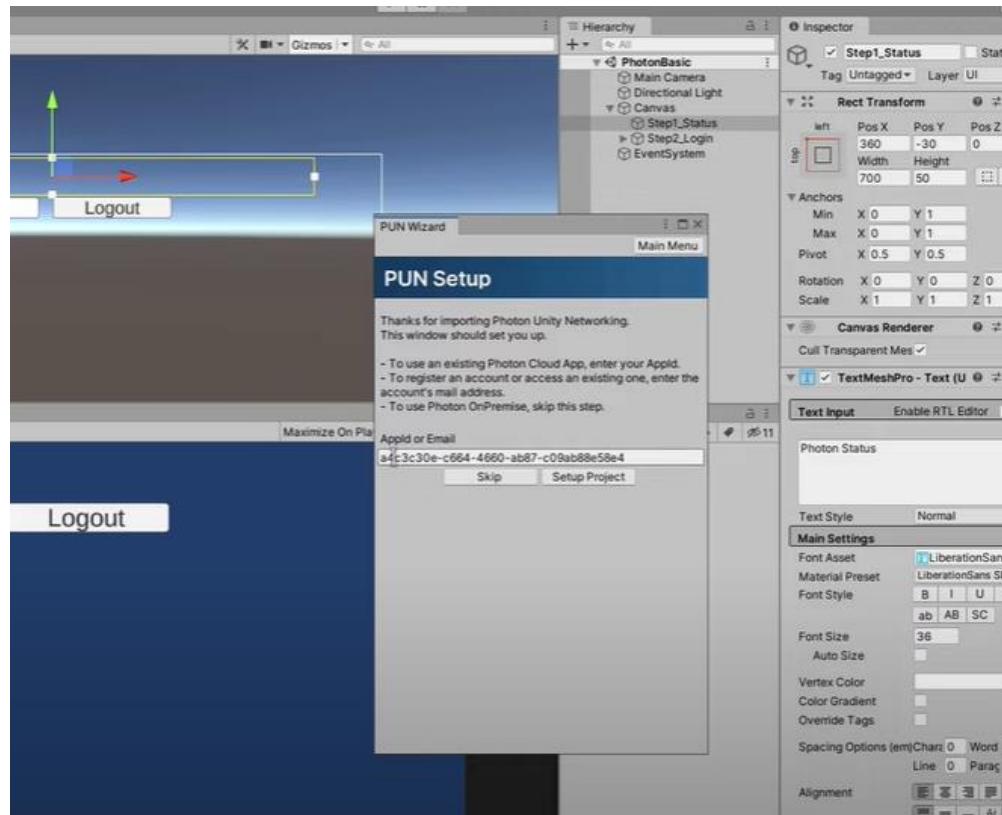
**Bước 4: Thêm SDK vào dự án:** Sau khi tải SDK, ta cần thêm các thư viện của SDK vào dự án của mình.



Hình 3.1. 4. Import PUN 2

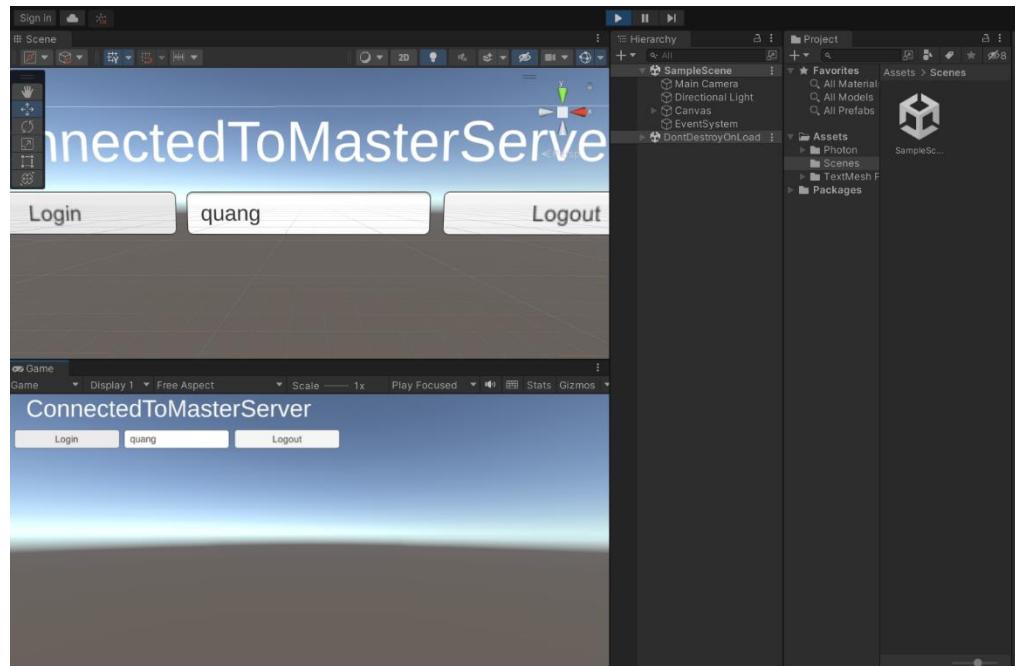
**Bước 5: Kết nối với Photon Server:** Ta cần kết nối ứng dụng của mình với Photon Server để có thể gửi và nhận dữ liệu. Các thông tin kết nối này

được cung cấp khi tạo ứng dụng. Để kết nối ta paste mã App ID đã copy ở bước trên vào trong panel PUN Setup như trong hình



Hình 3.1. 5: Paste AppId đã copy vào để setup

**Bước 6: Xây dựng trò chơi:** Sau khi kết nối thành công, có thể bắt đầu xây dựng trò chơi của mình. Các chức năng như xử lý đăng nhập, kết nối, gửi và nhận dữ liệu từ các người chơi khác đều được cung cấp bởi SDK của Photon Engine.



Hình 3.1. 6: Kiểm tra xem đã kết nối

## II. Triển khai game

### 1. Giao diện Splash screen

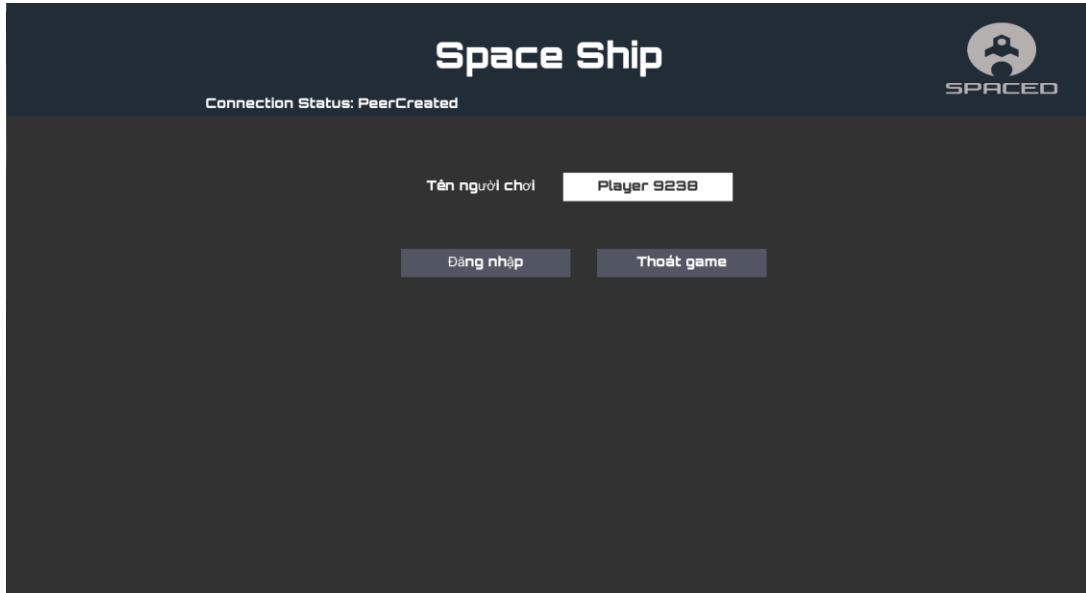


Hình 3.2. 1: Splash screen

Đây là màn hình splash của game, màn này nhằm mục đích là để tạo ra một ấn tượng đầu tiên game đối với người chơi. Nó giúp cho người chơi có thể nhận ra được game ngay khi khởi động và tạo ra một cảm giác chuyên nghiệp và đáng tin cậy với người chơi.

Ngoài ra, màn hình splash cũng giúp cho game được tải lên và khởi động dần dần một cách trơn tru hơn. Nó cho phép cho các tài nguyên và dữ liệu của ứng dụng hoặc game được tải lên trong nền mà không ảnh hưởng đến trải nghiệm của người chơi.

### 1. Màn hình đăng nhập



Hình 3.2. 2: Giao diện màn đăng nhập

Button	Mục đích
Đăng nhập	Khi người chơi click sẽ tiến hành đăng nhập với server game
Thoát game	Thoát khỏi trò chơi

Bảng 3.2. 1: Các button giao diện đăng nhập

Đây là màn hình đăng nhập của game, nó bao gồm các chức năng chính sau:

- Nhập tên người chơi: Trên giao diện có một ô `inputText` nhằm cho phép người chơi nhập vào tên của mình trong suốt quá trình chơi game. Khi màn đăng nhập được khởi động mặc định tên người chơi sẽ được hệ thống tạo ra ngẫu nhiên và nếu không thích nó người dùng có thể hoàn toàn thay đổi.
- Đăng nhập: Khi người chơi click vào button này, người chơi sẽ được tiến hành đăng nhập vào với server game. Và người chơi sẽ đăng nhập với tên lấy từ ô `inputText`.

```

public void OnLoginButtonClicked()
{
    string playerName =
    PlayerNameInput.text;

    if (!playerName.Equals(""))
    {
        PhotonNetwork.LocalPlayer.NickName =
        playerName;

        StartCoroutine(ConnectPhoton());
    }
    else
    {
        Debug.LogError("Player Name is
invalid.");
    }
}

```

Đoạn code trên là phần xử lý đăng nhập của người chơi. Đầu tiên biến playerName sẽ nhận vào tên người dùng từ ô inputText sau đó tiến hành check equal cho biến playerName, nếu biến đó không trống thì tiến hành đăng nhập bằng cách setup NickName người chơi bằng với giá trị trong biến playerName sau đó gọi đến hàm ConnectPhoton()

PhotonNetwork.LocalPlayer.NickName = playerName;

StartCoroutine(ConnectPhoton());

- Thoát game: Đây là button sẽ giúp người chơi thoát khỏi trò chơi chỉ bằng 1 lần click

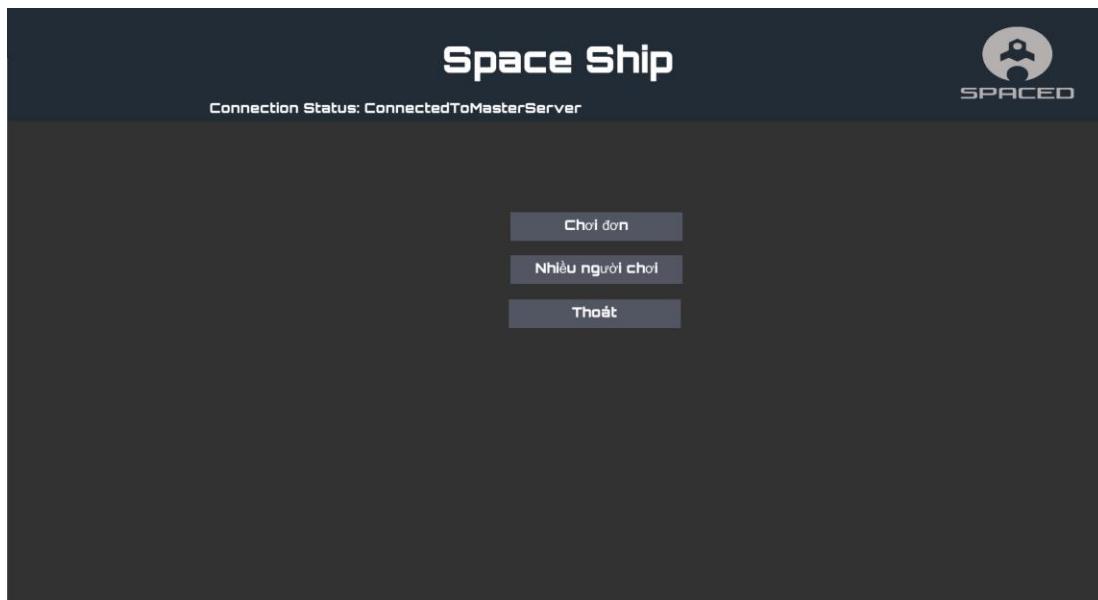
```

public void OnExitGameButtonClicked()
{
    Application.Quit();
    Debug.Log("thoát game");
}

```

Application.Quit() là một hàm hỗ trợ từ unity để thoát game đang chạy

## 2. Giao diện chọn chế độ



Hình 3.2. 3: Giao diện chọn chế độ chơi

Button	Mục đích
Chơi đơn	Đưa người chơi vào chế độ chơi đơn
Nhiều người chơi	Đưa người chơi vào chế độ chơi online với nhiều người
Thoát	Thoát về màn login

Bảng 3.2. 2: Các button màn hình chọn chế độ chơi

Đây là màn hình chọn chế độ chơi của game, nó có các chức năng chính như sau:

- Chơi đơn: Khi người click vào button này, người chơi sẽ được đưa đến màn menu của chế độ chơi đơn.

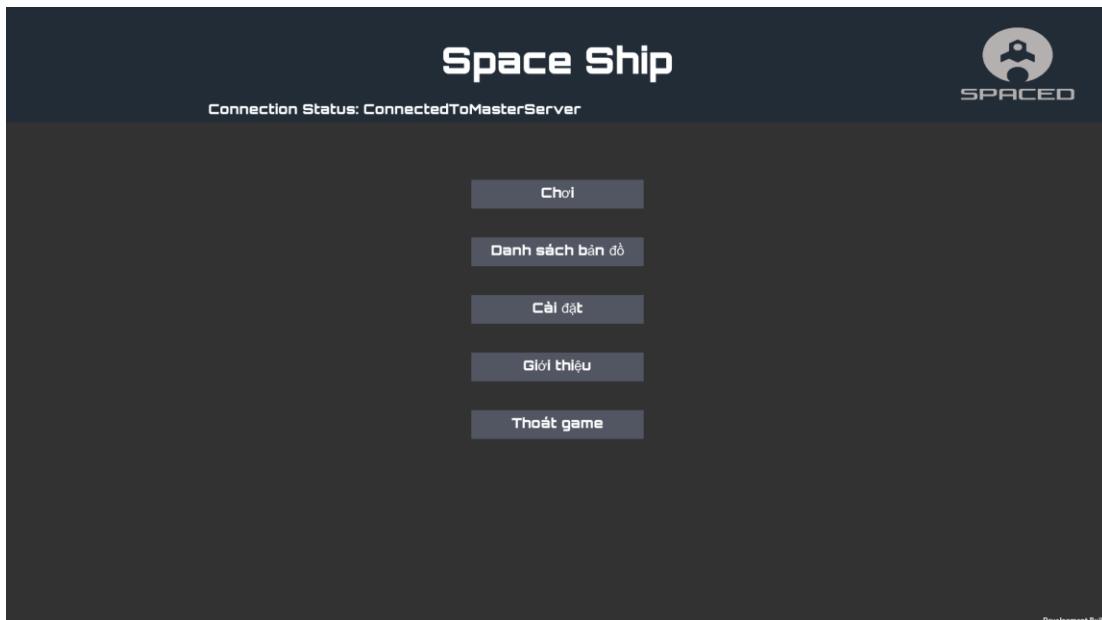
```
public void OnSingleButtonClicked()
{
    single=true;
    PlayerPrefs.SetInt("SingleScene", 1);
    SetActivePanel(SinglePanel.name);
}
```

Để xử lý việc đưa người chơi di chuyển qua các màn thì ở đây nhóm làm bằng các set active cho các màn theo tên của màn đó được truyền vào hàm SetActivePanel(), hàm này do nhóm tự định nghĩa. Biến single và PlayerPrefs.SetInt dùng cho việc xử lý phần back lại màn sau này.

- Nhiều người chơi: Khi người click vào button này, người chơi sẽ được đưa đến màn menu của chế độ nhiều người chơi. Và việc xử lý phần click vào button cũng giống với việc xử lý với button chơi đơn.

```
public void OnMultiButtonClicked()
{
    single = false;
    PlayerPrefs.SetInt("SingleScene", 0);
    SetActivePanel(SelectionPanel.name);
}
```

### 3. Giao diện chế độ chơi đơn



Hình 3.2. 4: Giao diện chế độ 1 người chơi

Button	Mục đích
Chơi	Đưa người chơi vào trong giao

	diện game
Danh sách bản đồ	Đưa người chơi vào chọn bản đồ
Cài đặt	Đưa người chơi vào cài đặt trong game
Giới thiệu	Đưa người chơi vào màn giới thiệu về game như cốt truyện, người phát triển
Thoát game	Thoát về màn chọn chế độ chơi

Bảng 3.2. 3: Các button giao diện chế độ chơi đơn

Đây là giao diện menu của chế độ chơi đơn, nó có các chức năng chính sau:

- Chơi: Khi click vào button này người chơi sẽ được đưa đến phòng chơi của chế độ chơi đơn.

```
public void OnButtonSinglePlayerClicked()
{
    string roomName = "Room " +
Random.Range(1000, 10000);

    byte maxPlayers = 1;

    // thiết lập phòng: số lượng người, thời
    // gian giới hạn cho người chơi
    RoomOptions options = new RoomOptions {
        MaxPlayers = maxPlayers, PlayerTtl = 10000 };

    PhotonNetwork.CreateRoom(roomName,
options, null);
}
```

Vì là chơi đơn nên phòng sẽ có `maxPlayers = 1` và tên phòng sẽ được tạo random. Sau đó một đối tượng "RoomOptions" được khởi tạo và được thiết lập với các thông số của phòng chơi như số lượng tối đa người chơi và thời gian sống của người chơi (PlayerTtl). Cuối cùng, phương thức "PhotonNetwork.CreateRoom()" được gọi với các đối số tương ứng là tên phòng, các tùy chọn của phòng và một

- đối tượng "TypedLobby" (trong trường hợp này không được sử dụng đến). Phương thức này được sử dụng để tạo ra một phòng chơi mới với các thông số được thiết lập trong đối tượng "RoomOptions".
- Danh sách bản đồ: Khi click vào button này người chơi sẽ được đưa đến screen của chọn bản đồ chơi.

```
public void OnMapListButtonClicked()
{
    if (!PhotonNetwork.InLobby)
    {
        PhotonNetwork.JoinLobby();
    }

    SetActivePanel(MapListPanel.name);
}
```

- Để xử lý thì nó cũng giống như các button chuyển màn khác là sử dụng hàm SetActivePanel() và truyền vào đó tên của scene muốn chuyển đến và điều kiện trước khi chuyển là người chơi phải đang trong lobby menu vậy nên mới có việc check ở hàm if().
- Cài đặt: Khi click vào button này người chơi sẽ được chuyển đến scene cài đặt của game. Việc xử lý click chuyển scene thì ở đây cũng dùng hàm SetActivePanel() và truyền tên scene vào.

```
public void OnButtonSettingClicked()
{
    SetActivePanel(MusicSettingPanel.name);
}
```

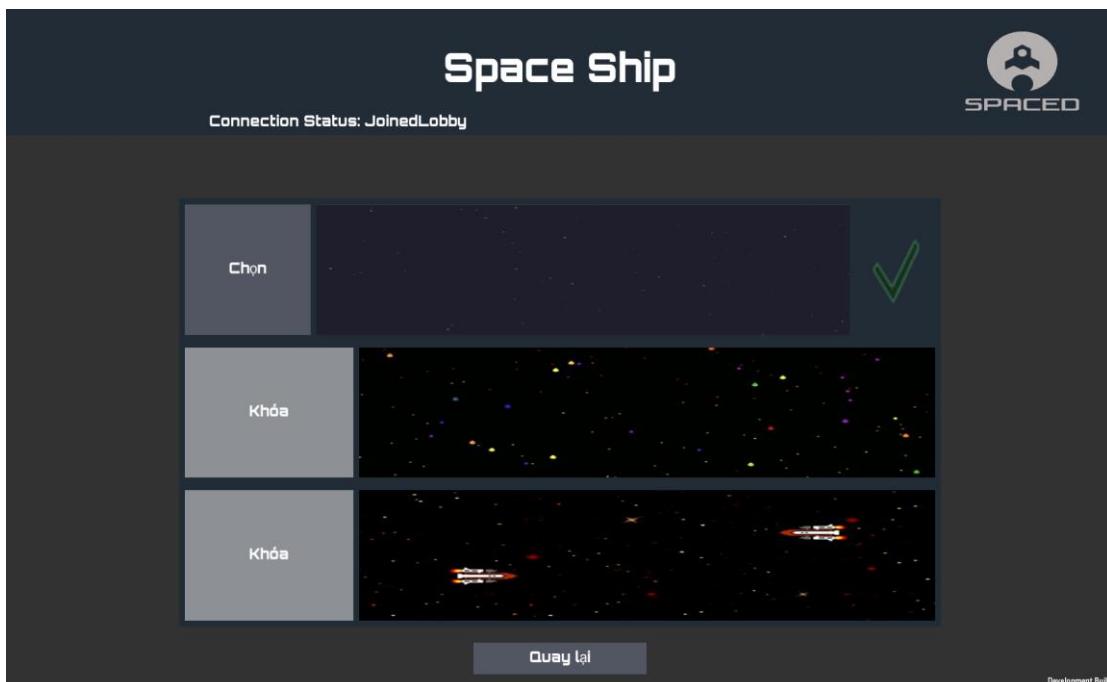
- Giới thiệu: Khi click người chơi sẽ được đưa đến scene giới thiệu về game. Xử lý click cũng giống với việc xử lý click cài đặt ở trên.

```
public void OnAboutButtonClicked()
{
    SetActivePanel(AboutPanel.name);
}
```

- Thoát: Khi click người chơi sẽ quay trở lại màn chọn chế độ chơi game.

```
public void OnBackMode()
{
    SetActivePanel(ModePanel.name);
}
```

#### 4. Giao diện chọn bản đồ



Hình 3.2. 5: Giao diện chọn bản đồ

Button	Mục đích
Chọn	Để người chơi chọn map
Khóa	Để cho người chơi biết map chưa đủ điều kiện để mở
Quay lại	Thoát về giao diện của người chơi đơn

Bảng 3.2. 4: Các button giao diện chọn bản đồ

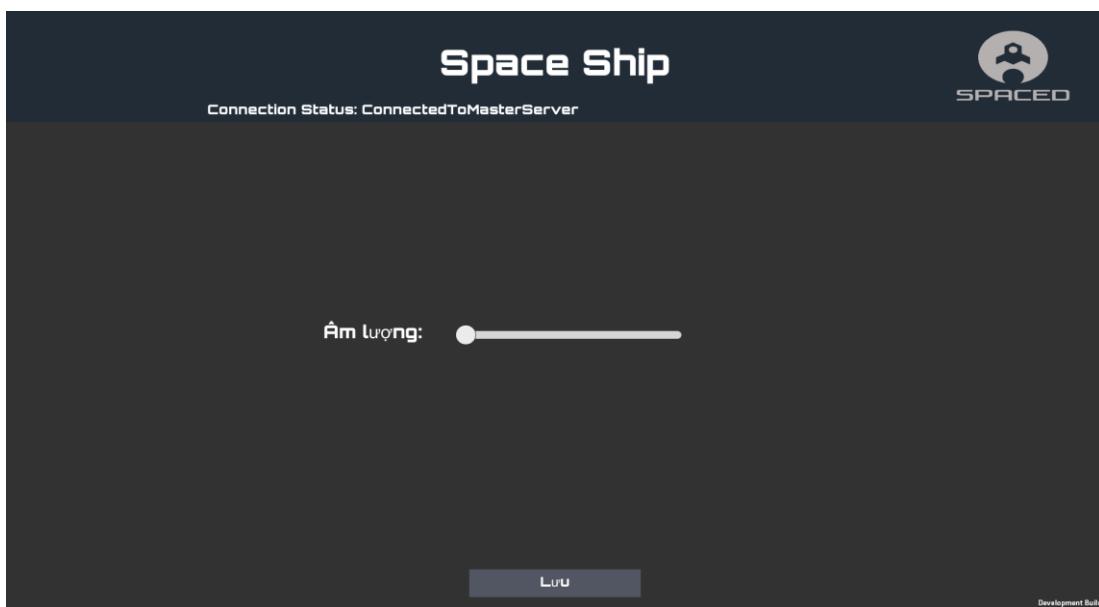
Đây là giao diện chọn bản đồ chơi, có các chức năng chính sau:

- Chọn: Người chơi chỉ được click chọn những map nào không bị khóa và sau khi click thì sẽ xuất hiện dấu tick xanh bên cạnh map đã chọn. Và phần chọn map này đang được nhóm nên kế hoạch phát triển vào các phần cập nhật sau.
- Quay lại: Người chơi sẽ được quay lại menu khi click

```
public void OnBackButtonClicked()
{
    // nguoi choi trong room
    if (PhotonNetwork.InLobby)
    {
        PhotonNetwork.LeaveLobby();
    }
    // nguoi choi khong o trong room
    SetActivePanel(single == true ?
SinglePanel.name : SelectionPanel.name);
}
```

Vì game có 2 chế độ chơi nên sẽ có 2 main menu cho 2 chế độ và khi back sẽ cần phải check theo điều kiện single(chế độ chơi đơn).

## 5. Giao diện cài đặt



Hình 3.2. 6: Giao diện cài đặt

Button	Mục đích
--------	----------

Kéo âm lượng	Tăng giảm âm lượng
Lưu	Ghi lại chỉnh sửa vào dữ liệu và về giao diện của người chơi đơn

Bảng 3.2. 5: Các button giao diện cài đặt

Giao diện scene cài đặt có các chức năng chính sau:

- Kéo âm lượng: Khi người dùng kéo hoặc giảm thanh âm lượng thì đang là cài đặt mức âm lượng cho game.
- Save: Khi click giá trị của âm lượng sẽ được lưu

```

public void OnSaveButtonClicked()
{
    float volumeValue = musicSlider.value;
    PlayerPrefs.SetFloat("VolumeValue",
volumeValue);
    LoadValues();
    if (PhotonNetwork.InLobby)
    {
        PhotonNetwork.LeaveLobby();
    }
    // nguoi choi khong o trong room
    SetActivePanel(single == true ?
SinglePanel.name : SelectionPanel.name);

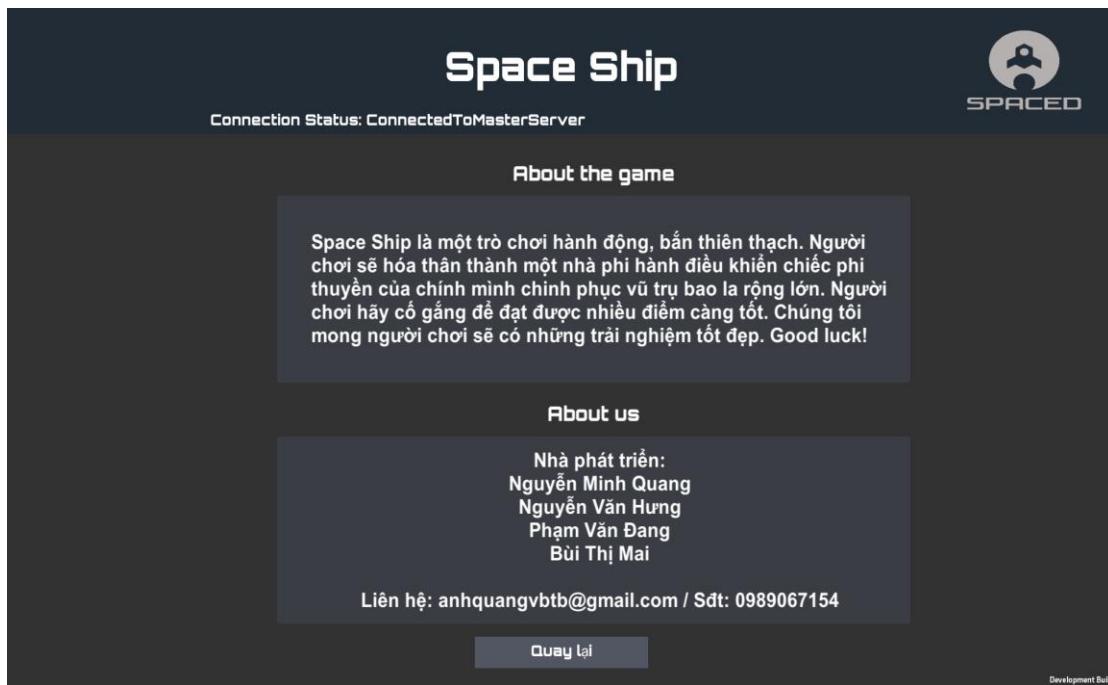
}
void LoadValues()
{
    float volumeValue =
PlayerPrefs.GetFloat("VolumeValue");
    musicSlider.value = volumeValue;

}

```

Đầu tiên biến volumeValue = musicSlider.value; đang nhận giá trị từ slider musicSliders và lưu vào trong local store sau đó sẽ gọi hàm LoadValue() để tiến hành cập nhật giá trị mới cho slider. Cuối cùng là đưa người chơi quay lại menu bằng SetActivePanel().

## 6. Giao diện giới thiệu game



Hình 3.2. 7: Giao diện giới thiệu game

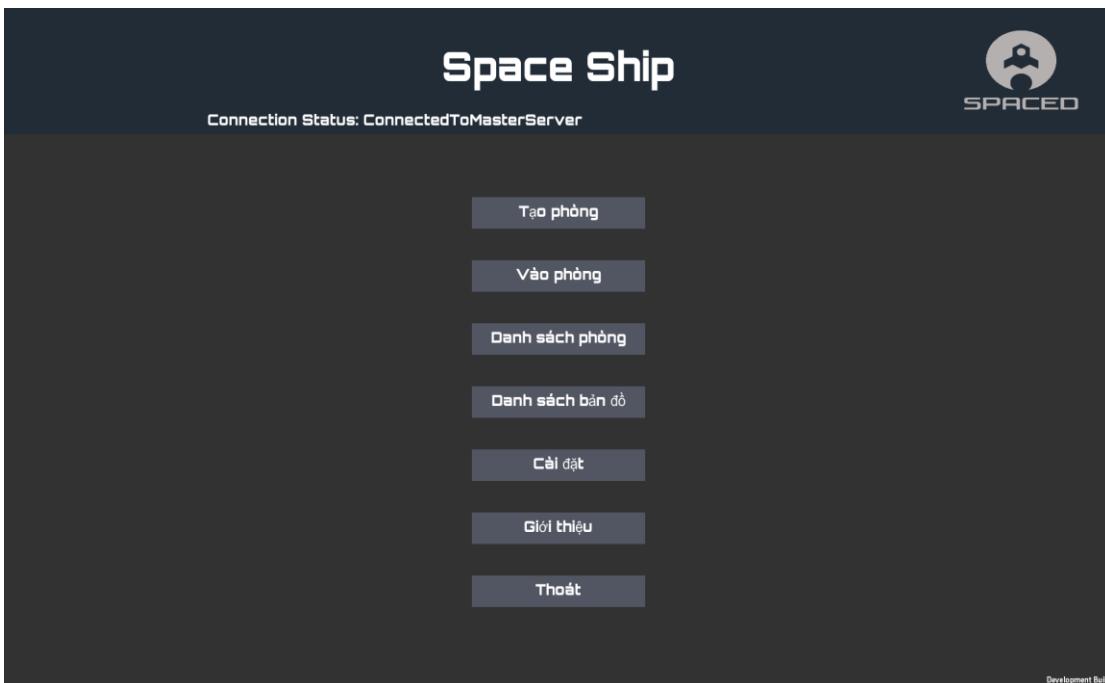
Button	Mục đích
Quay lại	Để người chơi về giao diện của người chơi đơn

Bảng 3.2. 6: Các button giao diện giới thiệu game

Scene giới thiệu về game và nhóm phát triển

- Quay lại: Đưa người chơi quay lại menu

## 7. Giao diện chế độ nhiều người chơi



Hình 3.2. 8: Giao diện chế độ đa người chơi

Button	Mục đích
Tạo phòng	Đưa người chơi tạo phòng để người chơi khác có thể tham gia
Vào phòng	Đưa người chơi vào 1 phòng chơi bất kì nào đó
Danh sách phòng	Đưa người chơi vào chọn phòng
Danh sách bản đồ	Đưa người chơi vào chọn bản đồ
Cài đặt	Đưa người chơi vào cài đặt trong game
Giới thiệu	Đưa người chơi vào màn giới thiệu về game như cốt truyện, người phát triển
Thoát game	Thoát về màn chọn chế độ chơi

Bảng 3.2. 7: Các button giao diện chế độ đa người chơi

Đây là giao diện menu của chế độ nhiều người chơi, có các chức năng chính sau:

- Tạo phòng: Khi người chơi click sẽ đưa người chơi đến scen tạo phòng, việc xử lý click thì cũng sẽ dùng SetActivePanel() và truyền vào đó tên của scene muốn chuyển.
- Vào phòng: Khi click sẽ đưa người chơi vào một phòng bất kỳ nào đó
- Danh sách phòng: Khi click sẽ đưa người chơi vào scene hiển thị danh sách các phòng hiện có
- Các chức năng còn lại như danh sách bản đồ, cài đặt, giới thiệu, thoát game đều giống với các chức năng của menu của chế độ chơi đơn.
- Source code xử lý việc chuyển scene

```
public void SetActivePanel(string activePanel)
{
    LoginPanel.SetActive(activePanel.Equals(LoginPanel.name));

    SelectionPanel.SetActive(activePanel.Equals(SelectionPanel.name));

    CreateRoomPanel.SetActive(activePanel.Equals(CreateRoomPanel.name));

    JoinRandomRoomPanel.SetActive(activePanel.Equals(JoinRandomRoomPanel.name));

    RoomListPanel.SetActive(activePanel.Equals(RoomListPanel.name));

    InsideRoomPanel.SetActive(activePanel.Equals(InsideRoomPanel.name));

    MapListPanel.SetActive(activePanel.Equals(MapListPanel.name));

    ModePanel.SetActive(activePanel.Equals(ModePanel.name));
}
```

```

.name));

SinglePanel.SetActive(activePanel.Equals(SinglePanel.name));

LoaderUI.SetActive(activePanel.Equals(LoaderUI.name));

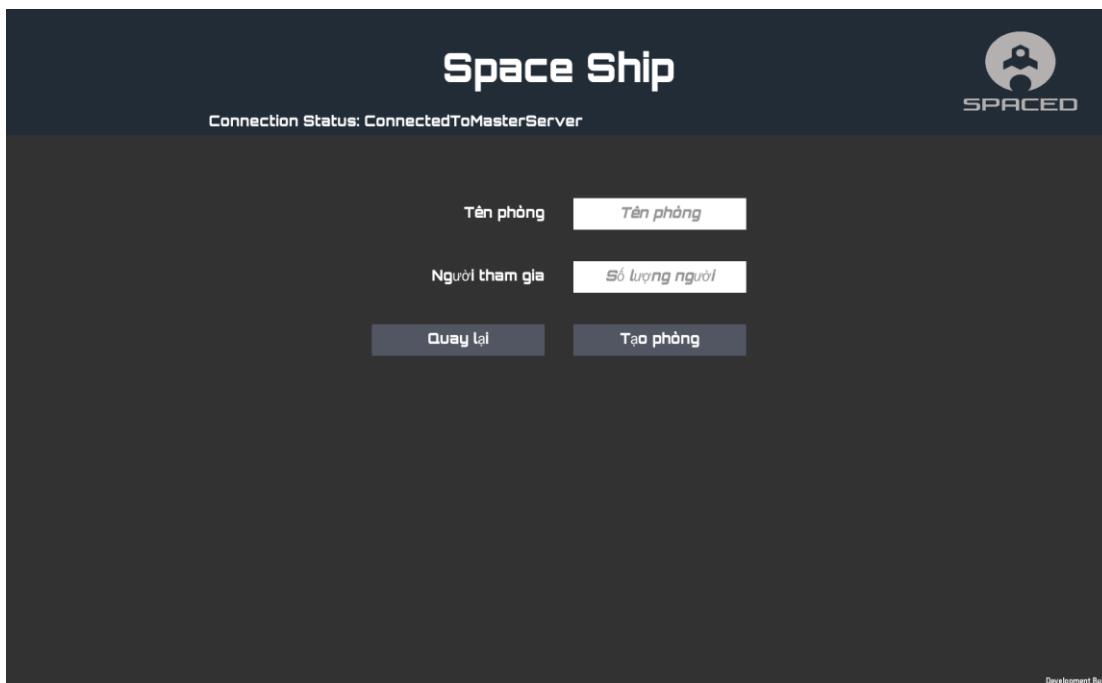
MusicSettingPanel.SetActive(activePanel.Equals(MusicSettingPanel.name));

AboutPanel.SetActive(activePanel.Equals(AboutPanel.name));
}

```

Vì hàm SetActivePanel() để public nên ta có thể truyền tên scene muốn chuyển đến từ của sổ inspector của Unity hoặc trong code thì gọi hàm và truyền tên vào.

## 8. Giao diện tạo phòng



Hình 3.2. 9: Giao diện tạo phòng

TextInput	Mục đích
Tên phòng	Tạo tên phòng để người chơi khác để phân biệt với các phòng
Số lượng người	Tạo giới hạn về lượng người trong phòng

Bảng 3.2. 8: Các text input giao diện tạo phòng

Button	Mục đích
Quay lại	Để người chơi trở về giao diện của nhiều người chơi
Tạo phòng	Dưa người chơi đến giao diện phòng chờ người chơi khác

Bảng 3.2. 9: Các button giao diện tạo phòng

- Tên phòng: Cho phép người chơi nhập tên phòng muốn đặt, nhằm  
giúp bạn bè dễ dàng tìm kiếm nhau và vào cùng 1 phòng.
- Số lượng người: Cho phép người chơi nhập số lượng cụ thể mong  
muốn số lượng người trong 1 phòng, giới hạn max là 8 người.
- Tạo phòng: Khi click hệ thống sẽ tiến hành tạo phòng theo những gì  
người dùng đã cài đặt về tên phòng và số lượng player

```
public void OnCreateRoomButtonClicked()
{
    string roomName =
    RoomNameInputField.text;
    roomName =
    (roomName.Equals(string.Empty)) ? "Room " +
    Random.Range(1000, 10000) : roomName;

    byte maxPlayers;
    byte.TryParse(MaxPlayersInputField.text,
    out maxPlayers);
    maxPlayers =
```

```

(byte)Mathf.Clamp(maxPlayers, 2, 8);

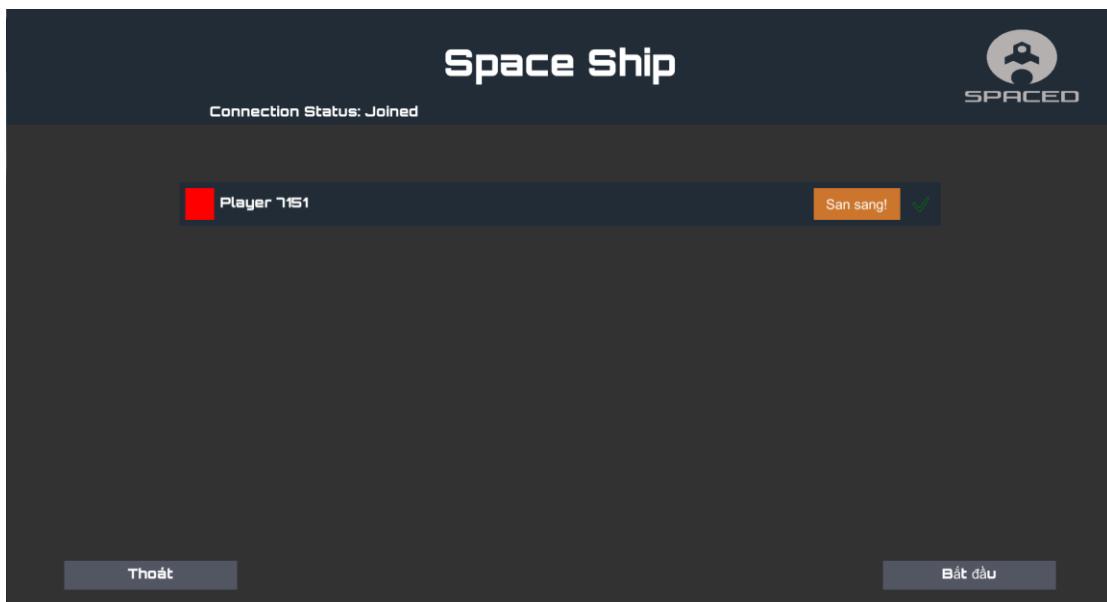
        // thiet lap phong: so luong nguoi, thoi
        // gian gioi han cho nguoi choi
        RoomOptions options = new RoomOptions {
            MaxPlayers = maxPlayers, PlayerTtl = 10000 };

            PhotonNetwork.CreateRoom(roomName,
options, null);
}

```

Nếu trong trường hợp người chơi không nhập tên phòng thì tên phòng sẽ được tạo ngẫu nhiên roomName = (roomName.Equals(string.Empty)) ? "Room " + Random.Range(1000, 10000) : roomName; và số lượng maxPlayer sẽ giữ nguyên nếu giá trị nhập vào trong khoảng từ 2 đến 8, nếu nhỏ hơn 2 sẽ là 2 và nếu quá 8 thì sẽ là 8. Sau đó tiến hành tạo phòng với PhotonNetwork.CreateRoom() đầu vào là option của room và tên room. Giá trị PlayerTtl chính là thời gian tồn tại của người chơi trong phòng sau khi người đó thoát ra ngoài room.

## 9. Giao diện phòng



Hình 3.2. 10: Giao diện trong phòng chờ chơi game

Button	Mục đích
Sẵn sàng	Người chơi và chế độ đồng ý vào trò chơi
Bắt đầu	Khi tất cả người chơi sẵn sàng thì button này sẽ hiện ra với phía của người chủ phòng để bắt đầu vào game
Thoát	Đưa người chơi về giao diện của nhiều người chơi

Bảng 3.2. 10: Các button giao diện phòng chờ game

Giao diện phòng có các chức năng chính sau:

- Sẵn sàng: Khi click button này đồng nghĩa với việc người chơi đã sẵn sàng tham gia game.

```

public void Start()
{
    // check chu phong
    if
        (PhotonNetwork.LocalPlayer.ActorNumber != ownerId)
    {

        PlayerReadyButton.gameObject.SetActive(false);
    }
    else
    {
        Hashtable initialProps = new
        Hashtable() { { SpaceShip.PLAYER_READY,
        isPlayerReady }, { SpaceShip.PLAYER_LIVES,
        SpaceShip.PLAYER_MAX_LIVES } };
        //set thuoc tinh cho ng chs

        PhotonNetwork.LocalPlayer.SetCustomProperties(in
        itialProps);
    }
}

```

```

PhotonNetwork.LocalPlayer.SetScore(0);
    // xu ly sk click san sang

PlayerReadyButton.onClick.AddListener(() =>
{
    isPlayerReady = !isPlayerReady;
    SetPlayerReady(isPlayerReady);

    Hashtable props = new
Hashtable() { { SpaceShip.PLAYER_READY,
isPlayerReady } };

PhotonNetwork.LocalPlayer.SetCustomProperties(pr
ops);

    if
(PhotonNetwork.IsMasterClient)
{
    FindObjectOfType<LobbyBodyPanel>().LocalPlayerPr
opertiesUpdated();
    }
    });
}
}

```

Trong đoạn code trên đầu tiên sẽ kiểm tra nếu người chơi hiện tại (tức là người chơi đang thực thi đoạn mã này) không phải là chủ sở hữu của phòng chơi (được xác định bằng cách so sánh ActorNumber của LocalPlayer với ownerId), thì nút "PlayerReadyButton" sẽ bị vô hiệu hóa bằng cách gọi phương thức SetActive(false) của đối tượng PlayerReadyButton. Ngược lại, nếu người chơi hiện tại là chủ sở hữu của phòng chơi, đoạn mã tiếp theo sẽ được thực thi. Một đối tượng Hashtable được khởi tạo với hai cặp key-value: PLAYER\_READY và isPlayerReady (một biến boolean), và PLAYER\_LIVES và PLAYER\_MAX\_LIVES (hai hằng số được định nghĩa trước đó).

- trong lớp SpaceShip) để set thuộc tính cho người chơi. Và sau đó là đặt giá trị điểm số ban đầu của người chơi =0, tiếp theo đó là phần xử lý hành động click button sẵn sàng bằng phương thức onClick.AddListener() được sử dụng để đăng ký một sự kiện click cho nút "PlayerReadyButton". Khi người chơi click vào nút này, biến boolean isPlayerReady sẽ được đảo ngược (từ true thành false hoặc ngược lại), và các thuộc tính tùy chỉnh của LocalPlayer cũng sẽ được cập nhật. Nếu người chơi hiện tại là chủ sở hữu của phòng chơi, phương thức LocalPlayerPropertiesUpdated() của đối tượng LobbyBodyPanel sẽ được gọi để cập nhật thông tin người chơi.
- Bắt đầu: Khi tất cả người chơi đã sẵn sàng thì chủ phòng sẽ tiến hành click button này để chuyển tất cả người chơi trong room sang scene chơi game.

```
public void OnStartGameButtonClicked()
{
    PhotonNetwork.CurrentRoom.isOpen =
false;
    PhotonNetwork.CurrentRoom.isVisible =
false;

    PhotonNetwork.LoadLevel("Game Scene");
}
```

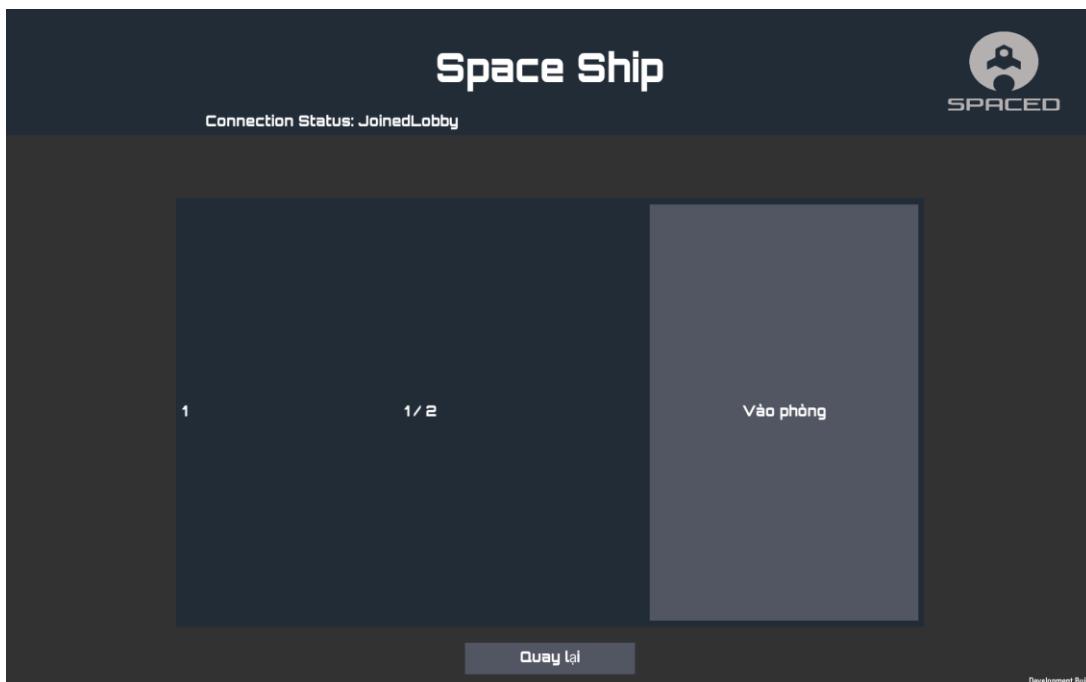
- Khi bắt đầu thì trước tiên phòng chơi sẽ bị đóng không cho ai tham gia nữa bằng hàm isOpen = false và cũng không hiển thị trong danh sách chờ bằng hàm isVisible = false. Sau đó là sẽ tiến hành chuyển scene bằng LoadLevel().
- Thoát: Người chơi sẽ được đưa trở lại menu scene của chế độ nhiều người chơi.

```
public void OnLeaveGameButtonClicked()
{
    PhotonNetwork.LeaveRoom();
    SetActivePanel(single == true ?
SinglePanel.name : SelectionPanel.name);
```

}

Để rời khỏi phòng thì ở đây Photon hỗ trợ người dùng phương thức LeaveRoom() sau đó người chơi được đưa ra ngoài menu scene bằng SetActivePanel().

## 10. Giao diện danh sách phòng



Hình 3.2. 11: Giao diện danh sách phòng

Button	Mục đích
Vào phòng	Đưa người chơi vào phòng chờ trò chơi được người chơi khác tạo
Quay lại	Đưa người chơi về giao diện của nhiều người chơi

Bảng 3.2. 11: Các button giao diện danh sách phòng

Trong giao diện danh sách phòng có những chức năng chính sau:

- Hiển thị danh sách phòng: Người chơi sẽ thấy trong scene là một list các danh sách phòng để có thể lựa chọn tham gia. Ở mỗi phòng đều có hiển thị tên, số lượng người để người chơi tiện theo dõi.
- Tham gia: Click vào button này người chơi sẽ được tham gia vào phòng tương ứng

```

JoinRoomButton.onClick.AddListener(() =>
{
    if (PhotonNetwork.InLobby)
    {
        PhotonNetwork.LeaveLobby();
    }

    PhotonNetwork.JoinRoom(roomName);
});

```

Khi click thì phương thức AddListener() được gọi để lắng nghe sự kiện và để tham gia vào phòng thì điều kiện là người chơi phải thoát khỏi lobby(menu scene) sau đó thực hiện tham gia phòng bằng phương thức JoinRoom() và truyền vào đó tên phòng.

- Quay lại: Người chơi sẽ được đưa ra khỏi giao diện danh sách phòng và quay trở lại với menu scene

```

public void OnBackButtonClicked()
{
    // nguoi choi trong room
    if (PhotonNetwork.InLobby)
    {
        PhotonNetwork.LeaveLobby();
    }
    // nguoi choi khong o trong room
    SetActivePanel(single == true ?
SinglePanel.name : SelectionPanel.name);
}

```

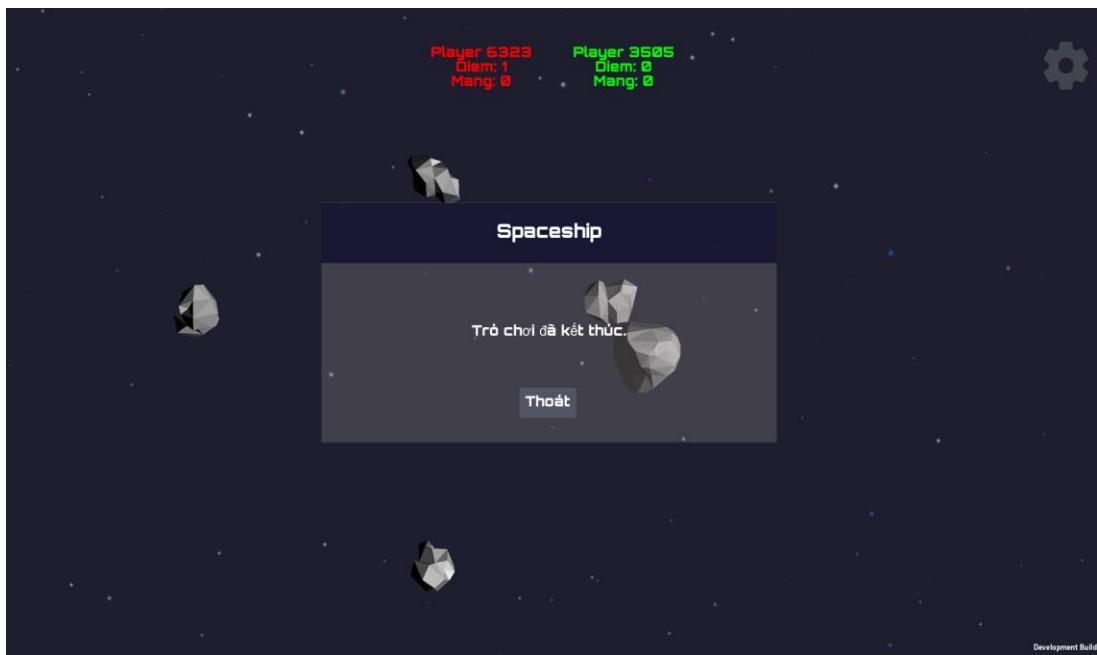
## 11. Giao diện trò chơi



Hình 3.2. 12: Giao diện chơi game



Hình 3.2. 13: Giao diện thông kê khi va chạm



Hình 3.2. 14: Giao diện kết thúc trò chơi

Button	Mục đích
Icon cài đặt	Đưa người chơi vào cài đặt trong trò chơi
Thoát	Đưa người chơi về giao diện chọn chế độ

Bảng 3.2. 12: Các button trong giao diện chơi game

Giao diện chơi game có các chức năng chính sau:

- Cài đặt: Click vào thì một cửa sổ sẽ hiện ra cho phép người chơi cài đặt âm lượng và thoát khỏi phòng chơi.
- Thoát: Cửa sổ này sẽ hiện khi trò chơi kết thúc, người chơi click và sẽ đưa người chơi trở lại với giao diện chọn chế độ chơi
- Hiển thị điểm và trạng thái: Trên giao diện chơi game sẽ có phần hiển thị điểm của từng người chơi trong phòng tương ứng với màu sắc của phi thuyền người chơi.

```
// Khởi tạo
playerListEntries = new Dictionary<int,
GameObject>();

foreach (Player p in
```

```

PhotonNetwork.PlayerList)
{
    GameObject entry =
Instantiate(PlayerOverviewEntryPrefab);

entry.transform.SetParent(gameObject.transform);
    entry.transform.localScale =
Vector3.one;
    entry.GetComponent<Text>().color =
SpaceShip.GetColor(p.GetPlayerNumber());
    entry.GetComponent<Text>().text =
string.Format("{0}\nDiem: {1}\nMang: {2}",
p.NickName, p.GetScore(),
SpaceShip.PLAYER_MAX_LIVES);

    playerListEntries.Add(p.ActorNumber,
entry);
}
// Cập nhật
public override void
OnPlayerPropertiesUpdate(Player targetPlayer,
Hashtable changedProps)
{
    GameObject entry;
    if
(playerListEntries.TryGetValue(targetPlayer.ActorNumber, out entry))
    {
        entry.GetComponent<Text>().text =
string.Format("{0}\nDiem: {1}\nMang: {2}",
targetPlayer.NickName, targetPlayer.GetScore(),
targetPlayer.CustomProperties[SpaceShip.PLAYER_LIVES]);
    }
}

```

Số lượng người chơi sẽ quyết định có bao nhiêu số lượng trạng thái được hiển thị foreach (Player p in PhotonNetwork.PlayerList). Sau đó chúng sẽ được đổi màu theo màu của người chơi tương ứng

```
entry.GetComponent<Text>().color =
SpaceShip.GetColor(p.GetPlayerNumber());
và khởi tạo giá trị ban đầu
entry.GetComponent<Text>().text =
string.Format("{0}\nĐiem: {1}\nMang: {2}",
p.NickName, p.GetScore(),
SpaceShip.PLAYER_MAX_LIVES);
```

Quá trình cập nhật điểm đầu tiên, một đối tượng GameObject được khởi tạo với giá trị tương ứng với ActorNumber của targetPlayer trong từ điển "playerListEntries" (được khởi tạo bên ngoài phương thức này).

Sau đó, đối tượng Text của đối tượng được lưu trữ trong entry được sử dụng để cập nhật thông tin của người chơi. Cụ thể, định dạng chuỗi được sử dụng để hiển thị thông tin bao gồm tên người chơi (NickName), điểm số (Score) và số mạng còn lại (PLAYER\_LIVES) của người chơi. Thông tin này được lấy từ targetPlayer thông qua các phương thức GetScore() và phương thức CustomProperties[] để truy cập thuộc tính tùy chỉnh PLAYER\_LIVES của người chơi.

Cuối cùng, thông tin được cập nhật vào đối tượng Text của entry để hiển thị thông tin mới của người chơi

- Khi trò chơi kết thúc sẽ có phần hiển thị người chiến thắng và số điểm của người chiến thắng.

```
private IEnumerator EndOfGame(string winner,
int score)
{
    float timer = 5.0f;
    /*
    if (score >
highScoreManager.GetHighScore())
    {
```

```

highScoreManager.SaveHighScore(score, winner);
    } */
    while (timer > 0.0f)
    {
        InfoText.text = string.Format("Nguoi
        choi {0} chien thang voi {1} diem.\n\n\nTro lai
        giao dien dang nhap sau {2} giay.", winner,
        score, timer.ToString("n2"));

        yield return new
WaitForEndOfFrame();

        timer -= Time.deltaTime;
    }

//PhotonNetwork.LeaveRoom();
Popup.SetActive(true);
InfoText.text = string.Format("");
}

```

Để hiển thị người chiến thắng và số điểm của người chiến thắng ở đây cần phải sử dụng 1 hàm xử lý bất đồng bộ, và đoạn text hiển thị sẽ được duy trì 5s hết 5s sẽ hiển thị popup thoát game.

- Xử lý di chuyển của phi thuyền: Trong trò chơi người chơi sẽ điều khiển phi thuyền quay theo các hướng và cung cấp cho nó 1 lực đẩy và lực đẩy này sẽ duy trì mãi để phi thuyền có cảm giác như luôn trôi nổi trong không gian.

```

if (!photonView.IsMine)
{
    return;
}

if (!controllable)
{
    return;
}

```

```

    }

    Quaternion rot = rigidbody.rotation *
Quaternion.Euler(0, rotation * RotationSpeed *
Time.fixedDeltaTime, 0);
    rigidbody.MoveRotation(rot);

    Vector3 force = (rot * Vector3.forward)
* acceleration * 1000.0f * MovementSpeed *
Time.fixedDeltaTime;
    rigidbody.AddForce(force);

    if (rigidbody.velocity.magnitude >
(MaxSpeed * 200.0f))
    {
        rigidbody.velocity =
rigidbody.velocity.normalized * MaxSpeed *
200.0f;
    }

    CheckExitScreen();
}

```

Về điều chỉnh hướng quay thì ở đây nhóm nghiên cứu và sử dụng `Quaternion.Euler()` nhằm thực hiện việc quay sẽ trở trên mượt mà và dễ dàng hơn vì hiện nay phi thuyền chỉ đang quay với trục y trong không gian 3 chiều. Về phần di chuyển của phi thuyền thì ở đây nhóm xử lý bằng cách cung cấp cho phi thuyền 1 lực force bằng hàm `AddForce()` nhằm giữ phi thuyền tự động trôi nổi trên bề mặt không gian. Hướng đi mới của đối tượng Game Object được tính toán bằng cách nhân hướng quay mới (`rot`) với hướng đi mặc định của đối tượng (`Vector3.forward`). Điều này đảm bảo rằng đối tượng sẽ di chuyển theo hướng nào được chỉ định bởi hướng quay mới. Gia tốc (`acceleration`) được nhân vào hướng đi mới để đảm bảo rằng đối tượng sẽ di chuyển với một mức độ gia tốc nhất định. Gia tốc càng lớn, đối tượng sẽ di chuyển nhanh hơn. Hằng số `1000.0f` được sử dụng để đổi đơn vị từ đơn vị  $m/s^2$  sang đơn vị N (Newton). Điều này đảm bảo rằng lực đẩy tính toán được sẽ có đơn vị N, đơn vị lực

phổ biến trong vật lý. Tốc độ di chuyển của đối tượng (MovementSpeed) được nhân vào lực đẩy để đảm bảo rằng đối tượng sẽ di chuyển với một tốc độ nhất định. Tốc độ càng lớn, đối tượng sẽ di chuyển nhanh hơn. Thời gian đã trôi qua (Time.fixedDeltaTime) được nhân vào lực đẩy để tính toán lực đẩy trong khoảng thời gian đã trôi qua. Điều này đảm bảo rằng lực đẩy sẽ được áp dụng một cách liên tục và mượt mà cho đối tượng Game Object.

- Xử lý khi phi thuyền đi ra khỏi màn hình:

```
private void CheckExitScreen()
{
    if (Camera.main == null)
    {
        return;
    }

    if (Mathf.Abs(rigidbody.position.x) >
    (Camera.main.orthographicSize *
    Camera.main.aspect))
    {
        rigidbody.position = new Vector3(-
        Mathf.Sign(rigidbody.position.x) *
        Camera.main.orthographicSize *
        Camera.main.aspect, 0, rigidbody.position.z);
        rigidbody.position -=
        rigidbody.position.normalized * 0.1f;
    }

    if (Mathf.Abs(rigidbody.position.z) >
    Camera.main.orthographicSize)
    {
        rigidbody.position = new
        Vector3(rigidbody.position.x,
        rigidbody.position.y, -
        Mathf.Sign(rigidbody.position.z) *
        Camera.main.orthographicSize);
    }
}
```

```

        rigidbody.position -=
rigidbody.position.normalized * 0.f;
    }
}

```

Hàm CheckExitScreen() sẽ tính toán toán vị trí mới của player trên màn hình khi nó đi ra khỏi screen sẽ là phía đối diện của nó và lệch đi một khoảng cách là 0,1f.

- Xử lý bắn đạn:

```

if (Input.GetButton("Jump") && shootingTimer <=
0.0)
{
    shootingTimer = 0.2f;
    // truyền đến máy khác là đang thực
    hiện ban
    photonView.RPC("Fire",
    RpcTarget.AllViaServer, rigidbody.position,
    rigidbody.rotation);
}

if (shootingTimer > 0.0f)
{
    shootingTimer -= Time.deltaTime;
}

```

Khi người chơi thực hiện nhấn space(chính là jump trong unity) thì phi thuyền sẽ bắn đạn và khi thực hiện bắn sẽ gọi phương thức Fire() bằng photonView.RPC() để thực hiện đồng bộ hóa trong trò chơi và cho các người chơi khác biết là mình đang thực hiện bắn.

### III. Tiêu kết chương 3

Tóm lại, cài đặt và phát triển game có thể coi là giai đoạn phức tạp cần nhiều kỹ năng và kiến thức để thực hiện trong một dự án game. Tuy có khó khăn nhưng trong quá trình cài đặt và xây dựng game, chúng ta đã học được nhiều kỹ thuật và công nghệ để tạo ra một trò chơi hoàn chỉnh và ấn tượng.

## **DANH MỤC TÀI LIỆU THAM KHẢO**

- [1] YenNhi, "DidongViet," 2 5 2023. [Online]. Available:  
<https://didongviet.vn/dchannel/multiplayer/>.
- [2] A. Media, "yeugame.vn," 6 8 2020. [Online]. Available:  
<https://yeugame.vn/game-la-gi-dinh-nghia-the-nao-la-game/>.
- [3] James, "Nuclino," 1 8 2022. [Online]. Available:  
<https://www.nuclino.com/articles/game-design-document-template>.
- [4] M. Hạnh, "Quantrimang," 12 5 2021. [Online]. Available:  
<https://quantrimang.com/cong-nghe/kien-thuc-ve-giao-thuc-mang-tcp-ip-48>.
- [5] TrangTran, "Vinahost," 12 6 2023. [Online]. Available:  
<https://blog.vinahost.vn/giao-thuc-udp-la-gi/>.
- [6] B. b. t. TopDev, "TopDev," 21 5 2022. [Online]. Available:  
<https://topdev.vn/blog/http-la-gi/>.
- [7] T. Q. Dat, "Viblo," 29 10 2017. [Online]. Available:  
<https://viblo.asia/p/websocket-la-gi-Ljy5VxkbZra>.
- [8] "Photon Engine," [Online]. Available: <https://www.photonengine.com/>.
- [9] "PUN," Photon, [Online]. Available:  
<https://doc.photonengine.com/pun/current/getting-started/pun-intro>.