 **BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH**



**BÁO CÁO BÀI TẬP LỚN**

**CHUYÊN ĐỀ CÔNG NGHỆ PHẦN MỀM**

**ĐỀ TÀI: XÂY DỰNG TRÒ CHƠI JUMP JUMP**

**BẰNG CÔNG NGHỆ UNITY**

GVHD: Trần Phong Nhã

**Sinh viên thực hiện:**

Trần Minh Nhựt – 6151071080

Nguyễn Thanh Nhân – 6151071017

Phạm Hoàng Yến - 6151071031

TP. Hồ Chí Minh, tháng 11 năm 2023

**MỤC LỤC**

[**LỜI CẢM ƠN** 5](#_Toc152091610)

[**NHẬN XÉT CỦA GIẢNG VIÊN** 6](#_Toc152091611)

[**PHÂN CÔNG CÔNG VIỆC** 7](#_Toc152091612)

[**Chương 1. Tổng quan đề tài** 8](#_Toc152091613)

[**1.1.** **Lý do chọn đề tài** 8](#_Toc152091614)

[**1.2.** **Mục tiêu của đề tài** 9](#_Toc152091615)

[**1.3. Lựa chọn công cụ** 9](#_Toc152091616)

[**1.4. Phạm vi chọn đề tài** 9](#_Toc152091617)

[**1.5. Đối tượng nghiên cứu** 10](#_Toc152091618)

[**Chương 2. Cơ sở lý thuyết** 11](#_Toc152091619)

[**2.1.**  **Công nghệ Unity** 11](#_Toc152091620)

[**2.1.1.** **Giới thiệu** 11](#_Toc152091621)

[**2.1.2.**  **Tính năng và công dụng phổ biến của Unity Engine** 12](#_Toc152091622)

[**2.1.3.**  **Ưu điểm** 13](#_Toc152091623)

[**2.1.4.** **Nhược điểm** 14](#_Toc152091624)

[**2.2. Visual Studio** 14](#_Toc152091625)

[**2.2.1.**  **Giới thiệu** 14](#_Toc152091626)

[**2.2.2.**  **Ưu điểm** 15](#_Toc152091627)

[**2.2.3.**  **Nhược điểm** 15](#_Toc152091628)

[**2.3. Ngôn ngữ C# (C-sharp)** 16](#_Toc152091629)

[**2.3.1.**  **Ưu điểm** 16](#_Toc152091630)

[**Chương 3. Triển khai trò chơi “Jump Jump”** 18](#_Toc152091631)

[**3.1. Kịch bản game** 18](#_Toc152091632)

[**3.1.1.** **Môi trường** 18](#_Toc152091633)

[**3.1.2.** **Đồ hoạ** 18](#_Toc152091634)

[**3.1.3.** **Âm thanh** 18](#_Toc152091635)

[**3.1.4.** **Cơ chế gameplay** 18](#_Toc152091636)

[**3.1.5.** **Các màn chơi** 18](#_Toc152091637)

[**3.1.6.** **Hệ thống số điểm** 18](#_Toc152091638)

[**3.2. Các nội dung thực hiện** 19](#_Toc152091639)

[**3.2.1.**  **Thiết kế đồ hoạ** 19](#_Toc152091640)

[**3.2.2.** **Coroutine và Spawn Enemy** 22](#_Toc152091641)

[**3.2.3.**  **Collider và Rigidbody** 23](#_Toc152091642)

[**3.2.4.**  **Particle System** 24](#_Toc152091643)

[**3.2.5.**  **Game Manager** 24](#_Toc152091644)

[**3.2.6.**  **Tạo Audio Manager** 26](#_Toc152091645)

[**3.2.7.**  **Waves Data và save game bằng PlayerPref** 27](#_Toc152091646)

[**3.2.8.**  **Sử dụng cách thuật toán Object Pooling** 28](#_Toc152091647)

[**3.2.9.**  **Singleton** 28](#_Toc152091648)

[**3.2.10.**  **Observers** 28](#_Toc152091649)

[**3.2.11.**  **Hệ thống cơ chế chơi** 28](#_Toc152091650)

[**3.2.12.**  **Cơ chế chiến đấu** 29](#_Toc152091651)

[**3.2.13.**  **Testing và debugging** 29](#_Toc152091652)

[**Chương 4. Triển khai trò chơi “Jump Jump”** 30](#_Toc152091653)

[**4.1. Giao diện bắt đầu** 30](#_Toc152091654)

[**4.2. Giao diện game** 30](#_Toc152091655)

[**4.3. Giao diện kết thúc** 31](#_Toc152091656)

[**Kết luận** 32](#_Toc152091657)

[**Tài liệu tham khảo** 34](#_Toc152091658)

**MỤC LỤC HÌNH ẢNH**

[Hình 1. Unity Engine 9](#_Toc152091591)

[Hình 2. Môi trường 1 19](#_Toc152091592)

[Hình 3. Môi trường 2 19](#_Toc152091593)

[Hình 4. Môi trường 3 20](#_Toc152091594)

[Hình 5. Player trong game 20](#_Toc152091595)

[Hình 6. Bản thảo Player 20](#_Toc152091596)

[Hình 7. Cạm bẫy 1 21](#_Toc152091597)

[Hình 8. Cạm bẫy 2 21](#_Toc152091598)

[Hình 9. Vật phẩm 21](#_Toc152091599)

[Hình 10. Địa hình 21](#_Toc152091600)

[Hình 11. Điểm kết thúc 22](#_Toc152091601)

[Hình 12. Audio Natural Music 26](#_Toc152091602)

[Hình 13. Audio Deadth Music 26](#_Toc152091603)

[Hình 14. Audio Jump 27](#_Toc152091604)

[Hình 15. Audio Collect 27](#_Toc152091605)

[Hình 16. Audio Finish 27](#_Toc152091606)

[Hình 17. Giao diện bắt đầu game 30](#_Toc152091607)

[Hình 18. Giao diện game 30](#_Toc152091608)

[Hình 19. Giao diện kết thúc 31](#_Toc152091609)

# **LỜI CẢM ƠN**

Sau quá trình học tập và rèn luyện bộ môn Công Nghệ Thông Tin tại trường Đại học Giao Thông Vận Tải – Phân Hiệu tại thành phố Hồ Chí Minh em đã được trang bị các kiến thức cơ bản, các kỹ năng thực tế để có thể hoàn thành đề tài bài tập lớn của em

Em xin gửi lời cám ơn chân thành đến quý thầy, cô bộ môn Công nghệ Thông tin trường Đại học Giao thông Vận tải – Phân hiệu tại thành phố Hồ Chí Minh đã quan tâm, hướng dẫn, truyền đạt những kiến thức và kinh nghiệm cho chúng em trong suốt thời gian học tập và thực hiện bài tập lớn này một cách tận tình. Em xin chúc quý thầy, cô thật nhiều sức khỏe và luôn đạt được thành công trong cuộc sống. Đặc biệt em xin cảm ơn thầy Trần Phong Nhã là người đã trực tiếp hướng dẫn và chỉ bảo em trong quá trình thực hiện đề tài “Xây dựng trò chơi Jump Jump bằng công nghệ Unity” này.

Sau một thời gian nỗ lực thực hiện thì đề tài cũng đã hoàn thành. Nhưng không thể tránh khỏi những sai sót do em còn chưa có nhiều kinh nghiệm thực tế. Em kính mong nhận được sự góp ý và nhận xét từ quý thầy, cô để em có thể hoàn thiện và hoàn thành tốt hơn cho đề tài của mình.

Lời sau cùng em một lần nữa kính chúc quý thầy, cô bộ môn Công nghệ Thông tin Trường Đại học Giao thông Vận tải – Phân hiệu tại thành phố Hồ Chí Minh thật nhiều sức khỏe và thành công.

Em xin chân thành cảm ơn!

# **NHẬN XÉT CỦA GIẢNG VIÊN**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

**……………………………………………………………………………………**

# **PHÂN CÔNG CÔNG VIỆC**

|  |  |  |  |
| --- | --- | --- | --- |
| **Họ và tên** | **MSSV** | **Phân công** | **Tự chấm** |
| Phạm Hoàng Yến | 6151071031 | Tìm assets (player, enemy), coroutine và spawn enemy | 9 |
| Nguyễn Thanh Nhân | 6151071017 | Tìm asstets (môi trường, cạm bẫy), collider và rigidbody | 9 |
| Trần Minh Nhựt | 6151071080 | Lên kịch bản game, singleton, thiết kế lối chơi | 9 |

# **Chương 1. Tổng quan đề tài**

* 1. **Lý do chọn đề tài**

Giải trí là một trong những nhu cầu thiết yếu của con người, ngày nay khi cuộc sống ngày càng phát triển đồng nghĩa với việc áp lực của con người ngày càng nhiều, do đó nhu cầu giải trí là không thể thiếu và cũng không phải ai cũng có thời gian để tham gia các hoạt động bên ngoài để giải tỏa căng thẳng nên chơi game là lựa chọn tối ưu của họ trong trường hợp này

Khi bắt đầu vào lĩnh vực làm game, các developer có thể thỏa sức sáng tạo để có thể tạo ra một thế giới như trong tưởng tượng của bản thân và hiện nay lập trình game không còn khó tiếp cận như ngày xưa nữa mà đã có cá công cụ và các phần mềm Game Engine hỗ trợ , giúp cho lập trình viên có thể tạo ra những game có đồ họa, cốt truyện , âm thanh và lối chơi cuốn hút có thể thu hút nhiều người chơi mà có thể cho họ có những trãi nghiệm tuyệt vời

Xây dựng một game là cơ hội để rèn luyện và phát triển kỹ năng công nghệ thông tin. Việc phát triển một game yêu cầu kiến thức về lập trình, đồ họa, thiết kế giao diện, quản lý dữ liệu và nhiều khía cạnh công nghệ khác. Đề tài game cho phép bạn áp dụng và nâng cao kỹ năng của mình trong lĩnh vực công nghệ thông tin, đồng thời mang lại niềm vui trong quá trình bạn phát triển con game đó . Do đó, để tạo nên một game giải trí và thư giãn, thách thức và cạnh tranh, kết nối xã hội, phát triển kỹ năng rèn luyện tư duy và làm hài lòng người chơi là một việc không hề dễ dàng.

Trong khuôn khổ đề tài của mình, em chỉ hoàn thành các chức năng, đồ họa, âm thanh, hiệu ứng, UI, lối chơi và lưu điểm chỉ dừng ở mức đơn giản nhất. Mặc dù em đã cố gắng rất nhiều để hoàn thành công việc do thời gian có giới hạn, kiến thức có hạn và kiến thức chưa nhiều nên còn nhiều thiếu xót cần được bổ sung. Em mong nhận được ý kiến đóng góp của thầy cô

* 1. **Mục tiêu của đề tài**

- Tạo ra một trò chơi chất lượng cao: Mục tiêu là xây dựng một trò chơi với đồ họa 2D đẹp mắt, âm thanh hấp dẫn và gameplay mượt mà, mang lại trải nghiệm hành động tuyệt vời cho người chơi.

- Thiết kế cấp độ và mức độ khó đa dạng: Mục tiêu là tạo ra một loạt các cấp độ khác nhau với mức độ khó tăng dần, từ dễ đến khó, để thử thách người chơi và giữ cho trò chơi hấp dẫn.

- Cung cấp hệ thống điều khiển linh hoạt và phản ứng: Mục tiêu là tạo ra một hệ thống điều khiển dễ dùng, cho phép người chơi di chuyển nhân vật của người chơi cách linh hoạt và phản ứng nhanh.

- Tích hợp hệ thống điểm số và thành tích: Mục tiêu là cung cấp một hệ thống điểm số để ghi nhận thành tích cao nhất của người chơi cũng như cung cấp thành tích cá nhân và thành tích xã hội để thúc đẩy sự chơi lại và nâng cao trải nghiệm của người chơi.

## **1.3. Lựa chọn công cụ**

Unity Engine



Hình 1. Unity Engine

## **1.4. Phạm vi chọn đề tài**

Tập trung vào phát triển triển một trò chơi hành động vũ trụ trên nền tảng các nền tảng di động, bao gồm các khía cạnh

- Thiết kế và phát triển trò chơi: Đề tài sẽ tập trung vào quá trình thiết kế và phát triển trò chơi Jump Jump. Các yếu tố quan trọng bao gồm mô hình hóa các laoij địa hình, tạo hình nhân vật và kẻ thù, xây dựng cấp độ, hệ thống điều khiển, âm thanh và hiệu ứng hình ảnh.

- Các kỹ thuật và công nghệ sử dụng Unity: Đề tài sẽ giới thiệu các kỹ thuật và công nghệ sử dụng trong việc phát triển trò chơi Jump Jump bằng Unity. Điều này bao gồm xử lý va chạm, điều khiển di chuyển, quản lý tài nguyên, hiệu ứng hình ảnh, âm thanh và giao diện người dùng.

- Tối ưu hóa và hiệu suất: Đề tài sẽ nghiên cứu các phương pháp và kỹ thuật để tối ưu hóa hiệu suất trò chơi Jump Jump, nhằm đảm bảo trải nghiệm chơi mượt mà và không gây lag. Việc tối ưu hóa có thể bao gồm quản lý bộ nhớ, tối ưu hóa các hàm tính toán và tối ưu hóa đồ họa.

## **1.5. Đối tượng nghiên cứu**

- Nhà phát triển game, lập trình viên và sinh viên quan tâm đến việc phát triển trò chơi sử dụng công nghệ Unity. Đề tài cung cấp kiến thức và hướng dẫn chi tiết về quy trình phát triển trò chơi, các kỹ thuật và công nghệ cần thiết, cũng như các yếu tố thiết kế và tối ưu hóa.

# **Chương 2. Cơ sở lý thuyết**

## **2.1. Công nghệ Unity**

### **2.1.1. Giới thiệu**

Unity là một công nghệ phát triển game đa nền tảng mạnh mẽ và phổ biến. Được ra mắt lần đầu vào năm 2005 bởi công ty Unity Technologies, Unity đã nhanh chóng trở thành một trong những công nghệ phát triển game phổ biến nhất trên thị trường. Với khả năng hỗ trợ đa nền tảng và giao diện đồ họa trực quan, Unity đã thu hút sự quan tâm của các nhà phát triển game từ các công ty lớn đến các cá nhân đam mê.

Một trong những yếu tố quan trọng giúp Unity trở thành một công nghệ phổ biến là khả năng tương thích đa nền tảng. Unity cho phép nhà phát triển tạo ra game cho nhiều nền tảng như PC, điện thoại di động, máy tính bảng, console và thậm chí cả thực tế ảo và thực tế tăng cường. Điều này giúp tiết kiệm thời gian và công sức khi phát triển game trên nhiều nền tảng khác nhau mà không cần viết lại mã nguồn từ đầu.

Một điểm mạnh khác của Unity là giao diện đồ họa trực quan và dễ sử dụng. Unity cung cấp một giao diện người dùng trực quan, cho phép nhà phát triển kéo và thả các đối tượng và thành phần vào môi trường làm việc. Điều này giúp nhà phát triển tạo nhanh các trò chơi mà không cần phải viết mã phức tạp. Ngoài ra, Unity cũng hỗ trợ nhiều ngôn ngữ lập trình như C#, JavaScript và Boo, cho phép nhà phát triển lựa chọn ngôn ngữ phù hợp với kỹ năng và sở thích của mình.

Unity cung cấp một loạt công cụ và tài nguyên hỗ trợ mạnh mẽ cho việc phát triển game. Có sẵn một cộng đồng lớn các nhà phát triển và nguồn tài nguyên trực tuyến, như các tài liệu hướng dẫn, video giảng dạy và các bản mẫu có sẵn để giúp nhà phát triển nắm bắt nhanh chóng các khía cạnh cơ bản và nâng cao kỹ năng của mình.

Điều đáng kể khác về Unity là khả năng tích hợp các công nghệ và dịch vụ bên thứ ba. Unity hỗ trợ tích hợp dễ dàng các công nghệ như AI, thực tế ảo, thực tế tăng cường, mạng lưới và hệ thống đa người chơi. Nó cũng có khả năng tích hợp các dịch vụ như PlayFab để quản lý cơ sở dữ liệu người chơi, giao dịch và quảng cáo.

Tóm lại, Unity là một công nghệ phát triển game đa nền tảng mạnh mẽ và linh hoạt. Với khả năng tương thích đa nền tảng, giao diện đồ họa trực quan và hỗ trợ mạnh mẽ từ cộng đồng, Unity là lựa chọn hàng đầu cho các nhà phát triển game. Với sự kết hợp của công nghệ, công cụ và ngôn ngữ lập trình như Unity, Visual Studio 2022, C# và JSON, nhà phát triển có thể tạo ra những trò chơi ấn tượng và đa dạng trên nhiều nền tảng khác nhau.

### **2.1.2. Tính năng và công dụng phổ biến của Unity Engine**

Unity Engine có nhiều tính năng và công dụng phổ biến giúp nhà phát triển tạo ra các trò chơi và ứng dụng tương tác. Một số tính năng và công dụng quan trọng của Unity:

- Đa nền tảng: Unity Engine hỗ trợ phát triển trên nhiều nền tảng khác nhau, bao gồm PC, Mac, Linux, iOS, Android, Xbox, PlayStation, Nintendo Switch và nhiều nền tảng khác. Điều này cho phép nhà phát triển triển khai sản phẩm của họ trên nhiều thiết bị và hệ điều hành, đáp ứng nhu cầu đa dạng của người dùng.

- Công cụ phát triển mạnh mẽ: Unity Engine cung cấp một loạt công cụ phát triển mạnh mẽ như Unity Editor, Animator, Timeline, Particle System và hơn thế nữa. Những công cụ này giúp nhà phát triển xây dựng và điều chỉnh các thành phần trong trò chơi như đồ họa, âm thanh, hình ảnh, điều khiển và xử lý logic.

- Đồ họa và hiệu ứng đa dạng: Unity Engine hỗ trợ đồ họa và hiệu ứng đa dạng, cho phép nhà phát triển tạo ra các trò chơi với đồ họa chất lượng cao. Nó hỗ trợ các công nghệ như shading, lighting, post-processing effects, particle effects và physics simulation để tạo ra các hiệu ứng đặc biệt và trải nghiệm hấp dẫn.

- Hỗ trợ 2D và 3D: Unity Engine hỗ trợ phát triển cả trò chơi 2D và 3D. Đối với trò chơi 2D, Unity cung cấp các công cụ và tính năng như Sprite Renderer, 2D Physics, Tilemap Editor và Animation System. Đối với trò chơi 3D, Unity hỗ trợ rendering 3D, hệ thống vật lý 3D, AI và các công cụ quản lý tài nguyên.

- Scripting languages: Unity Engine hỗ trợ việc viết mã bằng nhiều ngôn ngữ lập trình như C#, JavaScript và Boo. Tuy nhiên, trong các phiên bản gần đây, Unity đã chuyển sự tập trung chính vào việc hỗ trợ C#. Điều này cung cấp cho nhà phát triển một ngôn ngữ mạnh mẽ để điều khiển trò chơi và xử lý logic.

- Asset Store: Unity Asset Store là một cửa hàng trực tuyến lớn, cung cấp hàng ngàn tài nguyên sẵn có như mã nguồn, 3D models, textures, animation, sound effects, music và nhiều thứ khác. Asset Store giúp nhà phát triển tiết kiệm thời gian và công sức trong việc tạo và quản lý các tài nguyên cho trò chơi của mình.

- Hỗ trợ AR/VR: Unity Engine hỗ trợ phát triển ứng dụng thực tế tăng cường (AR) và thực tế ảo (VR). Các công cụ và tính năng AR/VR của Unity giúp nhà phát triển tạo ra trò chơi và ứng dụng tương tác chân thực trên các thiết bị AR/VR như Oculus Rift, HTC Vive, Microsoft HoloLens và các thiết bị di động hỗ trợ ARKit và ARCore.

Tính năng và công dụng phổ biến của Unity Engine đã giúp cho ngành công nghiệp trò chơi và ứng dụng tương tác phát triển mạnh mẽ và đa dạng.

### **2.1.3. Ưu điểm**

Đa nền tảng: Unity cho phép nhà phát triển tạo game cho nhiều nền tảng như PC, điện thoại di động, máy tính bảng, console, thực tế ảo và thực tế tăng cường. Điều này giúp tiết kiệm thời gian và công sức trong việc phát triển game đa nền tảng.

Giao diện đồ họa trực quan: Unity cung cấp giao diện người dùng trực quan và dễ sử dụng, cho phép nhà phát triển kéo và thả các đối tượng và thành phần vào môi trường làm việc. Điều này giúp tăng tốc quá trình phát triển và giảm độ phức tạp của mã nguồn.

Cộng đồng và tài nguyên phong phú: Unity có một cộng đồng lớn và sôi động của các nhà phát triển trên toàn thế giới. Có sẵn nhiều tài liệu hướng dẫn, video giảng dạy và các bản mẫu để giúp nhà phát triển học tập và giải quyết các vấn đề phát triển game. Điều này tạo điều kiện thuận lợi cho việc chia sẻ kiến thức và kinh nghiệm.

Tích hợp dễ dàng: Unity hỗ trợ tích hợp các công nghệ và dịch vụ bên thứ ba như AI, thực tế ảo, thực tế tăng cường và các hệ thống đa người chơi. Điều này giúp mở rộng khả năng phát triển và tạo ra các trò chơi phong phú và đa dạng.

### **2.1.4. Nhược điểm**

Hiệu suất: Đôi khi Unity có thể gặp các vấn đề hiệu suất, đặc biệt là khi xử lý các trò chơi phức tạp hoặc có đồ họa cao. Điều này yêu cầu nhà phát triển tối ưu hóa mã nguồn và tài nguyên để đảm bảo trải nghiệm chơi game mượt mà.

Kích thước file: Các trò chơi được phát triển bằng Unity có thể có kích thước file khá lớn. Điều này có thể gây khó khăn trong việc tải xuống và cài đặt trò chơi cho người dùng, đặc biệt là trên các thiết bị di động có dung lượng bộ nhớ hạn chế.

Hạn chế của miễn phí: Mặc dù Unity có phiên bản miễn phí để sử dụng, nhưng nếu muốn truy cập vào các tính năng và công cụ nâng cao hơn, như tích hợp dịch vụ hoặc phát triển cho các nền tảng đặc biệt, người dùng cần phải mua các gói cấp cao hơn.

Học hỏi ban đầu: Đối với người mới bắt đầu, việc học Unity có thể đòi hỏi thời gian và nỗ lực để làm quen với giao diện và cách làm việc của nó. Điều này đặc biệt đúng đối với những người không có kinh nghiệm lập trình trước đây

## **2.2. Visual Studio**

### **2.2.1. Giới thiệu**

Visual Studio là một môi trường phát triển tích hợp (IDE) được Microsoft phát triển để hỗ trợ việc phát triển các ứng dụng phần mềm đa nền tảng, bao gồm ứng dụng di động, web và desktop. Với một loạt các tính năng và công cụ mạnh mẽ, Visual Studio được coi là một trong những IDE hàng đầu trên thị trường.

### **2.2.2. Ưu điểm**

Đa nền tảng: Visual Studio hỗ trợ phát triển ứng dụng đa nền tảng, cho phép nhà phát triển xây dựng ứng dụng cho Windows, iOS, Android và nhiều hệ điều hành khác. Điều này giúp tiết kiệm thời gian và công sức cho việc phát triển ứng dụng trên nhiều nền tảng.

Đa ngôn ngữ: Visual Studio hỗ trợ nhiều ngôn ngữ lập trình phổ biến như C#, C++, Python, Java và nhiều ngôn ngữ khác. Điều này cho phép nhà phát triển lựa chọn ngôn ngữ phù hợp với dự án và sở thích cá nhân.

Giao diện người dùng thân thiện: Visual Studio 2019 có giao diện người dùng thân thiện và dễ sử dụng. Nó cung cấp các công cụ hỗ trợ như trình gỡ lỗi, trình soạn thảo mã, trình biên dịch và trình kiểm tra. Điều này giúp tăng năng suất và sự tiện lợi cho quá trình phát triển.

Tích hợp công cụ và dịch vụ: Visual Studio 2019 tích hợp nhiều công cụ và dịch vụ hữu ích như trình kiểm tra mã, quản lý phiên bản (version control), triển khai và quản lý ứng dụng. Nhà phát triển có thể tận dụng các tính năng này để nâng cao hiệu suất và chất lượng phát triển.

Cộng đồng lớn và hỗ trợ mạnh mẽ: Visual Studio 2019 có một cộng đồng lớn và nhiều tài liệu, ví dụ như tài liệu hướng dẫn, diễn đàn và các tài liệu tham khảo. Điều này giúp nhà phát triển dễ dàng tìm kiếm thông tin, giải quyết vấn đề và chia sẻ kiến thức.

### **2.2.3. Nhược điểm**

Tài nguyên hệ thống: Visual Studio yêu cầu tài nguyên hệ thống cao và cần có một máy tính có cấu hình đủ mạnh để chạy mượt mà. Điều này có thể gây ra trễ và ảnh hưởng đến hiệu suất khi phát triển.

Kích thước cài đặt: Việc cài đặt Visual Studio 2019 có thể tốn khá nhiều không gian đĩa cứng, đặc biệt khi chọn cài đặt tất cả các gói và tính năng. Điều này có thể làm giảm dung lượng lưu trữ sẵn có và làm chậm quá trình cài đặt.

Khó khăn trong việc tùy chỉnh: Một số người dùng cho rằng việc tùy chỉnh Visual Studio 2019 có thể phức tạp và khó khăn. Điều này có thể gây khó khăn cho những nhà phát triển muốn tùy chỉnh giao diện hoặc cấu hình IDE theo ý muốn.

## **2.3. Ngôn ngữ C# (C-sharp)**

C# (C-sharp) là một ngôn ngữ lập trình đa mục đích được phát triển bởi Microsoft. Nó là một thành phần quan trọng của .NET Framework và được sử dụng rộng rãi trong việc phát triển các ứng dụng desktop, web và di động.

### **2.3.1. Ưu điểm**

Dễ học và sử dụng: C# được thiết kế để có cú pháp gần gũi với ngôn ngữ tự nhiên, làm cho việc học và sử dụng nó dễ dàng hơn so với nhiều ngôn ngữ lập trình khác. Nó cung cấp nhiều tính năng tiện ích và cú pháp đơn giản giúp nhà phát triển tập trung vào việc giải quyết vấn đề thay vì lo lắng về các chi tiết kỹ thuật phức tạp.

Hỗ trợ mạnh mẽ từ .NET Framework: C# là một ngôn ngữ lập trình chính thức của .NET Framework, một nền tảng mạnh mẽ cho việc phát triển ứng dụng. .NET Framework cung cấp một loạt các thư viện, lớp và công cụ hỗ trợ phong phú cho việc xây dựng ứng dụng đa dạng.

An toàn và bảo mật: C# có một hệ thống kiểm soát kiểu mạnh mẽ, giúp ngăn chặn lỗi phát sinh do sử dụng sai kiểu dữ liệu. Nó cũng hỗ trợ các tính năng bảo mật như kiểm tra quyền truy cập và xác thực, giúp bảo vệ ứng dụng khỏi các cuộc tấn công và lỗ hổng bảo mật.

Hiệu suất cao: Với việc chạy trên nền tảng .NET và sử dụng công nghệ biên dịch JIT (Just-In-Time), C# có khả năng cung cấp hiệu suất cao và tăng cường tốc độ thực thi của ứng dụng.

Hỗ trợ đa nền tảng: C# không chỉ hỗ trợ phát triển ứng dụng trên nền tảng Windows, mà còn có thể được sử dụng để phát triển ứng dụng trên các nền tảng khác như Linux và macOS thông qua các framework như .NET Core và Xamarin.

**2.3.2. Nhược điểm**

Giới hạn trên các nền tảng không phải Windows: Mặc dù C# hỗ trợ đa nền tảng, tuy nhiên, việc phát triển ứng dụng C# trên các nền tảng không phải Windows có thể gặp một số hạn chế và khó khăn. Một số tính năng và công cụ chỉ hỗ trợ tốt trên nền tảng Windows.

Thời gian biên dịch lâu: C# sử dụng công nghệ biên dịch JIT, điều này có thể làm tăng thời gian biên dịch ứng dụng. Mặc dù việc biên dịch JIT cung cấp hiệu suất cao khi thực thi, nhưng nó có thể ảnh hưởng đến thời gian khởi động ban đầu của ứng dụng.

Hạn chế truy cập vào các ngôn ngữ khác: Mặc dù C# có thể tương tác với các ngôn ngữ khác thông qua .NET Framework, nhưng việc truy cập trực tiếp vào các thư viện và API của các ngôn ngữ khác có thể gặp hạn chế.

Hạn chế trên các thiết bị nhúng: Mặc dù C# có thể được sử dụng để phát triển ứng dụng nhúng, nhưng so với các ngôn ngữ lập trình nhúng truyền thống như C hay C++, nó có một số hạn chế trong việc tương tác trực tiếp với phần cứng và kiểm soát các tài nguyên thấp.

# **Chương 3. Triển khai trò chơi “Jump Jump”**

## **3.1. Kịch bản game**

### **3.1.1. Môi trường**

Trò chơi “Jump Jump” diễn ra trong nhiều môi trường khác nhau . Môi trường được thiết kế với các hìnhvới các địa hình và cạm bẩy đa dạng. Các màn chơi có thể có sự biến đổi của môi trường.

### **3.1.2. Đồ hoạ**

Đồ hoạ của trò chơi “Jump Jump” được thiết kế để tạo ra hiệu ứng hấp dẫn và dễ thương. Các địa hình, cạm bẫy và vật phẩm được thiết kế với chi tiết và hiệu ứng đặc biệt, tạo nên cảm giác thư giản và đồng thời dễ nhận biết

### **3.1.3. Âm thanh**

Âm thanh trong trò chơi “Jump Jump” đóng vai trò quan trọng để tăng cường trải nghiệm chơi game. Bao gồm nhạc nền thư giản , âm thanh nhảy, khi nhân vật tử trận, âm thanh khi va chạm hoặc tiếp xúc với các đối tượng khác.

### **3.1.4. Cơ chế gameplay**

Trò chơi “Jump Jump” có cơ chế gameplay nhanh nhẹn và thử thách. Người chơi sẽ đối mặt với đối thủ khó khăn, phải làm sao để tránh bẫy và lấy vật phẩm để tăng điểm để tồn tại. Trong đó có các chức năng điều khiển nhan vật (Player) như di chuyển sang trái (phím A), di chuyển sang phải (phím D), nhảy ( phím space ) và nhận lấy vật phẩm để qua màn và nếu bị trúng bẫy thì người chơi sẽ về nơi btaws đầu và chơi lại.

### **3.1.5. Các màn chơi**

Trò chơi “Jump Jump” thường chia thành các màn chơi khác nhau. Mỗi màn chơi có độ khó tăng dần và yêu cầu người chơi vượt qua các thử thách đặc biệt. Độ khó của màn chơi sẽ tăng lên khi qua các màn

### **3.1.6. Hệ thống số điểm**

Trò chơi “Jump Jump” có hệ thống điểm số để ghi nhận thành tích của người chơi. Người chơi sẽ nhận điểm khi lấy vật phẩm.

## **3.2. Các nội dung thực hiện**

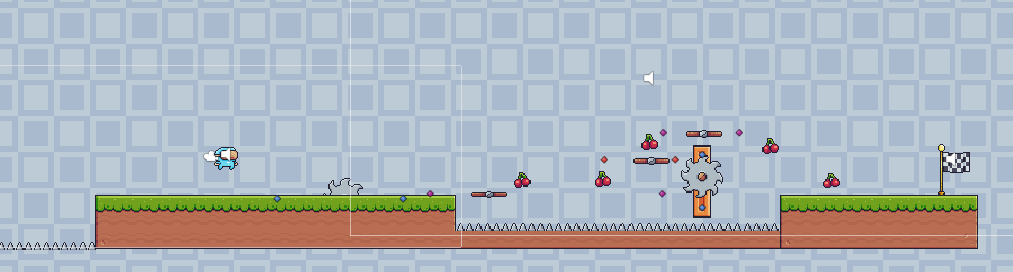
### **3.2.1. Thiết kế đồ hoạ**

**Thiết kế môi trường:**

Gồm có 3 môi trường có 3 độ khó khác nhau

* Môi trường 1:

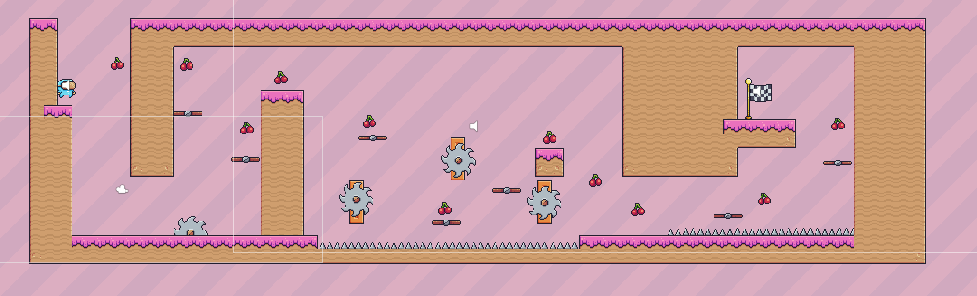
Với cạm bẫy không quá nhiều cho người chơi làm quen với cách chơi



Hình 2. Môi trường 1

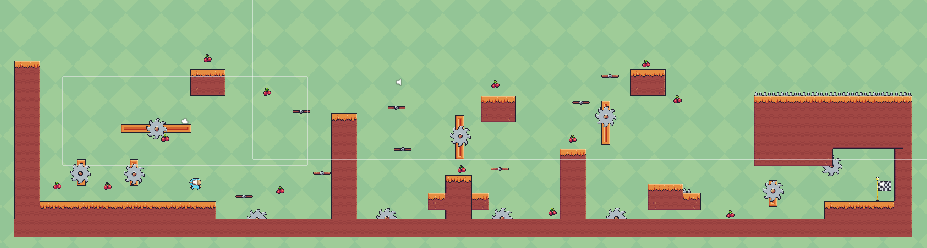
* Môi trường 2:

Với nhiều cạm bẫy hơn mà người chơi cần phải quan sát kỹ để đến được đích



Hình 3. Môi trường 2

Môi trường 3:

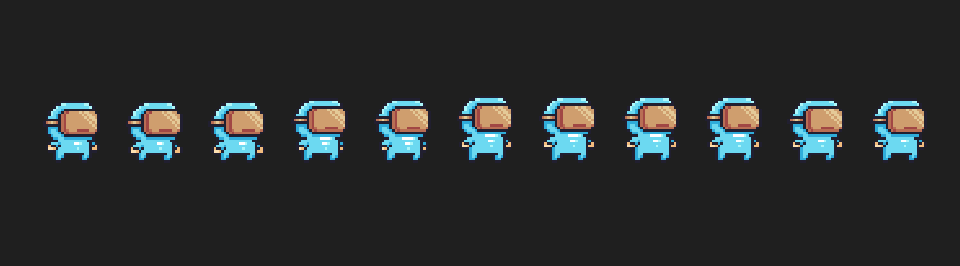


Hình 4. Môi trường 3

**Thiết kế player**:



Hình 5. Player trong game

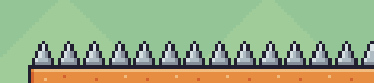


Hình 6. Bản thảo Player

**Cạm bẫy**



Hình 7. Cạm bẫy 1



Hình 8. Cạm bẫy 2

**Vật phẩm**



Hình 9. Vật phẩm

**Địa hình di chuyển:**



Hình 10. Địa hình

**Phần kết thúc màn và qua màn**



Hình 11. Điểm kết thúc

Khi nhân vật người chơi điểu khiển chạm vào vật phẩm này thì nhân vật của người chơi sẽ được chuyển qua một màn chơi khác

### **3.2.2. Coroutine và Spawn Enemy**

- Coroutine là một tính năng trong Unity cho phép thực hiện các tác vụ chờ đợi hoặc chạy đồng thời mà không cần block luồng chính của game. Điều này rất hữu ích khi xử lý các tác vụ phức tạp như việc tạo ra và quản lý kẻ địch trong trò chơi Jump Jump

- Spawning enemy (tạo đối tượng tàu quái vật) là quá trình tạo ra và triển khai các đối tượng quái vật trong trò chơi. Quá trình spawn enemy giúp tạo ra sự đa dạng và thách thức trong trò chơi. Khi người chơi tiếp cận các đối tượng quái vật mới, trò chơi trở nên thú vị hơn và đòi hỏi người chơi phải áp dụng các kỹ năng và chiến thuật để đối phó với chúng. Điều này giúp tạo ra sự thách thức và tương tác cho người chơi. Quá trình spawn enemy thường bao gồm các bước sau

• Xác định vị trí và thời gian spawn: Để tạo ra một đối tượng quái vật, cần xác định vị trí mà nó sẽ xuất hiện trong môi trường game. Đồng thời, cần đặt một thời gian hoặc điều kiện để quái vật được triển khai, ví dụ như sau một khoảng thời gian cố định, khi người chơi đạt được một mức độ cụ thể hoặc khi người chơi tiếp cận một khu vực nhất định.

• Tạo đối tượng quái vật: Sau khi đã xác định vị trí và thời gian spawn, cần tạo một đối tượng quái vật mới. Điều này bao gồm tạo ra một đối tượng đồ họa cho quái vật, áp dụng các thuộc tính như sức mạnh, tốc độ, mức độ khó khăn và hành vi của quái vật. Đối tượng quái vật có thể được tạo ra từ một mẫu (prefab) có sẵn hoặc được tạo ra động trong quá trình chơi game.

• Triển khai đối tượng quái vật: Khi đã có đối tượng quái vật được tạo ra, cần triển khai nó vào vị trí đã xác định. Điều này thường bao gồm di chuyển đối tượng quái vật đến vị trí spawn thông qua việc cập nhật tọa độ của nó trên màn hình.

• Quản lý và cập nhật đối tượng quái vật: Khi đối tượng quái vật đã được triển khai, cần quản lý và cập nhật nó trong quá trình chơi game. Điều này bao gồm việc kiểm tra va chạm với người chơi hoặc đối tượng khác, xử lý hành vi di chuyển và tấn công của quái vật, và xóa bỏ quái vật khi nó bị tiêu diệt hoặc thoát khỏi màn chơi.

### **3.2.3. Collider và Rigidbody**

Trong lập trình game, Collider và Rigidbody là hai thành phần quan trọng trong hệ thống vật lý của đối tượng trong môi trường game.

Collider (cụm từ viết tắt của "collision detector") là một thành phần được gắn kết với đối tượng trong trò chơi để xác định khu vực và hình dạng của nó trong không gian. Collider định nghĩa các ranh giới và biên giới xung quanh đối tượng, cho phép xác định xem khi nào các đối tượng va chạm với nhau. Khi hai collider va chạm vào nhau, một sự kiện va chạm có thể được kích hoạt để thực hiện các hành động phản ứng tương ứng, chẳng hạn như xử lý va chạm, đổi hướng hoặc kích hoạt hiệu ứng.

Rigidbody (thành phần "cơ thể cứng") là một thành phần chịu trách nhiệm xử lý vật lý và chuyển động của đối tượng trong trò chơi. Rigidbody cung cấp các thuộc tính như vận tốc, gia tốc, khối lượng và ma sát, cho phép đối tượng tương tác với lực và chuyển động dựa trên quy tắc vật lý. Với Rigidbody, đối tượng có thể di chuyển, rơi xuống, nhảy lên hoặc bị tác động bởi các lực khác nhau trong môi trường game.

Sự kết hợp giữa Collider và Rigidbody giúp xây dựng các hành vi vật lý chân thực trong game. Collider xác định khu vực và xử lý va chạm, trong khi Rigidbody quản lý vận tốc và chuyển động của đối tượng. Sự tương tác giữa hai thành phần này tạo nên một hệ thống vật lý đáng tin cậy và chính xác, giúp tạo ra trải nghiệm chơi game thú vị và chân thực hơn.

### **3.2.4. Particle System**

Particle System là một công cụ trong lĩnh vực đồ họa và hiệu ứng trong game để tạo ra và điều khiển các hạt nhỏ (particles). Nó cho phép tạo ra các hiệu ứng đặc biệt như hơi nước, khói, lửa, bụi, ánh sáng, hiệu ứng hạt nổi, và nhiều hiệu ứng khác.

Các hạt trong Particle System có thể có các thuộc tính khác nhau như tốc độ, hướng di chuyển, tuổi thọ, màu sắc, kích thước, hình dạng và mật độ. Các hạt có thể được tạo ra từ một nguồn gốc (emitter) và theo các quy tắc cụ thể để di chuyển, biến mất hoặc tương tác với môi trường.

Các Particle System cho phép tạo ra các hiệu ứng hấp dẫn và sống động trong game, tăng cường trải nghiệm thị giác và tạo ra cảm giác chân thực. Chúng được sử dụng rộng rãi trong các trò chơi để tạo ra hiệu ứng thời tiết, phản ứng vật lý, hiệu ứng phá hủy, hiệu ứng ma thuật và nhiều tác dụng khác.

Các tính năng chính của Particle System bao gồm:

- Tạo và điều chỉnh các thuộc tính của các hạt như tốc độ, hướng di chuyển, tuổi thọ, màu sắc và kích thước.

- Tùy chỉnh hình dạng và hình ảnh của các hạt.

- Xác định các quy tắc và điều kiện để di chuyển và biến mất các hạt.

- Tạo ra các hiệu ứng đa dạng như hơi nước, khói, lửa, bụi, ánh sáng và nhiều hiệu ứng khác.

- Tương tác với môi trường và các đối tượng khác trong game.

### **3.2.5. Game Manager**

**Tạo UI:**

* Trạng thái Home:
  + Một màn hình chào mừng với tiêu đề và hình ảnh liên quan đến game Space Shooter.
  + Nút "Play" để bắt đầu trò chơi.
* Trạng thái Gameplay: Giao diện chơi game chính, chứa các phần tử sau:
  + Khu vực chơi: Nơi diễn ra hành động chính trong game.
  + Thanh điểm số: Hiển thị điểm số của người chơi.
  + Thanh máu: Hiển thị mức máu hoặc sức khỏe của người chơi.
  + Thanh tiến trình nâng cấp: Hiển thị tiến trình nâng cấp đạn và player
  + Nút tạm dừng: Cho phép người chơi tạm dừng trò chơi và mở trạng thái Pause.
* Trạng thái Pause: Giao diện tạm dừng, chứa các phần tử sau:
  + Màn hình tạm dừng.
  + Nút "Continue" để tiếp tục trò chơi từ trạng thái tạm dừng.
  + Nút "Home" để thoát khỏi trò chơi và quay trở về trạng thái Home.
* Trạng thái Game Over: Giao diện hiển thị sau khi người chơi thua cuộc, chứa các phần tử sau:
  + Màn hình Game Over.
  + Hiển thị điểm số đạt được trong trò chơi.
  + Nút "Home" để thoát khỏi trò chơi và quay trở về trạng thái Home.

**Luồng sự kiện trong game:**

* Màn hình chính:
  + - Hiển thị tiêu đề và biểu tượng của trò chơi.
    - Hiển thị điểm số cao nhất hiện tại.
    - Cung cấp tùy chọn như "Play"
* Màn chơi:
  + Bắt đầu từ màn chơi đầu tiên, di chuyển và tiêu diệt tàu quái vật để qua các màn chơi tiếp theo.
  + Mỗi màn chơi có thể có mục tiêu giống nhau như tiêu diệt địch và né đạn.
  + Hiển thị điểm số và thành tích của người chơi.
  + Cung cấp lựa chọn dừng, tiếp tục vào màn chơi tiếp theo hoặc quay lại màn hình chính.
* Game Over:
  + Hiển thị thông báo "Game Over" nếu người chơi thất bại trong một màn chơi.
  + Cung cấp lựa chọn quay lại màn hình chính.

### **3.2.6. Tạo Audio Manager**

Audio Manager là một thành phần trong game được sử dụng để quản lý và điều khiển âm thanh trong trò chơi. Nó đảm nhận nhiều chức năng để cung cấp trải nghiệm âm thanh tốt nhất cho người chơi.

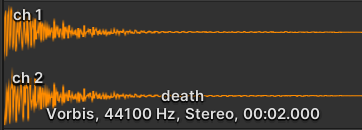
Các âm thanh bao gồm:

- Natural Music



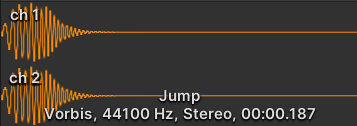
Hình 12. Audio Natural Music

- Death Music



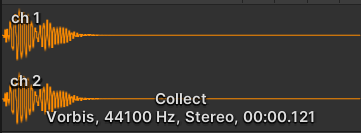
Hình 13. Audio Deadth Music

- Jump



Hình 14. Audio Jump

- Collect



Hình 15. Audio Collect

- Finish



Hình 16. Audio Finish

### **3.2.7. Waves Data và save game bằng PlayerPref**

Waves Data (Dữ liệu của các màn chơi) là thông tin về các màn chơi trong game, bao gồm số lượng và loại kẻ thù xuất hiện, tốc độ và cường độ của các màn chơi. Waves Data được sử dụng để quản lý và điều chỉnh sự xuất hiện và khó khăn của các màn chơi trong game. Nó giúp đảm bảo rằng trò chơi trở nên thú vị và thách thức hơn khi người chơi tiến qua từng màn.

Save game bằng PlayerPref (Người chơi ưu tiên lưu) là một cơ chế lưu trữ dữ liệu đơn giản và tiện lợi trong Unity. PlayerPref cho phép lưu trữ các giá trị như điểm số, cấp độ hoàn thành, trạng thái của trò chơi và các thông tin khác. Điểm mạnh của PlayerPref là nó dễ sử dụng và có thể lưu trữ dữ liệu ngay trên thiết bị của người chơi.

### **3.2.8. Sử dụng cách thuật toán Object Pooling**

Thuật toán Object Pooling: Object Pooling là một kỹ thuật được sử dụng trong lập trình game để quản lý và tái sử dụng các đối tượng trong trò chơi một cách hiệu quả. Thay vì tạo ra và hủy bỏ các đối tượng liên tục, Object Pooling giữ một số lượng đối tượng đã tạo sẵn trong một "bể" (pool) để sử dụng lại khi cần thiết. Khi cần sử dụng một đối tượng, thay vì tạo mới, ta sẽ lấy một đối tượng từ pool và đặt lại sau khi sử dụng xong. Điều này giúp giảm tải cho việc tạo/hủy đối tượng liên tục và tăng hiệu suất của trò chơi.

### **3.2.9. Singleton**

Singleton là một mẫu thiết kế (design pattern) trong lập trình mà đảm bảo chỉ có một đối tượng duy nhất được tạo ra từ một lớp cụ thể. Singleton đảm bảo rằng chỉ có thể truy cập vào một thể hiện duy nhất của lớp đó và cung cấp một cách tiếp cận toàn cục để truy xuất đến nó. Điều này thường được sử dụng khi muốn chia sẻ dữ liệu hoặc trạng thái giữa các thành phần của trò chơi mà không cần tạo ra nhiều thể hiện của lớp.

### **3.2.10. Observers**

Observers (Quan sát viên): Observers là một mô hình thiết kế trong lập trình mà cho phép các thành phần trong trò chơi theo dõi và tự động cập nhật khi có sự thay đổi trong trạng thái của một thành phần khác. Các thành phần quan sát (observers) đăng ký để nhận thông báo từ đối tượng quan sát (subject) và được thông báo khi trạng thái của subject thay đổi. Điều này giúp tạo ra sự giao tiếp và phản ứng linh hoạt giữa các thành phần trong trò chơi, và giúp giảm sự phụ thuộc trực tiếp giữa chúng.

### **3.2.11. Hệ thống cơ chế chơi**

Phát triển hệ thống điều khiển cho người chơi di chuyển tàu vũ trụ và tương tác với môi trường.

### **3.2.12. Cơ chế chiến đấu**

Thiết kế hệ thống chiến đấu để người chơi có thể đối đầu với các tàu quái vật và bảo vệ tàu vũ trụ của mình.

Xác định các kỹ năng và vũ khí có sẵn để người chơi có thể sử dụng và nâng cấp.

### **3.2.13. Testing và debugging**

Kiểm tra và sửa lỗi trong game để đảm bảo tính năng hoạt động tốt và trải nghiệm chơi game tốt hơn.

# **Chương 4. Triển khai trò chơi “Jump Jump”**

## **4.1. Giao diện bắt đầu**

**Giao diện bắt đầu game :**

Với giao diện dễ dàng tiếp cận



Hình 17. Giao diện bắt đầu game

## **4.2. Giao diện game**



Hình 18. Giao diện game

## **4.3. Giao diện kết thúc**



Hình 19. Giao diện kết thúc

# **Kết luận**

**Kết quả đạt được:**

Mục tiêu của dự án là tiếp cận và làm quen với công nghệ Unity, thời gian nghiên cứu và sự hướng dẫn để xây dựng một game 2D sử dụng Unity.

Game sẽ được thiết kế dưới dạng 2D để tạo ra một trải nghiệm thú vị và dễ tiếp cận các game thủ.

Giao diện được thiết kế đơn giản và dễ sử dụng, phù hợp với mọi lứa tuổi. Các chức năng sẽ được tối ưu hóa để tương tác với người chơi một cách thuận tiện và hiệu quả.

Về kiến thức cũng cố và vững vàng hơn trong kĩ thuật lập trình C-sharp, được tiếp cận làm quen với công nghệ mới, sử dụng các tool quản source code như Source Tree…

Về dự án đã xây dựng hoàn thành các chức năng cơ bản

**Hạn chế:**

Về giao diện người dung chưa được bắt mắt.

Về các màn chơi chưa đa dạng các Enemy, chưa có màn đấu với Boss, chưa có vật phẩm nâng cấp vũ khí.

Về server và code logic chưa được chặt chẽ và chưa hoàn toàn kiểm soát được quá trình từ phía người dùng cục bộ lên server.

Về việc triển khai chưa triển khai và công khai trên các ứng dụng như CHPlay

**Hướng phát triển:**

Về giao diện cập nhật giao diện phù hợp và đa dạng với người dùng

Về trển khai Game sẽ hoạt động trên các nền tảng thông dụng hiện nay như IOS, Android, Windown, MacOS…

Bổ sung các tính năng mới như hệ thống vũ khí đa dạng, vật phẩm nâng cấp tàu vũ trụ, chế độ chơi đa người, hoặc các màn chơi khác nhau với độ khó tăng dần.

Tạo ra các kịch bản đa dạng và thú vị để người chơi không bị nhàm chán. Có thể bổ sung các màn chơi đặc biệt, trò chơi minigame hoặc nhiệm vụ phụ để tăng tính thử thách và sự đa dạng cho game.

Nếu muốn, có thể phát triển chế độ chơi đa người qua mạng, cho phép người chơi thi đấu với nhau trực tuyến. Tạo ra hệ thống xếp hạng và thách thức giữa các người chơi để tăng tính cạnh tranh và độ nóng cho game.

# **Tài liệu tham khảo**

[1]. **Unity Learning**, *https://learn.unity.com*

[2]. **C# Observer design pattern** , *https://indiedevart.wordpress.com*

[3]. **Find asset and Package**, *https://assetstore.unity.com*

[4]. **Singleton in Unity**, *httpps://gamedevbeginner.com*

[5]. **Introduction to Object Pooling**, *https://learn.unity.com/tutorial/*

[6]. **Get and save data local device**, *https://gamedevbeginner.com/how-to-use-player-prefs-in-unity*