

**ĐỒ ÁN TỐT NGHIỆP**

**GAME 2D THỂ LOẠI PHIÊU LƯU –  
NHẬP VAI HÀNH ĐỘNG SỬ DỤNG  
UNITY VỚI ĐỒ HỌA PIXEL**

Ngành: **CÔNG NGHỆ THÔNG TIN**

Giảng viên hướng dẫn: Th.S Trịnh Đồng Thạch Trúc

Sinh viên thực hiện:

Nguyễn Trần Bảo Thương MSSV: 2180608085 Lớp: 21DTHJA2

Hồ Công Minh MSSV: 2180603504 Lớp: 21DTHJA2

Lý Hoàng Phúc MSSV: 2180605078 Lớp: 21DTHJA2



BỘ GIÁO DỤC VÀ ĐÀO TẠO  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM  
VIỆN CÔNG NGHỆ VIỆT – NHẬT

**ĐỒ ÁN TỐT NGHIỆP**

# **GAME 2D THỂ LOẠI PHIÊU LƯU – NHẬP VAI HÀNH ĐỘNG SỬ DỤNG UNITY VỚI ĐỒ HỌA PIXEL**

Ngành: **CÔNG NGHỆ THÔNG TIN**

Giảng viên hướng dẫn: Th.S Trịnh Đồng Thạch Trúc

Sinh viên thực hiện:

Nguyễn Trần Bảo Thương	MSSV: 2180608085	Lớp: 21DTHJA2
Hồ Công Minh	MSSV: 2180603504	Lớp: 21DTHJA2
Lý Hoàng Phúc	MSSV: 2180605078	Lớp: 21DTHJA2

TP. Hồ Chí Minh, tháng 05 năm 2025

**Viện Công nghệ Việt – Nhật**

**PHIẾU ĐĂNG KÝ  
ĐỀ TÀI ĐỒ ÁN/ KHÓA LUẬN TỐT NGHIỆP**

*Hệ: Chương trình Việt – Nhật*

**1. Họ và tên sinh viên/ nhóm sinh viên đăng ký đề tài (sĩ số trong nhóm: 3):**

(1) Nguyễn Trần Bảo Thương	MSSV: 2180608085	Lớp: 21DTHJA2
(2) Hồ Công Minh	MSSV: 2180603504	Lớp: 21DTHJA2
(3) Lý Hoàng Phúc	MSSV: 2180605078	Lớp: 21DTHJA2

Ngành : Công Nghệ Thông Tin

Chuyên ngành : Không phân chuyên ngành chuyên sâu

**2. Tên đề tài đăng ký : Game 2D thể loại phiêu lưu – nhập vai hành động sử dụng Unity với đồ họa pixel.**

Sinh viên đã hiểu rõ yêu cầu của đề tài và cam kết thực hiện đề tài theo tiến độ và hoàn thành đúng thời hạn.

*TP. HCM, ngày 31 tháng 3 năm 2025*

**Sinh viên đăng ký**  
*(Ký và ghi rõ họ tên)*

*Thương*  
Nguyễn Trần Bảo Thương

*Minh*  
Hồ Công Minh

*Phúc*  
Lý Hoàng Phúc

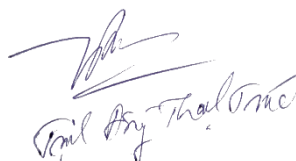
## PHIẾU GIAO NHIỆM VỤ THỰC HIỆN ĐỒ ÁN/ KHÓA LUẬN TỐT NGHIỆP

*(Mỗi nhóm một phiếu, GVHD ghi rõ tên đề tài và nhiệm vụ của từng sinh viên.  
GVHD chuyển cho SV để nộp về VP Viện Công nghệ Việt – Nhật)*

3. **Chủ đề/ hướng nghiên cứu đề tài:** Một trò chơi 2D được phát triển xoay quanh chủ đề bảo vệ môi trường, được lấy cảm hứng từ trò chơi điện tử tên Apple knight và Alba.
4. **Nhiệm vụ sinh viên thực hiện (Phân rõ nhóm trưởng):**

1) Tên đầy đủ: Nguyễn Trần Bảo Thương SĐT: 0375673418 Email: baothuong18022003@gmail.com Nhiệm vụ: Nhóm trưởng Thiết kế core gameplay Thiết kế Backend Quản lí âm thanh.	2) Tên đầy đủ: Hồ Công Minh SĐT: 0367446212 Email: congminhho.241103@gmail.com Nhiệm vụ: Thiết kế map Tìm và chèn asset vào game Kiểm thử Viết báo cáo.
3) Tên đầy đủ: Lý Hoàng Phúc SĐT: 0327527849 Email: lyhoangphuc2407@gmail.com Nhiệm vụ: Thiết kế UI Thiết kế các sự kiện Frontend Thiết kế quản lí kho vật phẩm.	

TP. HCM, ngày 24 tháng 2 năm 2025  
**Giảng viên hướng dẫn**  
(Ký và ghi rõ họ tên)

  
Giảng viên hướng dẫn

**PHIẾU XÁC NHẬN DUYỆT BÁO CÁO**  
**ĐỒ ÁN/ KHÓA LUẬN TỐT NGHIỆP**

**1. Họ và tên sinh viên/ nhóm sinh viên:**

SV: Nguyễn Trần Bảo Thương	MSSV: 2180608085	Lớp: 21DTHJA2
SV: Hồ Công Minh	MSSV: 2180603504	Lớp: 21DTHJA2
SV: Lý Hoàng Phúc	MSSV: 2180605078	Lớp: 21DTHJA2

**2. Tên đề tài:** Game 2D thể loại phiêu lưu – nhập vai hành động sử dụng Unity với đồ họa pixel.

**3. Xác nhận giảng viên đã đọc và duyệt cho sinh viên nộp ĐA/KLTN, bao gồm:**

☒ Báo cáo

☒ Poster

☐ Khác:

.....

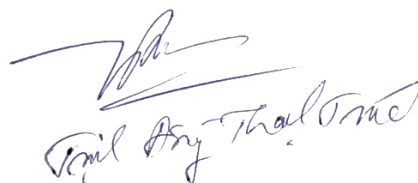
TP. HCM, ngày 27 tháng 5 năm 2025

**Giảng viên hướng dẫn phụ (Nếu có)**

(Ký và ghi rõ họ tên)

**Giảng viên hướng dẫn**

(Ký và ghi rõ họ tên)



Trần Thị Thanh Ngọc

## **LỜI CAM ĐOAN**

Nhóm chúng em thực hiện đồ án tốt nghiệp với đề tài “Game 2D thể loại phiêu lưu – nhập vai hành động sử dụng Unity với đồ họa pixel – The Last Gleam”, xin trân trọng cam đoan rằng toàn bộ nội dung được trình bày trong báo cáo này là kết quả của quá trình học tập, nghiên cứu và phát triển nghiêm túc, độc lập dưới sự hướng dẫn tận tình của giảng viên hướng dẫn.

Toàn bộ ý tưởng, nội dung thiết kế nhân vật, bối cảnh, lối chơi, hệ thống chức năng, mã nguồn và các tài nguyên được sử dụng trong game đều là sản phẩm do nhóm chúng em trực tiếp thực hiện hoặc sử dụng từ các nguồn hợp pháp, có dẫn nguồn rõ ràng và đúng quy định học thuật. Chúng em không sao chép, đạo nhái hay vi phạm bản quyền dưới bất kỳ hình thức nào.

Chúng em xin chịu hoàn toàn trách nhiệm trước Nhà trường, Khoa Công nghệ Thông tin, Viện Công nghệ Việt Nhật và giảng viên hướng dẫn về tính trung thực, tính sáng tạo và tính chính xác của toàn bộ nội dung báo cáo và sản phẩm game. Nếu phát hiện có bất kỳ hành vi gian lận hay vi phạm quy chế học thuật nào, chúng em xin hoàn toàn chịu trách nhiệm và chấp nhận mọi hình thức xử lý theo quy định.

Chúng em kính mong Hội đồng chấm đồ án, quý thầy cô và bạn đọc góp ý, đánh giá để sản phẩm của nhóm em được hoàn thiện hơn.

Trân trọng,

Nhóm TMP thực hiện.

## LỜI CẢM ƠN

Trong suốt thời gian từ khi bắt đầu học tập tại trường đến nay, chúng em đã nhận được rất nhiều sự quan tâm, giúp đỡ của Quý Thầy Cô, gia đình và bạn bè. Với lòng biết ơn sâu sắc, nhóm chúng em xin chân thành cảm ơn Khoa Công Nghệ Thông Tin và Viện Công Nghệ Việt - Nhật của trường Đại Học Công Nghệ - Hutech đã duyệt và tạo điều kiện cho chúng em thực hiện đề tài này.

Em xin chân thành cảm ơn thầy Trịnh Đồng Thạch Trúc, là người đã tận tình hướng dẫn nhóm chúng em, thầy đã có những trao đổi, hướng dẫn tận tình, chỉ bảo nhóm chúng em trong suốt thời gian thực hiện đề tài.

Em cũng xin gửi lời cảm ơn sâu sắc đến Quý Thầy Cô trong trường đã tận tình giảng dạy, trang bị cho chúng em những kiến thức quý báu trong những năm học vừa qua.

Nhóm chúng em xin gửi lòng biết ơn sâu sắc đến ba mẹ, các anh chị và bạn bè đã ủng hộ, giúp đỡ và động viên chúng em trong những lúc khó khăn cũng như trong suốt thời gian học tập và nghiên cứu.

Mặc dù chúng em đã cố gắng hoàn thành đề tài nhưng chắc chắn sẽ không tránh khỏi những thiếu sót. Kính mong nhận được sự cảm thông và góp ý của Quý Thầy Cô và các bạn. Xin chân thành cảm ơn!

# MỤC LỤC

DANH MỤC BẢNG BIỂU .....	iv
DANH MỤC HÌNH ẢNH .....	v
TÓM TẮT NỘI DUNG .....	vi
LỜI MỞ ĐẦU .....	1
CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI.....	1
1.1 Lí do chọn đề tài .....	1
1.2 Giới thiệu ý tưởng.....	1
1.3 Nội dung game.....	2
1.4 Kịch bản game .....	3
1.5 Các nhân vật và chức năng của từng lớp nhân vật .....	4
1.5.1 Nhân vật chính – Ember.....	4
1.5.2 Quái thường.....	5
1.5.3 Boss cuối – Morveth.....	7
1.5.4 Npc – dân làng.....	8
1.6 Phương pháp nghiên cứu và các công nghệ sử dụng.....	8
1.6.1 Phương pháp nghiên cứu.....	8
1.6.2 Các công nghệ sử dụng .....	9
1.7 Yêu cầu cấu hình.....	9
1.8 Yêu cầu chức năng trong game.....	9
1.8.1 Hệ thống nhân vật.....	9
1.8.2 Cơ chế chiến đấu .....	10
1.8.3 Cơ chế bản đồ và môi trường .....	10
1.8.4 Hệ thống giao diện .....	10
1.8.4 Vật phẩm nâng cấp.....	10
1.9 Kiểm thử trong game .....	10
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT .....	12
2.1 Tổng quan về Unity .....	12
2.1.1 Unity là gì.....	12
2.1.2 Sơ lược hình thành và phát triển .....	14
2.1.3 Ưu điểm của Unity .....	15
2.1.4 Các thành phần trong Unity Editor .....	16



2.1.5 Các khái niệm cơ bản trong Unity.....	17
2.2 Tổng quan về Unity DevOps Version Control .....	19
2.3 Tổng quan về C#.....	20
2.3.1 Các ưu điểm.....	21
2.3.2 Các nhược điểm.....	22
2.3.3 Mục tiêu của C# .....	22
2.3.4 Ứng dụng của C#.....	23
2.3.5 Đặc điểm của C# .....	24
2.3.6 Đặc trưng của C#.....	24
2.4 Tổng quan về thể loại game Platformer 2D.....	25
2.5 Tổng quan về cấu trúc game engine .....	26
2.6 So sánh với game cùng thể loại .....	27
2.6.1 So sánh với game Hollow Knight .....	27
2.6.2 So sánh với game Dead Cells.....	28
<b>CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG .....</b>	<b>29</b>
3.1 Mô hình Use-case tổng quát .....	29
3.2 Mô hình FSM (Finite State Machines) .....	30
3.3 Các giao diện của game .....	31
3.3.1 Giao diện khi vừa mở game .....	31
3.3.2 Giao diện khi vừa vào game.....	32
3.3.3 Giao diện option .....	33
3.3.4 Giao diện thông tin, chỉ số nhân vật hiện tại.....	34
3.3.5 Giao diện skill tree (cây kỹ năng) .....	35
3.3.6 Giao diện túi đồ .....	36
3.3.7 Giao diện carft vật phẩm trang bị.....	37
3.4 Một số giao diện khác trong game.....	38
3.5 Các State xử lý logic của nhân vật chính – Ember.....	42
3.6 Source code một số chức năng của nhân vật chính – Ember.....	43
3.6.1 Di chuyển .....	43
3.6.2 Lướt .....	44
3.6.3 Nhảy .....	45
3.6.4 Tấn công .....	46
3.6.5 Trượt tường (Wall Slide).....	47
3.6.6 Nhảy tường (Wall Jump).....	48

CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM.....	49
4.1 Kết quả thực nghiệm.....	49
4.2 So sánh với các game cùng thể loại.....	49
CHƯƠNG 5: KẾT LUẬN .....	51
5.1 Kết quả đạt được.....	51
5.2 Những điểm chưa đạt được.....	51
5.3 Định hướng phát triển.....	52
TÀI LIỆU THAM KHẢO.....	54

## **DANH MỤC BẢNG BIỂU**

Bảng 1. 1. Bảng yêu cầu cấu hình.....	9
Bảng 1. 2. Bảng Test-case chức năng di chuyển .....	10
Bảng 1. 3. Bảng Test-case chức năng lưu tiến trình. ....	11
Bảng 4. 1. Bảng so sánh với game cùng thể loại .....	49

## DANH MỤC HÌNH ẢNH

Hình 1. 1. Nhân vật chính .....	4
Hình 1. 2. Một số quái thường trong map rừng .....	5
Hình 1. 3. Một số quái thường trong map hang .....	6
Hình 1. 4. Một số quái thường trong map lâu đài .....	6
Hình 1. 5. Boss .....	7
Hình 1. 6. Dân làng .....	8
Hình 2. 1. Ảnh logo của ứng dụng Unity .....	12
Hình 2. 2. Ảnh logo của ứng dụng C# .....	20
Hình 3. 1. Mô hình Use-case tổng quát.....	29
Hình 3. 2. Mô hình FSM .....	30
Hình 3. 3. Giao diện khi vừa mở game .....	31
Hình 3. 4. Giao diện khi vừa vào game.....	32
Hình 3. 5. Giao diện option .....	33
Hình 3. 6. Giao diện thông tin chỉ số nhân vật.....	34
Hình 3. 7. Giao diện cây kĩ năng.....	35
Hình 3. 8. Giao diện túi đồ.....	36
Hình 3. 9. Giao diện craft vật phẩm trang bị.....	37
Hình 3. 10. Giao diện map hướng dẫn .....	38
Hình 3. 11. Giao diện map rừng 1 .....	39
Hình 3. 12. Giao diện map rừng 2.....	39
Hình 3. 13. Giao diện map hang 1 .....	40
Hình 3. 14. Giao diện map hang 2 .....	40
Hình 3. 15. Giao diện map lâu đài 1 .....	41
Hình 3. 16. Giao diện map lâu đài 2 .....	41
Hình 3. 17. Các State xử lí logic của nhân vật chính.....	42
Hình 3. 18. Code di chuyển trái, phải của nhân vật chính .....	43
Hình 3. 19. Code hành động lướt của nhân vật chính.....	44
Hình 3. 20. Code hành động nhảy của nhân vật chính .....	45
Hình 3. 21. Code hành động tấn công của nhân vật chính.....	46
Hình 3. 22. Code hành động trượt tường của nhân vật chính .....	47
Hình 3. 23. Code hành động nhảy tường của nhân vật chính .....	48

## TÓM TẮT NỘI DUNG

ゲーム「The Last Gleam」

ゲームの名称：The Last Gleam

はじめに：

近年、ゲーム業界では 2D プラットフォーマーゲームの力強い復活が見られています。このジャンルは、多くの世代のゲーマーの子供時代と深く結びついてきました。

Apple Knight、Hollow Knight、Dead Cells、Celeste などの作品は、ゲームプレイ、ビジュアル、感情表現に丁寧にこだわれば、2D ゲームでも世界中の何百万人ものプレイヤーを惹きつける深い体験を提供できることを証明しました。

私たちのチームが 2D プラットフォーマーゲームの開発をテーマに選んだのは、このジャンルの魅力だけでなく、現在のインディーゲーム界で人気を集め続けているピクセルアートの可能性にも注目したからです。

ピクセルアートは、3D グラフィックと比べて開発コストや時間を抑えられるだけでなく、クラシックかつ芸術的な美しさと象徴性を持ち、The Last Gleam の世界観と物語に非常に適しています。

ピクセルアートとゴシック調のスタイルを組み合わせることで、どこか懐かしく、同時に神秘的なゲーム世界が生まれ、プレイヤーを一枚一枚のシーンへと深く引き込む魅力を実現しています。

主な機能の概要：

本ゲームの基本的なプレイスタイルは、プレイヤーがキャラクターを操作し、さまざまな障害物を乗り越えるために特定のスキルを駆使するというものです。たとえば、キャラクターがトゲに触れたり、モンスターに攻撃された場合、現在の体力からダメージを受けます。

プレイヤーの操作が不十分で、キャラクターの HP がゼロになると、そのキャラクターは自動的に現在のステージのスタート地点または最後に到達したチェックポイントに戻されます。

ゲームには、装備のクラフト機能と、それらをキャラクターに装備してステータスを強化するシステムも備わっています。キャラクターは以下のような基本的なアクションを行うことができます：

- 移動
- 攻撃
- 壁に張り付く
- ジャンプ
- ダッシュ
- アイテムの拾得
- 装備の装着 など

また、ゲーム内には各地形エリアに応じたさまざまなモンスターが存在し、それぞれ異なる攻撃力・防御力を持っています。これらのモンスターを倒すことで、素材アイテムやソウル（Soul）がドロップされます。

ドロップされた素材は、装備をクラフトする際に使用でき、キャラクターのステータス強化に役立ちます。また、集めたソウルはスキルツリーでスキルを強化するために使用できます。

ステージ中には木箱が点在しており、その中にはランダムで素材アイテムが入っています。プレイヤーはこれらの木箱を壊してアイテムを獲得することができます。さらに、木を伐採したり岩を破壊することで、特定の素材を入手することも可能です。

ゲームの快適さと生存率を高めるために、回復用のアイテム使用機能も実装しており、ゲームの難易度を緩和しながら、戦略的なプレイを促進しています。

### 実験結果：

「The Last Gleam」のデモ版が完成した後、私たちのチームは Windows PC プラットフォーム上でゲームの主要機能およびパフォーマンスのテストを実施しました。

その結果、ゲームは安定して動作し、重大なバグは発生しませんでした。キャラクターの操作、戦闘システム、ボス戦、スキリエフェクト、装備のクラフトおよび装着機能などのコアメカニクスが正しく動作しており、ユーザーインターフェース（UI）も設計通りに機能していることが確認できました。

### 開発の方向性：

本プロジェクト「Unity を用いたピクセルアートスタイルの 2D アクション RPG ゲーム」は、当初の目標とスコープに沿って無事に完了しました。

この作品は、プログラミングスキル、ゲームプレイ設計の思考、ピクセルアートによる芸術的創造力を組み合わせた、私たちにとっての初めてのゲーム制作

となります。

本作は、冒険と探索、モンスターおよびボスとの戦闘要素を備えたアクション性の高い 2D 横スクロールゲームであり、物語面でも一定の深みを持っています。

- 完成した主な機能は以下のとおりです：
- キャラクターの操作システム
- スムーズで安定した戦闘（コンバット）システム
- モンスター用のシンプルな AI
- フェーズごとに構成されたボス（Morveth）の設計
- 独自の雰囲気演出するサウンドとライティング効果
- 難易度に応じて設計されたマップと明確なレベル構成

主人公キャラクターとボス・Morveth は、それぞれに独自のビジュアルスタイルを持ち、「光と闇」というテーマメッセージを視覚的に伝える役割を果たしています。

しかしながら、本プロジェクトにはいくつかの課題も残されています。たとえば、複数のプラットフォームへの対応が未実装である点や、ストーリーの内容に深みはあるものの、全体的にボリュームが不足しており、プレイヤーとの一体感をさらに高めるためには物語の拡張が必要です。

## LỜI MỞ ĐẦU

Trong thời đại công nghệ số phát triển không ngừng, ngành công nghiệp game không chỉ là lĩnh vực giải trí mà còn trở thành một hình thức nghệ thuật tương tác, kết hợp giữa kỹ thuật lập trình và tư duy sáng tạo. Việc phát triển một con game không chỉ đơn thuần là lập trình hay thiết kế, mà còn là quá trình hiện thực hóa một thế giới ảo có chiều sâu, có cảm xúc và mang đậm dấu ấn cá nhân.

Đề tài “Game 2D thể loại phiêu lưu – nhập vai hành động sử dụng Unity với đồ họa pixel” được thực hiện với mong muốn tiếp cận sâu hơn vào việc lập trình game, đồng thời rèn luyện các kỹ năng như thiết kế gameplay, xây dựng logic nhân vật, lập trình tương tác và tối ưu hóa trải nghiệm người chơi. Đây là cơ hội để nhóm thể hiện tư duy sáng tạo và niềm đam mê dành cho thế giới game, góp phần vào sự phát triển của ngành công nghiệp game nước nhà.

Báo cáo này trình bày toàn bộ quá trình hình thành, phát triển và hoàn thiện game, từ việc lên ý tưởng, chọn công cụ, xây dựng cốt truyện. Mặc dù đã rất cố gắng để hoàn thành đề tài, nhưng do thời gian có hạn và thiếu kinh nghiệm cũng như kỹ năng lập trình game còn hạn chế nên bản demo sẽ có không ít những thiếu sót. Chúng em rất mong nhận được sự cảm thông, nhận xét và góp ý từ Quý Thầy Cô để nhóm em có thể hoàn thiện tốt hơn trong tương lai. Chúng em xin chân thành cảm ơn!



# CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

## 1.1 Lí do chọn đề tài

Trong những năm gần đây, ngành công nghiệp game điện tử đang chứng kiến sự trở lại mạnh mẽ của các tựa game 2D platformer – dòng game từng gắn liền với tuổi thơ của nhiều thế hệ game thủ. Các sản phẩm như Apple Knight, Hollow Knight, Dead Cells, hay Celeste đã chứng minh rằng một tựa game 2D, nếu được đầu tư kỹ lưỡng về gameplay, hình ảnh và cảm xúc, vẫn có thể tạo nên trải nghiệm sâu sắc và thu hút hàng triệu người chơi trên toàn cầu.

Nhóm chúng em lựa chọn đề tài phát triển game 2D platformer không chỉ vì sức hút của thể loại này, mà còn bởi sự phổ biến và sức sống bền bỉ của đồ họa pixel art trong dòng game indie hiện nay. Pixel art không chỉ giúp tiết kiệm chi phí và thời gian phát triển so với đồ họa 3D phức tạp mà còn mang lại vẻ đẹp cổ điển, nghệ thuật và tính biểu tượng cao, rất phù hợp với không khí và câu chuyện của The Last Gleam. Sự kết hợp giữa pixel art và phong cách gothic giúp tạo ra một thế giới game vừa gần gũi, vừa huyền bí, thu hút game thủ phải đắm chìm trong từng khung hình.

Dòng game platformer là một thể loại lý tưởng cho những lập trình viên trẻ muốn thử sức, trải nghiệm và bắt đầu con đường sáng tạo của mình. Nó cho phép tiếp cận dễ dàng với cơ chế điều khiển, dễ kiểm thử, nhưng vẫn đầy thử thách trong việc thiết kế màn chơi và xây dựng hệ thống AI cho quái và npc trong game.

Chúng em lựa chọn đề tài này là để thử thách bản thân cũng như mong muốn tạo ra một sản phẩm game có chiều sâu nội dung, mang thông điệp về hy vọng, sự mất mát và ánh sáng trong bóng tối. Câu chuyện về Ember – Tia Lửa Cuối Cùng trong một thế giới đang hấp hối là phép ẩn dụ cho tinh thần vượt lên nghịch cảnh, giữ vững niềm tin dù trong hoàn cảnh đen tối nhất. Đây không đơn thuần là một tựa game đi cảnh chiến đấu, mà là một hành trình tinh thần nơi mỗi bước nhảy, mỗi đòn đánh đều mang ý nghĩa.

## 1.2 Giới thiệu ý tưởng

Game có tên là The Last Gleam. Game chủ yếu tập trung vào một nhân vật là Swordswoman của dòng 2D Platformer hay còn gọi là game phiêu lưu đi màn vượt chướng ngại vật và tiêu diệt các quái vật có trên đường đi của nhân vật. Lối chơi của các

game này liên quan chủ yếu tới việc di chuyển nhân vật phải có một số kỹ năng nhất định để có thể vượt qua các chướng ngại vật, ví dụ như khi nhân vật chạm vào gai hoặc bị quái vật tấn công thì nhân vật sẽ mất máu trong tổng lượng máu hiện có. Nếu nhân vật mà người chơi điều khiển không được tốt và máu của nhân vật bị rút xuống còn không máu thì nhân vật sẽ tự động quay trở lại vị trí xuất phát của màn chơi hiện tại hoặc vị trí checkpoint gần nhất, cơ chế craft trang bị và trang bị lên cho nhân vật để có thể tăng chỉ số cơ bản, nhân vật sẽ có những chức năng cơ bản như di chuyển, tấn công, bám tường, nhảy, lướt, nhặt vật phẩm và trang bị vật phẩm,... Ngoài ra game còn có các quái vật được chia theo từng khu vực địa hình với các chỉ số tấn công và phòng thủ khác nhau, khi chết chúng sẽ rơi ra một số vật phẩm nguyên liệu và linh hồn (Soul) và từ những vật phẩm nguyên liệu đó người chơi có thể craft thành trang bị giúp tăng thêm chỉ số nhân vật và dùng số linh hồn (Soul) để nâng cấp Skill trong Skill Tree. Trong màn chơi sẽ có những thùng gỗ có chứa vật phẩm nguyên liệu, người chơi cũng có thể phá những thùng gỗ đó để lấy ngẫu nhiên những vật phẩm nguyên liệu. Tất nhiên người chơi có thể chặt cây hoặc đập đá để có thể lấy vật phẩm nguyên liệu mong muốn. Để thêm phần tối ưu chúng em có thêm cơ chế sử dụng vật phẩm để hồi máu giúp tăng khả năng sống sót của nhân vật đồng thời giúp game dễ dàng hơn.

Để hoàn thành game, người chơi phải thật sự cẩn thận và có sự tập trung trong các bước di chuyển trong màn chơi. Cuối cùng phải thực sự hạ được boss cuối để kết thúc game.

### **1.3 Nội dung game**

Trong một thế giới từng rực rỡ ánh sáng tên là Elaria, Ngọn Lửa Thiêng là nguồn sống và ánh sáng duy nhất bị cướp đi bởi Morveth – The Withering Flame (Ngọn lửa từng rực rỡ, giờ chỉ còn lại tro tàn và mục nát). Hắn từng là một trong những chiến binh ánh sáng cao quý dưới trướng hội đồng Elaria, với kỹ năng chiến đấu và tài năng vốn có Morveth cùng với những đồng đội đã lập nên vô số những chiến công “Dưới lời thề với Elaria. Ta Morveth sẽ dẫn dắt các chiến sĩ anh dũng vô song cùng bước trên con đường chinh chiến từng bước đẩy Vực Bóng Đen vào đường cùng”. Nhưng sau khi tiêu diệt thủ lĩnh Vực Bóng Đen Morveth đã bị thao túng bởi ma lực của thủ lĩnh Vực Bóng Đen và không còn là chính mình, linh hồn anh đã chết và chỉ còn lại thân xác rỗng với toàn bộ

nguồn ma lực mạnh mẽ. Giờ đây thanh gươm mạnh nhất của Elaria đã không còn và toàn bộ Elaria lại một lần nữa chìm vào bóng tối.

Cả thế giới bị chìm trong màn đêm vĩnh cửu. Quái vật trỗi dậy từ lòng đất, và động thực vật bắt đầu trở nên biến dị, con người dần tuyệt vọng. Nhưng một tia hy vọng nhỏ nhoi vẫn còn tồn tại đó là Ember. Ember là một nữ chiến binh dũng cảm, cô là một trong những chiến binh ánh sáng còn sót lại và là người đồng đội cũ đã từng kề vai sát cánh với Morveth. Người chơi sẽ vào vai Ember – Tia Lửa Cuối Cùng Của Elaria một chiến binh mang trong mình ánh sáng thuần khiết với sứ mệnh cao cả đó là bảo vệ ngôi làng nhỏ và tiêu diệt Morveth nhằm khôi phục Elaria.

#### **1.4 Kịch bản game**

##### **Cảnh 1: Hồi ức trước màn đêm vĩnh cửu (Chuyển cảnh từ một Elaria rực rỡ ánh sáng sang một vùng đất hoang tàn.)**

Ember quỳ xuống bên tàn tích của một ngôi làng. Ánh sáng yếu ớt từ viên pha lê trong tay cô đang dần lụi tàn. Gió mang theo tiếng vọng từ quá khứ, những giọng nói thân quen giờ chỉ là ký ức...

Lão trưởng làng: Ember, con là tia sáng cuối cùng của chúng ta. Nếu con bỏ cuộc, Elaria thực sự sẽ biến mất!

Ember: Cháu biết, thưa ngài. Nhưng Morveth... hắn từng là một chiến binh của ánh sáng. Cháu không thể để linh hồn hắn bị bóng tối nuốt chửng.

##### **Cảnh 2: Cuộc đối đầu định mệnh (Trên đỉnh tàn tích của thành trì cũ, nơi từng là trung tâm của Elaria, Ember chạm trán Morveth.)**

Morveth: Ember... Người đến để chết sao?

Ember: Không. Ta đến để cứu người.

Morveth: (Cười lớn) Người còn tin vào những lời dối trá của ánh sáng ư? Ta đã thấy sự thật. Thế giới này không cần ánh sáng!

Ember: Ta không tin vào dối trá. Ta tin vào con người thật của Morveth, người đã từng chiến đấu vì Elaria!

Ember: (Thì thầm) Morveth... nếu anh vẫn còn chút gì thuộc về con người cũ, hãy lắng nghe tôi.

Morveth: Elaria đã không còn... ánh sáng là một lời nói dối. Và ta ta là kẻ duy

nhất nhìn thấu sự thật!

Morveth vùng liềm, ngọn lửa đen bùng lên. Ember rút kiếm, Cô siết chặt thanh kiếm trong tay, ánh sáng từ viên pha lê bùng lên một lần nữa, soi rọi bóng tối. Trận chiến sắp bắt đầu... Nhưng đây không chỉ là cuộc chiến giữa ánh sáng và bóng tối, mà còn là cuộc chiến giành lại linh hồn của một người từng là huyền thoại. Lao về phía trước. Cuộc chiến cuối cùng bắt đầu.

## **1.5 Các nhân vật và chức năng của từng lớp nhân vật**

### *1.5.1 Nhân vật chính – Ember*



*Hình 1. 1. Nhân vật chính*

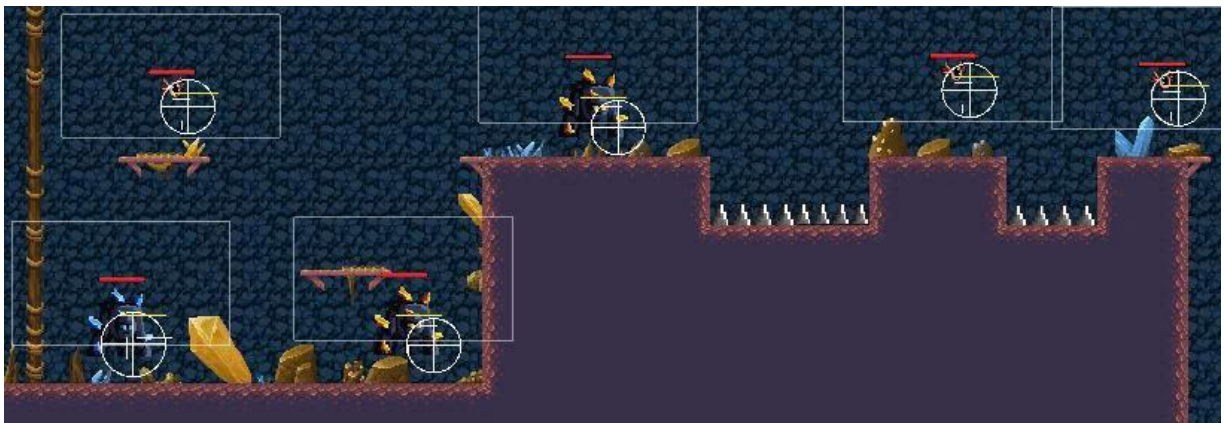
- Di chuyển qua trái.
- Di chuyển qua phải.
- Nhảy.
- Trượt tường.
- Nhảy tường.
- Lướt.
- Tấn công.
- Craft vật phẩm.

- Sử dụng vật phẩm.

### 1.5.2 Quái thường



Hình 1. 2. Một số quái thường trong map rừng







*Hình 1. 3. Một số quái thường trong map hang*

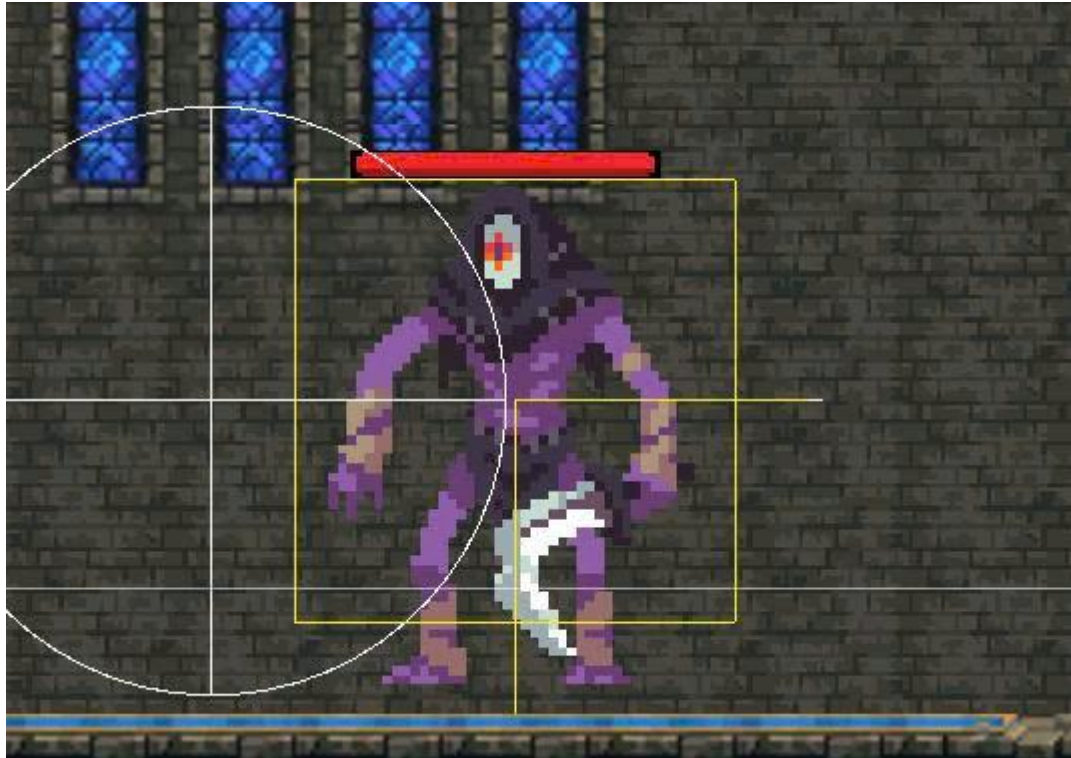


*Hình 1. 4. Một số quái thường trong map lâu đài*

- Di chuyển trái phải.

- Tấn công.
- Rút vật phẩm và Soul khi bị hạ gục

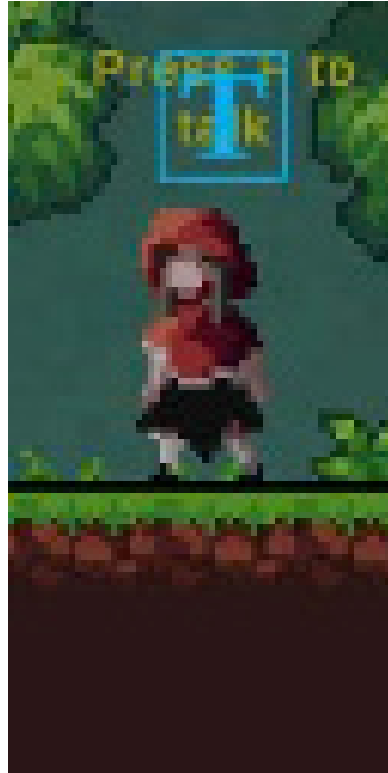
### 1.5.3 Boss cuối – Morveth



Hình 1. 5. Boss

- Di chuyển trái phải.
- Tấn công 1.
- Tấn công 2.
- Tấn công 3.

#### 1.5.4 Npc – dân làng



Hình 1. 6. Dân làng

- Nói chuyện.

### 1.6 Phương pháp nghiên cứu và các công nghệ sử dụng

#### 1.6.1 Phương pháp nghiên cứu

Dự án game The Last Gleam được phát triển chủ yếu dựa trên cảm hứng cá nhân từng thành viên trong nhóm, bắt nguồn từ niềm yêu thích đối với thể loại platformer hành động 2D và phong cách u tối, giàu chiều sâu cảm xúc. Nhóm chúng em lựa chọn tiếp cận theo hướng sáng tạo cá nhân để tạo ra một con game phản ánh trực tiếp cảm xúc, tư duy thẩm mỹ và trải nghiệm chơi game của người làm game.

Game được phát triển với mục tiêu tạo ra prototype có thể chơi được, nhằm thể hiện rõ ý tưởng gameplay và phong cách đồ họa. Sản phẩm có chứa các hệ thống nâng cao như: lưu trữ dữ liệu và hệ thống kỹ năng nâng cấp.

Toàn bộ tài nguyên đồ họa bao gồm nhân vật, hiệu ứng, nền, tilemap, âm thanh,... đều được nhóm chúng em tìm kiếm từ các nguồn miễn phí cho phép sử dụng công khai trên Internet, nhằm tập trung thời gian cho phần lập trình và thiết kế gameplay.



### 1.6.2 Các công nghệ sử dụng

Về mặt kỹ thuật, The Last Gleam được xây dựng bằng Unity Engine, một công cụ mạnh mẽ và phổ biến trong việc phát triển game 2D. Việc sử dụng ngôn ngữ lập trình C# giúp nhóm dễ dàng kiểm soát logic của game, tạo hệ thống điều khiển linh hoạt và phát triển các chức năng như tấn công, AI cơ bản, điều hướng camera và xử lý hiệu ứng thời tiết trong game.

Phát triển game thì chắc chắn không thể thiếu việc quản lý mã nguồn, đồng bộ dữ liệu và hỗ trợ làm việc nhóm, nhóm chúng em đã sử dụng Unity DevOps Version Control để lưu trữ, theo dõi và khôi phục các thay đổi theo thời gian thực. Việc áp dụng công cụ này giúp quá trình phát triển trở nên ổn định, giảm thiểu rủi ro mất dữ liệu và tăng hiệu suất làm việc của nhóm.

### 1.7 Yêu cầu cấu hình

*Bảng 1. 1. Bảng yêu cầu cấu hình*

Operating System	Window 10 (64 bit).
Processor – CPU	AMD Athlon64 3000 1.8 GHz, Intel Pentium IV 630 @ 3.0 GHz.
RAM	2 GB.
Video card – GPU	AMD Radeon HD 2900 XT (512 MB) NVIDIA GeForce 8800 GT (512 MB).
Hard drive	2 GB.
DirectX version	DirectX Redist June 2010.

### 1.8 Yêu cầu chức năng trong game

#### 1.8.1 Hệ thống nhân vật

- Người chơi điều khiển nhân vật chính di chuyển trái, phải, nhảy, leo trèo, lướt và tấn công.
- Nhân vật có thể sử dụng kỹ năng đặc biệt (mở khóa dần theo tiến trình game).
- Hệ thống HP (máu) và tài nguyên trong game.

### 1.8.2 Cơ chế chiến đấu

- Tấn công cận chiến với vũ khí chính (kiếm, rìu, v.v.).
- Né đòn và lướt né đòn
- Tích hợp AI cơ bản cho kẻ địch (tuần tra, phát hiện, tấn công).

### 1.8.3 Cơ chế bản đồ và môi trường

- Màn chơi được chia thành các khu vực nối liền nhau: Lâu Đài, Rừng Sâu, Hang Tối.
- Mỗi vùng có các điểm lưu game theo dạng cây thế giới.

### 1.8.4 Hệ thống giao diện

- HUD hiển thị thanh máu, kỹ năng, bình máu.
- Giao diện menu chính, menu tạm dừng, và menu chọn vật phẩm/kỹ năng.

### 1.8.4 Vật phẩm nâng cấp

- Hệ thống thu thập vật phẩm: vũ khí, trang bị, tài nguyên và nguyên liệu.
- Hệ thống chế tạo hoặc cường hóa vũ khí/kỹ năng.
- Quản lý kho đồ (Inventory).

## 1.9 Kiểm thử trong game

Bảng 1. 2. Bảng Test-case chức năng di chuyển

Thuộc tính	Nội dung
Tên kiểm thử.	Kiểm thử di chuyển và nhảy cơ bản của nhân vật chính.
Mô tả.	Đảm bảo rằng nhân vật chính có thể di chuyển trái/phải và nhảy bình thường khi không có vật cản.
Tiền điều kiện	Trò chơi đã vào Scene đầu tiên (khu vực hướng dẫn), nhân vật đang ở trạng thái idle.
Bước kiểm thử	1. Nhấn phím ← để di chuyển trái 2. Nhấn phím → để di chuyển phải

	3. Nhấn phím Space để nhảy.
Kết quả mong đợi	1. Nhân vật di chuyển mượt theo hướng tương ứng 2. Hoạt ảnh chạy được kích hoạt 3. Khi nhấn Space, nhân vật nhảy lên đúng độ cao đã định.
Kết quả thực tế	Đạt.
Tình trạng	Pass.

*Bảng 1. 3. Bảng Test-case chức năng lưu tiến trình.*

Thuộc tính	Nội dung
Tên kiểm thử.	Lưu và tải tiến trình trò chơi.
Mô tả.	Đảm bảo rằng người chơi có thể lưu game tại điểm lưu và tải lại đúng trạng thái đã lưu.
Tiền điều kiện	Nhân vật đã đến điểm lưu (Checkpoint).
Bước kiểm thử	1. Tương tác với điểm lưu game (đi qua Checkpoint). 2. Thoát game. 3. Mở lại và chọn “Chơi tiếp”.
Kết quả mong đợi	Game tải đúng trạng thái vị trí, HP, chỉ số, vật phẩm như lúc lưu.
Kết quả thực tế	Đạt.
Tình trạng	Pass.

## CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

### 2.1 Tổng quan về Unity



*Hình 2. 1. Ảnh logo của ứng dụng Unity*

#### 2.1.1 Unity là gì

Unity là một công cụ phát triển game đa năng và phổ biến, được sử dụng rộng rãi trên toàn cầu trong cả lĩnh vực game indie lẫn các sản phẩm thương mại quy mô lớn. Unity hỗ trợ xây dựng các game 2D và 3D với khả năng chạy mượt trên nhiều nền tảng khác nhau như Windows, macOS, Android, iOS, WebGL, console (PlayStation, Xbox, Nintendo Switch) và các thiết bị thực tế ảo (VR).

Unity mang lại sức mạnh kỳ diệu cho các nhân vật mà chúng ta muốn thể hiện sống động hơn trong một không gian 3D huyền diệu. Công nghệ tiên tiến này tạo ra bước đột phá mới trong công nghệ sản xuất game ngày nay, mang đến cho người chơi cảm giác vô cùng khác biệt và thú vị với từng bước di chuyển, công nghệ này sẽ được áp dụng trong thời gian tới. Việc sử dụng nó trong các game Việt Nam sẽ mở ra một trang mới trong thế giới game 2D và 3D tuyệt vời.

Unity được sử dụng để tạo trò chơi điện tử hoặc những nội dung có tính tương tác như thể hiện kiến trúc, hoạt hình 2D, 3D thời gian thực. Unity tương tự như Director, Blender Game Engine, Virtools hay Torque Game Builder về việc sử dụng môi trường

đồ họa tích hợp trong phát triển game.

Unity là một trong những engine được các nhà phát triển game có sở thích cực kỳ ưa chuộng do khả năng phát triển tuyệt vời của nó.. Trình chỉnh sửa chạy trên Windows và Mac OS và có thể tạo trò chơi cho Windows, Mac, Wii, iOS, Android.

Trò chơi cũng có thể được chơi trong trình duyệt web thông qua plugin Unity Web Player. Unity thêm khả năng xuất ra trò chơi dựa trên widget cho Mac, Xbox 360 và cả PlayStation 3.

Chỉ với một số tiền khá khiêm tốn (1.800 USD), bản Pro đã về tay chúng ta, tất nhiên số tiền này vẫn còn quá lớn ở Việt Nam, nhưng rất may là đã có bản Unity miễn phí. Tuy nhiên, nhiều tính năng quan trọng (Network) sẽ bị cắt giảm, nhưng đó không phải là vấn đề quá khó nếu muốn phát triển một game tầm trung.

Năm 2009, Unity nằm trong số 5 game engine hàng đầu cho sản xuất game chỉ với 4 năm phát triển. Unity đứng ở vị trí thứ 4 sau Unreal. Engine 3, Gamebryo Engine (được VTC Studio mua để phát triển SQUAD) và Cry Engine 2. Số lượng tài liệu hướng dẫn Unity rất phong phú. Ngoài ra, nó cũng có một cộng đồng tuyệt vời với diễn đàn riêng. Có gì không hiểu chúng ta có thể hỏi và được giải đáp tận tình, nhanh chóng.

Quá trình tạo địa hình cũng như truy xuất xuất từ 3DSMax, Maya, Cinema4D ... rất nhanh chóng. Sức mạnh và sự tiện lợi của Unity là vô cùng to lớn. Điểm mạnh: Unity có thể tạo ra nhiều loại game 2D và 3D chất lượng cao, dễ sử dụng cho các nhà phát triển game không chuyên chạy trên hầu hết mọi hệ điều hành.

Một người sử dụng phần mềm nguồn mở 3Dmax, Maya hoặc Blender là điều quá tuyệt vời, chúng tôi sẽ có lợi thế lớn nếu viết trò chơi trong Unity này, bởi vì việc tạo mô hình 2D và 3D trở nên dễ dàng hơn nhiều, sự kết hợp giữa lập trình viên và nhà thiết kế mô hình trở nên nhanh hơn và hiệu quả hơn. Trong Unity, chúng ta có thể import trực tiếp mô hình được thiết kế và sẽ tiếp tục được hoàn thành nếu chưa hoàn thành trong khi việc nhập chỉ được thực hiện một lần. Không giống như các công cụ khác để tạo trò chơi, chúng tôi phải xuất chúng ở một số dạng và mỗi khi chúng tôi chỉnh sửa phần mô hình, chúng ta phải import lại và điều đó mất quá nhiều thời gian.

Ngoài ra, Unity cũng cho phép chúng ta tạo mô hình trực tiếp nếu chúng ta muốn. Việc thiết lập các thuộc tính vật lý trong Unity cũng vô cùng dễ dàng và hỗ trợ nhiều

tính năng.

### *2.1.2 Sơ lược hình thành và phát triển*

Unity Core ban đầu được viết bởi Joachim Ante vào năm 2001. Sau đó, công ty được thành lập vào năm 2005 và bắt đầu với phiên bản 1.0. Năm 2007 Unity được cập nhật lên phiên bản 2.0. Unity bắt đầu hỗ trợ iPhone vào năm 2008. Tháng 6 năm 2010, Unity chính thức hỗ trợ Android và phát hành phiên bản 3.0 có hỗ trợ Android vào tháng 9 năm 2010, và bây giờ là Unity 5. Có thể thấy tốc độ phát triển của Unity khá nhanh.

- **Giải Thưởng**

Unity đã giành được nhiều giải thưởng quan trọng với những danh hiệu hàng đầu sau:

Năm 2006, Unity giành được giải "Best Use of Mac OS X Graphics" tại Apple's WWDC. Đây là lần đầu tiên một công cụ phát triển game đạt được chất lượng của giải thưởng danh giá này.

Năm 2009, Unity Technologies được lọt vào “Top 5 công ty game của năm” do Gamasutra tổ chức.

Năm 2010, Unity đoạt giải Best Engine Finalist do Develop Magazine bình chọn, giải Technology Innovation Award của Wall Street Journal ở thể loại phần mềm.

- **Khách Hàng**

Unity được trên 250.000 người đăng ký sử dụng gồm Bigpoint, Cartoon Network, Coca-Cola, Disney, Electronic Arts, LEGO, Microsoft, NASA, Ubisoft, Warner Bros, các hãng phim lớn nhỏ, các chuyên gia độc lập, sinh viên và những người đam mê.

- **Tính Năng Của Engine Unity**

Môi trường phát triển được tích hợp các chức năng kế thừa, chức năng chỉnh sửa đồ họa, chức năng kiểm tra chi tiết và đáng chú ý nhất là chức năng xem trước trò chơi trực tiếp ngay trong lúc xây dựng (live game preview).

Triển khai được trên nhiều nền tảng:

- Chương trình độc lập trên Windows và Mac OS.
- Trên web, thông qua Unity Web Player plugin cho Internet Explorer, Firefox, Safari, Opera, Chrome, cho cả Windows và Mac OS.
- Trên Mac OS Dashboard widget.

- Cho Nintendo Wii (cần mua license thêm.)
- Cho iPhone, iPad application (cần mua license thêm.)
- Cho Google Android (cần mua license thêm.)
- Cho Microsoft Xbox 360 (cần mua license thêm.)
- Cho Sony PlayStation 3 (cần mua license thêm.)
- Tài nguyên (model, âm thanh, hình ảnh, ...) được tải vào trong Unity và tự động cập nhật nếu tài nguyên có sự thay đổi. Unity hỗ trợ các kiểu định dạng từ 3DS Max, Maya, Blender, Cinema 4D và Cheetah3D.
- Graphics engine sử dụng Direct3D (Windows), OpenGL (Mac, Windows), OpenGL ES (iPhone OS), và các API khác trên Wii.
- Hỗ trợ bump mapping, reflection mapping, parallax mapping, Screen Space Ambient Occlusion v...v...

Unity Asset Server: Đây là một tính năng khá mới của Unity, theo đó Unity sẽ cung cấp một hệ thống quản lý theo dạng phiên bản cho tất cả asset và cả script. Đây là một kho chứa các tài nguyên cần thiết cho việc làm game. Khi import cũng như sửa chữa, trạng thái của asset ngay lập tức được cập nhật. Server chạy trên database opensource PostgreSQL và có thể truy cập trên cả Mac lẫn Windows, Linux. Asset Server đòi hỏi một khoản phí phụ trội là \$499 cho mỗi bản copy Unity, và một license Unity Pro.

### *2.1.3 Ưu điểm của Unity*

Chức năng cốt lõi đa dạng bao gồm: cung cấp công cụ dựng hình (kết xuất đồ họa) cho các hình ảnh 2D hoặc 3D, công cụ vật lý (tính toán và phát hiện va chạm), âm thanh, mã nguồn, hình ảnh động, trí tuệ nhân tạo, phân luồng, tạo dòng dữ liệu xử lý, quản lý bộ nhớ, dựng ảnh đồ thị và kết nối mạng. Nhờ có các engine mà công việc làm game trở nên ít tốn kém và đơn giản hơn.

Hỗ trợ đa nền tảng: Một trong các thế mạnh của Unity3D chính là khả năng hỗ trợ gần như toàn bộ các nền tảng hiện có bao gồm: PlayStation 3, Xbox 360, Wii U, iOS, Android, Windows, Blackberry 10, OS X, Linux, trình duyệt Web và cả Flash. Nói cách khác, chỉ với một gói engine, các studio có thể làm game cho bất kỳ hệ điều hành nào và dễ dàng convert chúng sang những hệ điều hành khác nhau. Đồng thời, đây cũng là giải

pháp cho các game online đa nền tảng – có thể chơi đồng thời trên nhiều hệ điều hành, phần cứng khác nhau như Web, PC, Mobile, Tablet....

Dễ sử dụng: Unity3D được built trong một môi trường phát triển tích hợp, cung cấp một hệ thống toàn diện cho các lập trình viên, từ soạn thảo mã nguồn, xây dựng công cụ tự động hóa đến trình sửa lỗi. Do được hướng đến đồng thời cả lập trình viên không chuyên và studio chuyên nghiệp, nên Unity3D khá dễ sử dụng. Hơn nữa, đây là một trong những engine phổ biến nhất trên thế giới, người dùng có thể dễ dàng tìm kiếm kinh nghiệm sử dụng của “tiền bối” trên các forum công nghệ.

Tính kinh tế cao: Unity Technologies hiện cung cấp bản miễn phí engine Unity3D cho người dùng cá nhân và các doanh nghiệp có doanh thu dưới 100.000 USD/năm. Với bản Pro, người dùng phải trả 1.500 USD/năm – một con số rất khiêm tốn so với những gì engine này mang lại.

#### *2.1.4 Các thành phần trong Unity Editor*

- Cửa sổ Hierarchy:

Tab hierarchy là nơi hiển thị các Game Object trong Scene hiện hành. Khi các đối tượng được thêm hoặc xóa trong Scence, tương ứng với các đối tượng đó trong cửa sổ Hierarchy. Tương tự trong tab Project, Hierarchy cũng có một thanh tìm kiếm giúp quản lý và thao tác với các Game Object hiệu quả hơn đặc biệt là với các dự án lớn.

Cửa sổ Game:

Đây là màn hình demo Game, là góc nhìn từ camera trong game. Thanh công cụ trong cửa sổ game cung cấp các tùy chỉnh về độ phân giải màn hình, thông số và tùy chọn bật tắt các component...

- Cửa sổ Project:

Đây là cửa sổ explorer của Unity, hiển thị thông tin của tất cả các tài nguyên (Assets) trong game của.

Cột bên trái hiển thị assets và các mục yêu thích dưới dạng cây thư mục tương tự như Windows Explorer. Khi click vào một nhánh trên cây thư mục thì toàn bộ nội dung của nhánh đó sẽ được hiển thị ở khung bên phải. Ta có thể tạo ra các thư mục mới bằng cách Right click → Create → Folder hoặc nhấn vào nút Create ở góc trên bên trái cửa sổ Project và chọn Folder. Các tài nguyên trong game cũng có thể được tạo ra bằng



cách này.

Phía trên cây thư mục là mục Favorites, giúp chúng ta truy cập nhanh vào những tài nguyên thường sử dụng.

Chúng ta có thể đưa các tài nguyên vào Favorites bằng thao tác kéo thả. Đường dẫn của thư mục tài nguyên hiện tại. Chúng ta có thể dễ dàng tiếp cận các thư mục con hoặc thư mục gốc bằng cách click chuột vào mũi tên hoặc tên thư mục.

- Cửa sổ Inspector:

Cửa sổ Inspector hiển thị chi tiết các thông tin về Game Object đang làm việc, kể cả những component được đính kèm và thuộc tính của nó. Có thể điều chỉnh, thiết lập mọi thông số và chức năng của Game Object thông qua cửa sổ Inspector.

Mọi thuộc tính thể hiện trong Inspector đều có thể dễ dàng tùy chỉnh trực tiếp mà không cần thông qua một kịch bản định trước. Tuy nhiên Scripting API cung cấp một số lượng nhiều và đầy đủ hơn do giao diện Inspector là có giới hạn.

Các thiết lập của từng component được đặt trong menu. Có thể click chuột phải, hoặc chọn icon hình bánh răng nhỏ để xuất hiện menu.

Ngoài ra Inspector cũng thể hiện mọi thông số Import Setting của asset đang làm việc như hiển thị mã nguồn của Script, các thông số animation,...

### 2.1.5 Các khái niệm cơ bản trong Unity

- GameObject:

Một đối tượng cụ thể trong game gọi là một game object, có thể là nhân vật, đồ vật nào đó.

Ví dụ: cây cối, xe cộ, nhà cửa, người,...

- Component:

Một GameObject sẽ có nhiều thành phần cấu tạo nên nó như là hình ảnh (sprite render), tập hợp các hành động (animator), thành phần xử lý va chạm (collision), tính toán vật lý (physical), mã điều khiển (script), các thành phần khác,... Mỗi thứ như vậy gọi là một component của GameObject.

- Sprite:

Là một hình ảnh 2D của một game object có thể là hình ảnh đầy đủ, hoặc có thể là một bộ phận nào đó.

- Animation:

Là tập một hình ảnh động dựa trên sự thay đổi liên tục của nhiều sprite khác nhau.

- Key Frame:

Key Frame hay Frame là một trạng thái của một animation. Có thể được tạo nên từ 1 sprite hay nhiều sprite khác nhau.

- Prefabs:

Là một khái niệm trong Unity, dùng để sử dụng lại các đối tượng giống nhau có trong game mà chỉ cần khởi tạo lại các giá trị vị trí, tỉ lệ biến dạng và góc quay từ một đối tượng ban đầu.

Ví dụ: Các đối tượng là đồng tiền trong game Mario đều có xử lý giống nhau, nên ta chỉ việc tạo ra một đối tượng ban đầu, các đồng tiền còn lại sẽ sử dụng prefabs. Hoặc khi ta lát gạch cho một cái nền nhà, các viên gạch cũng được sử dụng là prefabs.

- Sounds:

Âm thanh trong game

- Script:

Script là tập tin chứa các đoạn mã nguồn, dùng để khởi tạo và xử lý các đối tượng trong game. Trong Unity có thể dùng C#, JavaScript, BOO để lập trình Script.

- Scenes:

Quản lý tất cả các đối tượng trong một màn chơi game.

- Assets:

Bao gồm tất cả những gì phục vụ cho dự án game như sprite, animation, sound, script, scenes,...

- Camera:

Là một Game Object đặc biệt trong Scene, dùng để xác định tầm nhìn, quan sát các đối tượng khác trong game.

- Transform:

Là ba phép biến đổi tịnh tiến, quay theo các trục và phóng to thu nhỏ một đối tượng.

## 2.2 Tổng quan về Unity DevOps Version Control

Unity DevOps Version Control là hệ thống quản lý phiên bản mã nguồn được tích hợp trực tiếp trong môi trường phát triển Unity, giúp các nhóm phát triển game dễ dàng theo dõi, đồng bộ và quản lý các thay đổi trong dự án. Trước đây, công cụ này được biết đến với tên gọi Plastic SCM trước khi được Unity mua lại và tích hợp sâu vào hệ sinh thái của mình. Nó cung cấp cho chúng ta các tính năng quản lý mã nguồn hiện đại như:

- Versioning (Quản lý phiên bản): Lưu lại lịch sử tất cả các thay đổi của dự án, cho phép phục hồi hoặc so sánh các phiên bản khác nhau một cách nhanh chóng và chính xác.
- Branching và Merging (Tách nhánh và hợp nhất): Cho phép các thành viên trong nhóm làm việc đồng thời trên các tính năng riêng biệt mà không ảnh hưởng đến phiên bản chính, sau đó dễ dàng hợp nhất lại khi hoàn thành.
- Conflict Resolution (Giải quyết xung đột): Hỗ trợ phát hiện và xử lý các xung đột khi có nhiều người cùng chỉnh sửa một file, giúp đảm bảo tính nhất quán của dự án.
- Cloud Integration (Tích hợp đám mây): Unity DevOps Version Control hỗ trợ lưu trữ và đồng bộ dữ liệu trên đám mây, giúp các thành viên có thể truy cập và làm việc từ xa một cách tiện lợi.

## 2.3 Tổng quan về C#



Hình 2. 2. Ảnh logo của ứng dụng C#

C# là một ngôn ngữ lập trình hướng đối tượng được phát triển bởi Microsoft, nổi bật với cú pháp rõ ràng, dễ đọc và khả năng xử lý mạnh mẽ các tác vụ phức tạp. Trong lĩnh vực phát triển game, đặc biệt là với engine Unity, C# là ngôn ngữ chính được sử dụng để viết các kịch bản (scripts) điều khiển hành vi của các đối tượng, logic gameplay, tương tác người chơi, hệ thống AI, vật lý và nhiều thành phần khác. Sự phổ biến của C# trong Unity bắt nguồn từ nhiều ưu điểm vượt trội, bao gồm:

- Tính hướng đối tượng (OOP): C# hỗ trợ đầy đủ các tính năng của lập trình hướng đối tượng như kế thừa, đa hình, đóng gói, giúp tổ chức và tái sử dụng mã nguồn hiệu quả, dễ bảo trì và mở rộng.
- Đa nền tảng: Với Unity, mã C# có thể biên dịch và chạy trên nhiều hệ điều hành và thiết bị khác nhau mà không cần thay đổi nhiều về mặt code, tạo điều kiện thuận lợi cho việc phát triển game đa nền tảng.
- Thư viện chuẩn phong phú: C# có hệ thống thư viện chuẩn mạnh mẽ hỗ trợ xử lý chuỗi, toán học, quản lý bộ nhớ, đa luồng và nhiều chức năng nâng cao khác, giúp lập trình viên tận dụng tối đa nguồn lực có sẵn mà không phải xây dựng lại từ đầu.
- Tích hợp chặt chẽ với Unity API: Unity cung cấp một bộ API phong phú viết sẵn

bằng C#, giúp lập trình viên dễ dàng thao tác với các thành phần trong game như đối tượng, vật lý, âm thanh, hình ảnh, UI và các hiệu ứng đặc biệt. Việc viết script bằng C# trong Unity là cách thức chính để điều khiển mọi hành vi và trạng thái của game.

- Hiệu suất tốt: Mặc dù là ngôn ngữ bậc cao, C# vẫn có hiệu suất đủ mạnh để xử lý các tác vụ game real-time với độ trễ thấp, đặc biệt khi được tối ưu hợp lý.
- Cộng đồng lớn và tài liệu phong phú: Với hàng triệu lập trình viên trên toàn cầu sử dụng, C# sở hữu một cộng đồng rất năng động, cùng với rất nhiều tài liệu học tập, ví dụ mẫu, plugin và công cụ hỗ trợ, tạo điều kiện thuận lợi cho cả người mới và chuyên gia.

### *2.3.1 Các ưu điểm*

- Có tính phổ biến cao, C# là một ngôn ngữ thuộc nền tảng .NET của Microsoft, một trong những nền tảng lập trình hàng đầu hiện nay. .NET cung cấp cho các lập trình viên C# một thư viện phong phú và đa dạng, hỗ trợ nhiều chức năng và công nghệ khác nhau. Ví dụ, có thể sử dụng ASP.NET để xây dựng các ứng dụng web động và hiệu quả, Xamarin để tạo các ứng dụng di động chạy trên nhiều nền tảng khác nhau, hay Unity để phát triển các trò chơi 2D và 3D tuyệt vời.
- Được nhiều người yêu thích, C# là một ngôn ngữ dễ học và dễ sử dụng, với cú pháp rõ ràng và logic. C# cũng là một ngôn ngữ hướng đối tượng, cho phép thiết kế và tổ chức mã nguồn theo cách trực quan và hiệu quả. C# còn có nhiều tính năng tiên tiến và hấp dẫn, như lambda expressions, LINQ, async/await, generics, delegates, events và nhiều hơn nữa. Không có gì ngạc nhiên khi C# được xếp vào top 10 ngôn ngữ lập trình được yêu thích nhất theo khảo sát của Stack Overflow.
- C# là một ngôn ngữ dễ học và sử dụng, C# có cú pháp rõ ràng và đơn giản, dựa trên ngôn ngữ C và Java. C# cũng hỗ trợ các tính năng lập trình hướng đối tượng, như kế thừa, đa hình, giao diện và trừu tượng. Có thể bắt đầu học C# với những kiến thức cơ bản về lập trình và dần nâng cao kỹ năng của mình.
- C# là một ngôn ngữ linh hoạt và sáng tạo, C# cho phép viết từ các ứng dụng máy tính để bàn, web, di động, đến các ứng dụng thực tế ảo, trí tuệ nhân tạo và máy học. C# cũng liên tục cập nhật và cải tiến với các phiên bản mới, mang lại nhiều

tính năng mới và thú vị, như LINQ, async/await, lambda expressions và generics.

### 2.3.2 Các nhược điểm

- C# có thể gặp khó khăn khi tương tác với các thành phần được viết bằng các ngôn ngữ lập trình cấp thấp, như C hay C++. Có thể sử dụng các kỹ thuật như P/Invoke hay COM Interop để gọi các hàm hay sử dụng các đối tượng từ các ngôn ngữ khác, nhưng cũng có thể gây ra lỗi hoặc hiệu suất kém.
- C# có thể không phù hợp cho một số loại ứng dụng yêu cầu hiệu năng cao, như ứng dụng xử lý đồ họa, âm thanh hay video. C# chạy trên máy ảo CLR (Common Language Runtime), nên có thể bị ảnh hưởng bởi các yếu tố như thu gom rác (garbage collection), biên dịch thời gian chạy (JIT compilation), v.v.
- C# có thể không được hỗ trợ đầy đủ trên một số nền tảng khác nhau. Mặc dù có các công cụ như .NET Core, Mono hay Xamarin để biên dịch và chạy các ứng dụng C# trên các nền tảng khác, nhưng cũng có thể gặp phải một số vấn đề về tính tương thích, tính năng hay bảo mật.
- C# có thể không dễ học cho một số người mới bắt đầu lập trình. C# có cú pháp rõ ràng, nhưng cũng có nhiều quy tắc và ngoại lệ. C# cũng có nhiều tính năng và khái niệm phức tạp, như đa luồng (Multithreading), LINQ (Language Integrated Query), async/await,...

### 2.3.3 Mục tiêu của C#

- Ngôn ngữ được dự định là một ngôn ngữ lập trình đơn giản, hiện đại, hướng đến nhiều mục đích sử dụng, và là một ngôn ngữ lập trình hướng đối tượng.
- Ngôn ngữ và việc triển khai đáp ứng các nguyên tắc của ngành kỹ thuật phần mềm như kiểm tra chặt chẽ kiểu dữ liệu, kiểm tra giới hạn mảng, phát hiện các trường hợp sử dụng các biến chưa có dữ liệu, và tự động thu gom rác. Tính mạnh mẽ, sự bền bỉ, và năng suất của việc lập trình là rất quan trọng đối với ngôn ngữ này.
- Ngôn ngữ sẽ được sử dụng để phát triển các thành phần của phần mềm theo hướng thích hợp cho việc triển khai trong các môi trường phân tán.
- Khả năng di chuyển (portability) là rất quan trọng, đặc biệt là đối với những lập

trình viên đã quen với C và C++.

- Hỗ trợ quốc tế hóa (i18n).
- Ngôn ngữ sẽ được thiết kế để phù hợp với việc viết các ứng dụng cho cả hai hệ thống: hosted và nhúng, từ các phần mềm quy mô lớn, đến các phần mềm chỉ có các chức năng đơn giản.
- Mặc dù các ứng dụng C# có tính kinh tế đối với các yêu cầu về bộ nhớ và chế độ xử lý, ngôn ngữ này không cạnh tranh trực tiếp về hiệu năng và kích thước đối với ngôn ngữ C hoặc assembly.

#### 2.3.4 Ứng dụng của C#

- Phát triển web backend (ASP.NET MVC, ASP.NET core, Web API, GraphQL, gRPC, Blazor server, Uno platform, Mono).
- Phát triển web front end (Blazor WebAssembly, Uno platform).
- Phát triển desktop app (Winform, WPF, UWP, WinUI, Mono, Uno, MAUI, Blazor desktop...).
- Phát triển game 2D, 3D đa nền tảng (Game engine: Unity, Monogame, Godot, Stride, CryEngine, Flax Engine, Evergreen, NeoAxis, XNA ..).
- Phát triển thực tế ảo (VR), thực tế tăng cường (AR), thực tế hỗn hợp (MR) (HoloLens, Unity, CryEngine, Oculus quest..).
- Phát triển ứng dụng đồ họa 2D, 3D đa nền tảng (2D: SkiaSharp, ImageSharp...; 3D: OpenTK, SharpDX, SharpVulkan, Vulkan.NET, Veldrid, Silk.NET, Helix Toolkit, Aspose..).
- Phát triển mobile app, IOS native, Android native (Xamarin, MAUI, Uno platform).
- Phát triển đám mây (Azure, AWS, Google Cloud...).
- Học máy và trí tuệ nhân tạo (ML.Net, TensorFlow, csSharp..).
- Data science, bigdata (csSharp, Apache Spark).
- Blockchain (NEO, Stratis).
- Microservices and containers.
- Internet of thing (IoT, 5G).

- Hệ thống nhúng (Raspberry pi, PLC).

### 2.3.5 Đặc điểm của C#

C# theo một hướng nào đó, là ngôn ngữ lập trình phản ánh trực tiếp nhất đến .NET Framework mà tất cả các chương trình.NET chạy, và nó phụ thuộc mạnh mẽ vào framework này. Mọi dữ liệu cơ sở đều là đối tượng, được cấp phát và hủy bỏ bởi trình dọn rác Garbage-Collector (GC), và nhiều kiểu trừu tượng khác chẳng hạn như class, delegate, interface, exception... phản ánh rõ ràng những đặc trưng của.NET runtime.

So sánh với C và C++, ngôn ngữ này bị giới hạn và được nâng cao ở một vài đặc điểm nào đó, nhưng không bao gồm các giới hạn sau đây:

- Các con trỏ chỉ có thể được sử dụng trong chế độ không an toàn. Hầu hết các đối tượng được tham chiếu an toàn, và các phép tính đều được kiểm tra tràn bộ đệm. Các con trỏ chỉ được sử dụng để gọi các loại kiểu giá trị; còn những đối tượng thuộc bộ gom rác (garbage-collector) thì chỉ được gọi bằng cách tham chiếu.
- Các đối tượng không thể được giải phóng tường minh.
- Chỉ có đơn kế thừa, nhưng có thể cài đặt nhiều interface trừu tượng (abstract interfaces). Chức năng này làm đơn giản hóa sự thực thi của thời gian thực thi.
- C# thì an-toàn-kiểu (typesafe) hơn C++.
- Cú pháp khai báo mảng khác nhau ("int[] a = new int[5]" thay vì "int a[5]").
- Kiểu thứ tự được thay thế bằng tên miền không gian (namespace).
- C# không có tiêu bản.
- Có thêm Properties, các phương pháp có thể gọi các Properties để truy cập dữ liệu.
- Có reflection.
- Lập trình chức năng.
- Hỗ trợ kiểu động.

### 2.3.6 Đặc trưng của C#

- C# là ngôn ngữ lập trình mã nguồn mở, vì vậy C# là miễn phí với tất cả mọi người, đồng thời mọi người đều có thể cùng tham gia phát triển, đề xuất thiết kế ngôn ngữ C#



- C# là ngôn ngữ đa nền tảng vì vậy có thể biên dịch trên nhiều nền tảng máy tính khác nhau (Windows, Linux, MacOS)
- C# có hiệu suất cao và tốc độ thực thi nhanh do sử dụng trình biên dịch trung gian (CLR), điểm cộng nữa là tốc độ phát triển phần mềm nhanh chóng so với đa số các ngôn ngữ hiện tại.
- C# có IDE Visual Studio cùng nhiều plug-in vô cùng mạnh mẽ. ngoài ra có thể viết C# bằng bất kỳ text editor nào khác như Visual Studio Code, Vim, Netbeam...
- C# có cấu trúc khá gần gũi với các ngôn ngữ lập trình truyền thống, song cũng được bổ sung các yếu tố mang tính hiện đại nên dễ dàng tiếp cận cho người mới học và học nhanh với C#.
- C# có cộng đồng nhà phát triển vô cùng lớn mạnh.
- C# được phát triển và cải tiến không ngừng với tần suất 1 phiên bản/ 1 năm, đáp ứng các mong muốn cải thiện, cải tiến cho phù hợp với nhu cầu công nghệ của các nhà phát triển.
- C# có tài liệu tham khảo và hướng dẫn vô cùng phong phú và chất lượng, đồng thời có các buổi hội thảo giới thiệu tính năng mới và định hướng phát triển ngôn ngữ trong tương lai.
- C# và .NET được đánh giá là có design tốt, vì vậy cú pháp và logic rất nhất quán, mã nguồn C# dễ đọc và mở rộng.
- C# được thiết kế và phát triển bởi Microsoft nên rất được Microsoft quan tâm và hỗ trợ.

## 2.4 Tổng quan về thể loại game Platformer 2D

Game platformer 2D là một trong những thể loại game điện tử cổ điển và phổ biến nhất, xuất hiện từ những năm 1980 và đến nay vẫn giữ được sức hút mạnh mẽ nhờ gameplay đơn giản nhưng sâu sắc, lôi cuốn người chơi bằng sự kết hợp giữa kỹ năng điều khiển và khả năng quan sát, phán đoán. Trong đó nhân vật chính di chuyển chủ yếu trên một mặt phẳng hai chiều là trục X và trục Y và phải vượt qua các chướng ngại vật, kẻ thù, bẫy hoặc các câu đố để tiến tới mục tiêu cuối cùng. Thường các màn chơi được thiết kế theo dạng level based hoặc thể giới mở dạng side-scrolling.

Game thuộc thể loại này thường có các điểm nổi bật như:

- Điều khiển đơn giản, phản xạ nhanh: Người chơi thường điều khiển nhân vật bằng các phím di chuyển, nhảy, tấn công hoặc né tránh. Gameplay đòi hỏi phản xạ nhanh nhạy và kỹ năng phối hợp nhịp nhàng giữa các hành động.
- Thiết kế level đa dạng và thử thách: Mỗi màn chơi chứa nhiều thử thách như khoảng cách nhảy, bẫy, địa hình phức tạp, kẻ thù với các hành vi khác nhau, buộc người chơi phải tính toán và lên chiến lược hợp lý để vượt qua.
- Yếu tố khám phá: Nhiều game platformer tích hợp các yếu tố khám phá, bí mật hoặc các vật phẩm ẩn, tạo thêm chiều sâu và sự hấp dẫn cho trải nghiệm.
- Chiến đấu và boss: Ngoài việc di chuyển và nhảy, game thường kết hợp yếu tố chiến đấu với quái vật, miniboss và boss cuối màn, đòi hỏi người chơi phải vận dụng kỹ năng chiến đấu cũng như tư duy chiến thuật.
- Đồ họa và âm thanh phong cách đa dạng: Platformer 2D có thể sử dụng nhiều phong cách đồ họa khác nhau từ pixel art cổ điển, vẽ tay (hand-drawn) cho đến hình ảnh vector sắc nét, đi cùng với nền âm thanh và nhạc nền phù hợp tạo nên không khí cho game.

## 2.5 Tổng quan về cấu trúc game engine

Engine Unity hỗ trợ cho chúng ta UnityAPI để viết script game. UnityAPI là API lập trình game trong Unity rất mạnh. UnityAPI chứa các đối tượng và phương thức hỗ trợ hầu hết các đối tượng và các loại thành phần trong Unity.

Trong một scene thường có nhiều đối tượng game. Mỗi đối tượng này có thể có hoặc không có đoạn script nào gắn lên đó. Nếu muốn gắn script vào đối tượng, ta bắt buộc phải kế thừa class đó từ lớp MonoBehaviour của UnityAPI và tên class phải trùng với tên file script. Mỗi script khi gắn lên đối tượng game đều được đối tượng game xem như một thành phần bên trong và được cấp phát vùng nhớ khi chạy game.

Bên trong UnityAPI chứa rất nhiều lớp hỗ trợ lập trình game, trong đó có một số lớp quan trọng như:

- MonoBehaviour: tất cả các script muốn gắn vào một đối tượng game bắt buộc phải kế thừa từ lớp này.
- GameObject: lớp cha của tất cả các thực thể trong scene.

- Component: lớp cha của tất cả các thành phần có thể gắn vào đối tượng.
- Transform: giúp thay đổi vị trí, xoay, biến đổi tỉ lệ mô hình.
- Input: hỗ trợ lập trình với chuột, cảm ứng đa điểm, cảm biến gia tốc.
- Camera: giúp lập trình camera.
- Light: giúp tạo ánh sáng trong game.
- Projector: giúp chiếu texture lên bề mặt vật thể.
- ParticleEmitter: hỗ trợ tạo các hiệu ứng particle đẹp mắt.
- Audio: hỗ trợ lập trình với âm thanh.
- Animation: chạy chuyển động của mô hình nhân vật.
- Rigidbody: giúp tạo hiệu ứng vật lý liên quan đến trọng lực như bóng nảy, lăn, ...
- CharacterController: giúp điều khiển nhân vật di chuyển theo độ cao địa hình.
- Collider: hỗ trợ lập trình va chạm giữa các vật thể.
- GUI: giúp lập trình giao diện người dùng trên Unity.

## 2.6 So sánh với game cùng thể loại

### 2.6.1 So sánh với game *Hollow Knight*

Thể loại: Metroidvania, platformer hành động

Đặc điểm nổi bật: Thế giới mở phức tạp, phong cách nghệ thuật vẽ tay u tối, combat đa dạng.

- Giống:
  - Phong cách u tối, giàu cảm xúc, nhiều lớp ẩn dụ.
  - Kết hợp phiêu lưu, chiến đấu, khám phá.
  - Boss có tính biểu tượng cao, mang ý nghĩa về thế giới hoặc nhân vật.
- Khác:
  - Hollow Knight là game open-world dạng metroidvania, có hệ thống bản đồ lớn, nâng cấp kỹ năng và khả năng mở khóa vùng mới bằng kỹ năng mới.
  - The Last Gleam đi theo hướng màn chơi tuyến tính, tập trung vào trình diễn câu chuyện ngắn gọn nhưng đủ ấn tượng trong từng khu vực địa hình.

### 2.6.2 So sánh với game *Dead Cells*

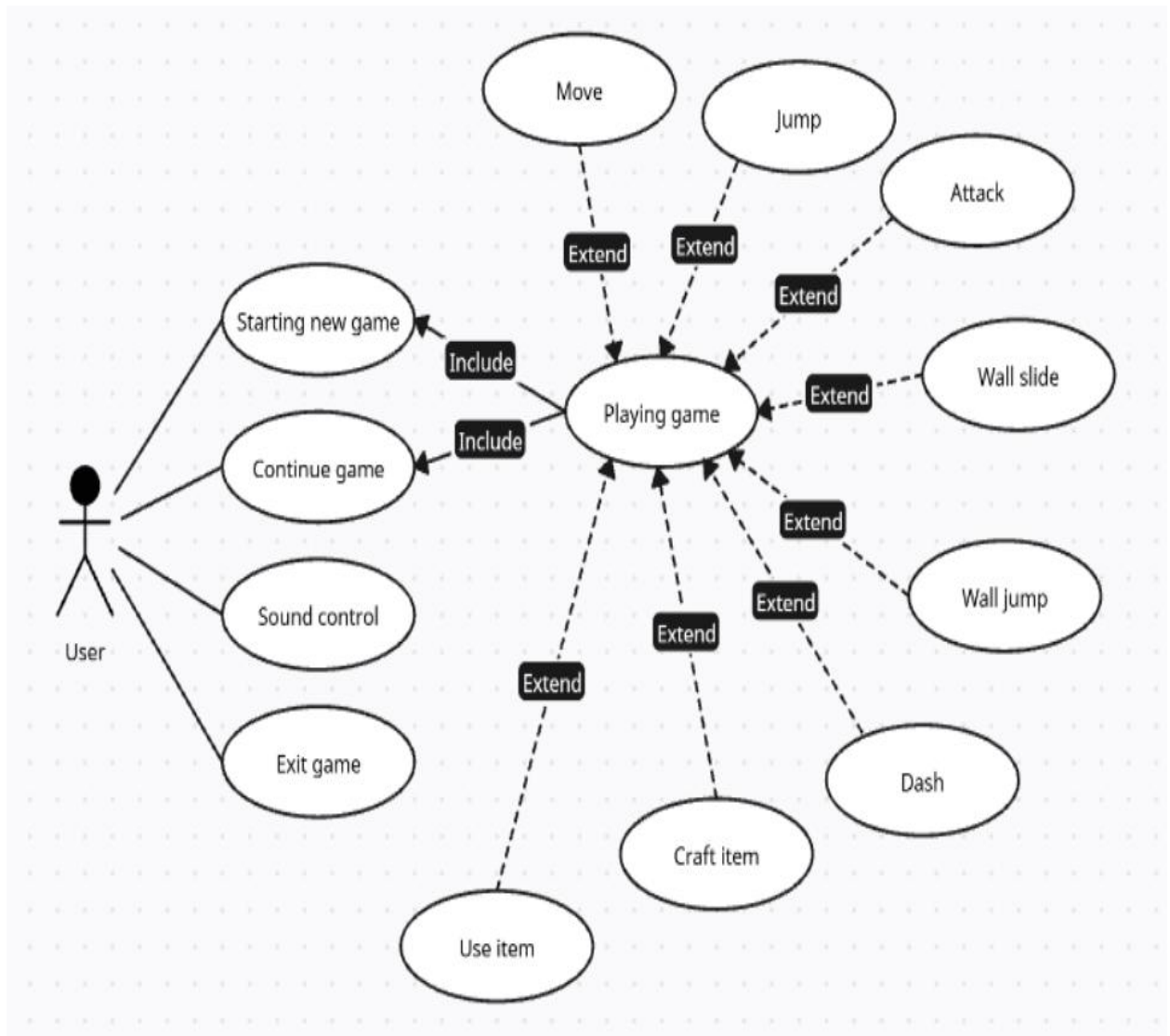
Thể loại: Roguelike, platformer hành động

Đặc điểm nổi bật: Combat tốc độ cao, chuyển động mượt mà, thiết kế màn chơi ngẫu nhiên, chơi lại liên tục.

- Giống:
  - Nhấn mạnh vào chiến đấu và phản xạ người chơi.
  - Hiệu ứng đòn đánh rõ ràng, cảm giác tay tốt.
  - Kết hợp ánh sáng, chuyển động và màu sắc để truyền tải không khí căng thẳng, gay cấn.
- Khác:
  - *Dead Cells* là game roguelike, có yếu tố ngẫu nhiên cao, hệ thống vũ khí thay đổi qua mỗi lượt chơi.
  - *The Last Gleam* là game có kịch bản cố định.

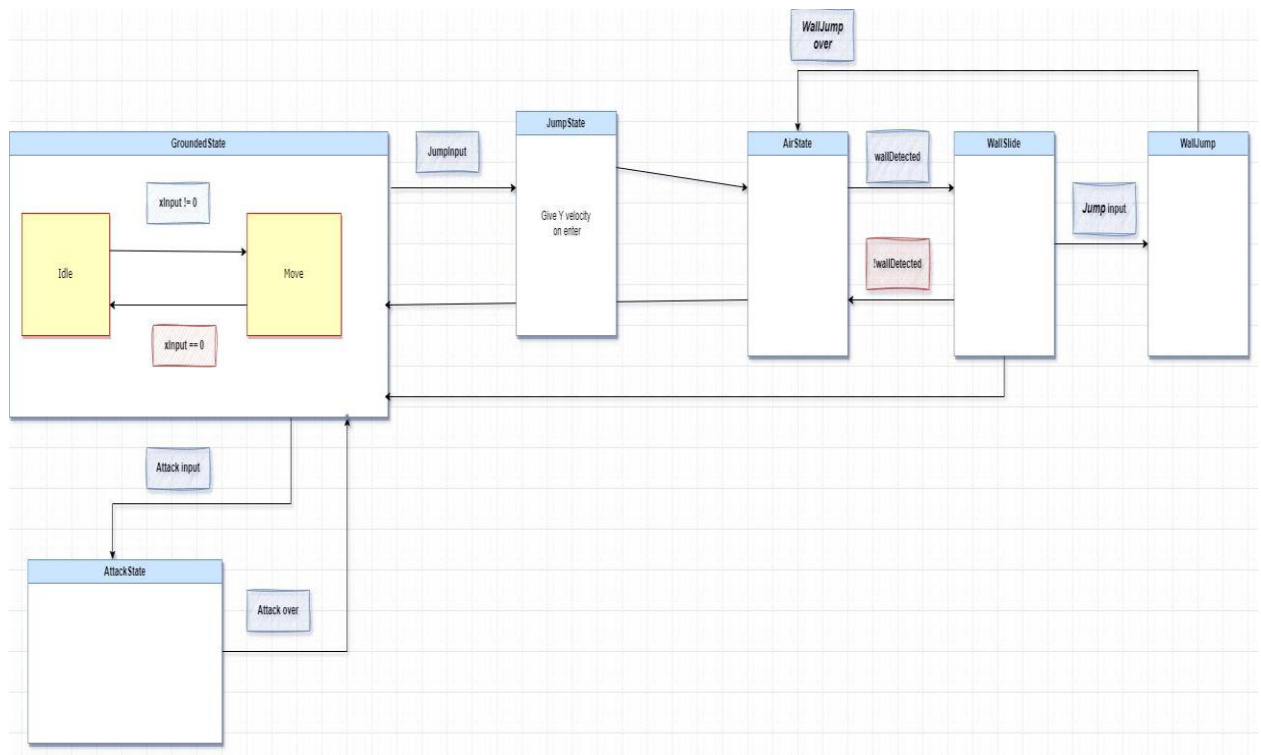
## CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

### 3.1 Mô hình Use-case tổng quát



Hình 3. 1. Mô hình Use-case tổng quát

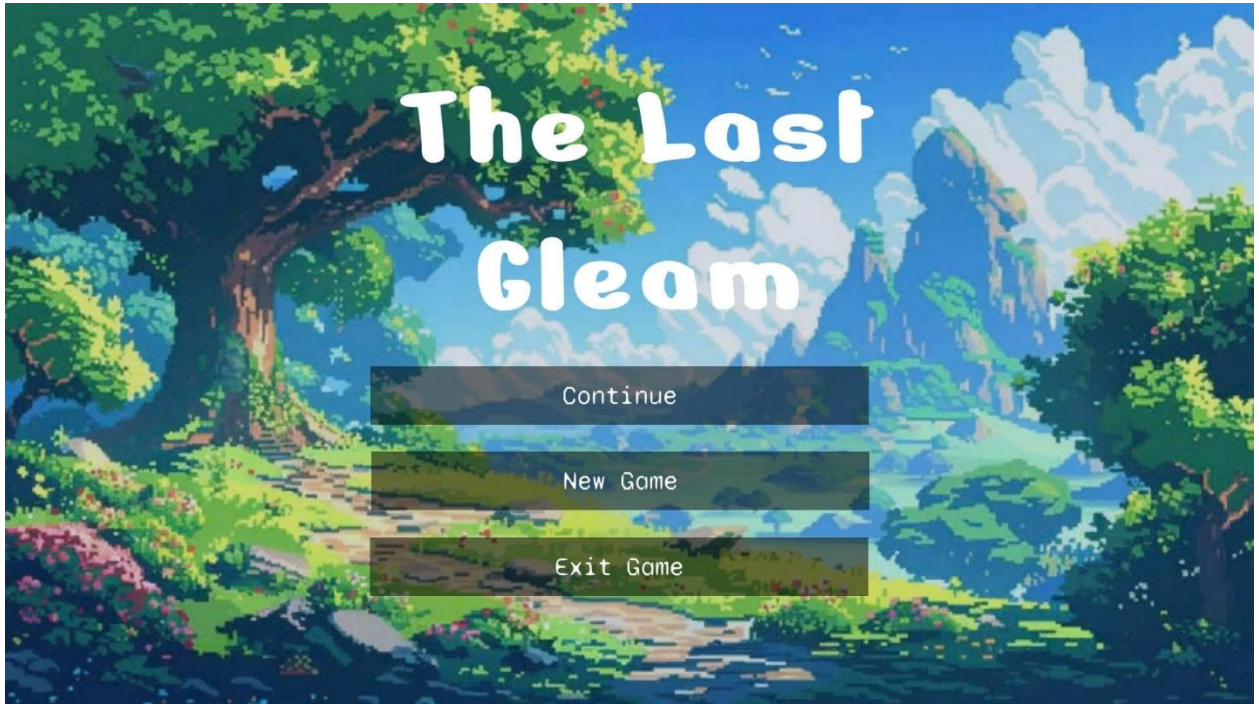
### 3.2 Mô hình FSM (Finite State Machines)



Hình 3. 2. Mô hình FSM

### 3.3 Các giao diện của game

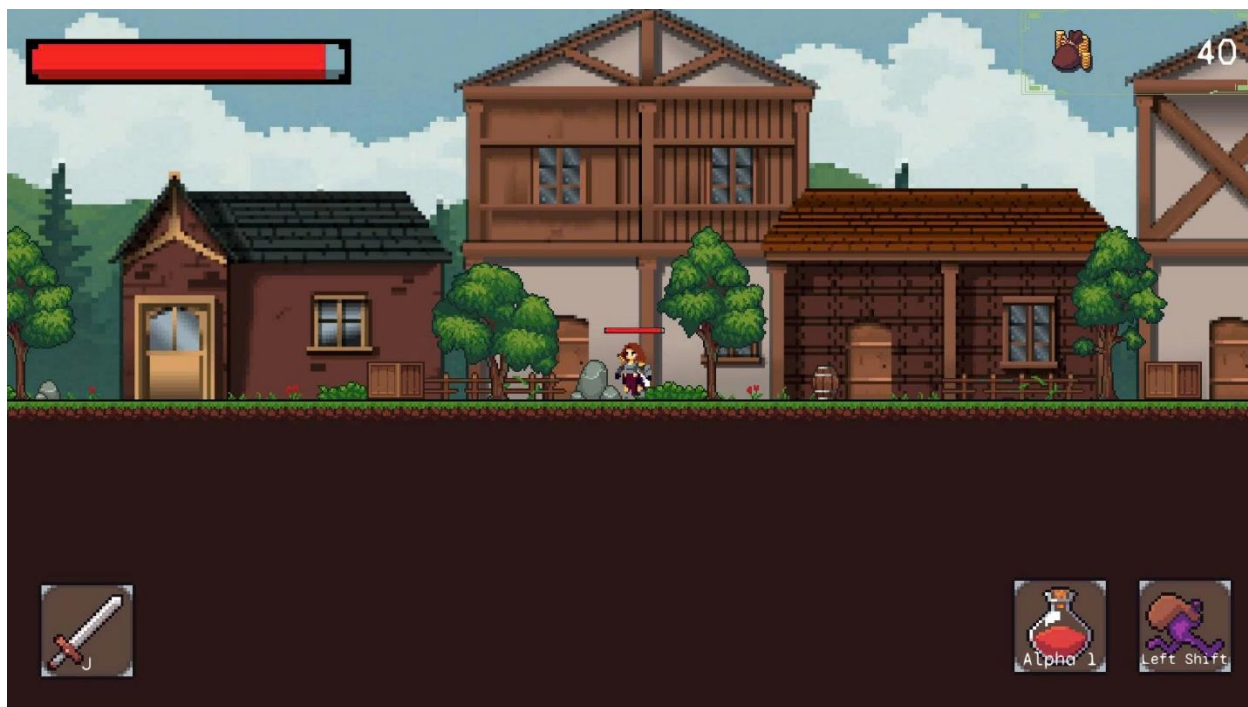
#### 3.3.1 Giao diện khi vừa mở game



*Hình 3. 3. Giao diện khi vừa mở game*

Khi vừa khởi động, game sẽ hiển thị ba lựa chọn dưới dạng button, người chơi sẽ có thể chọn tiếp tục chơi hoặc chọn chơi mới nếu là lần đầu chơi hoặc muốn chơi lại từ đầu. Cuối cùng là lựa chọn thoát game.

### 3.3.2 Giao diện khi vừa vào game



Hình 3. 4. Giao diện khi vừa vào game

Khi vào màn chơi chính, màn hình sẽ hiển thị toàn bộ chi tiết màn chơi bao gồm không gian bối cảnh và quái vật đồng thời cũng hiển thị thêm thanh máu, số tiền để người chơi có thể quan sát lượng máu và số tiền mình đang có một cách trực quan và chi tiết hơn.



### 3.3.3 Giao diện option



Hình 3. 5. Giao diện option

Ở giao diện option, game cho phép người chơi tùy chỉnh mức độ âm lượng của game bao gồm âm lượng nền và âm lượng hiệu ứng giúp nâng cao trải nghiệm người chơi hiệu quả hơn.

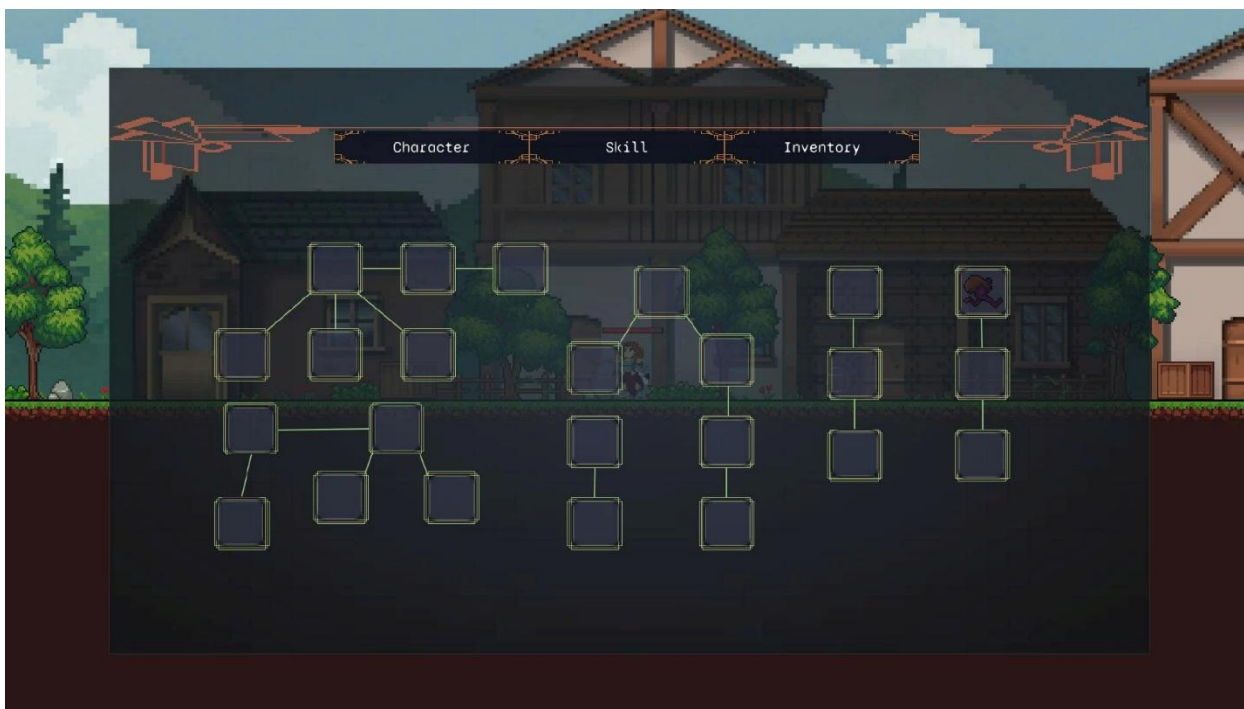
### 3.3.4 Giao diện thông tin, chỉ số nhân vật hiện tại



Hình 3. 6. Giao diện thông tin chỉ số nhân vật

Giao diện thể hiện thông tin chi tiết về chỉ số của nhân vật giúp người chơi theo dõi và định hướng phát triển sức mạnh. Các chỉ số được bố trí rõ ràng theo hai cột: Bên trái là nhóm chỉ số cơ bản và phòng thủ như máu (Health), giáp (Armor), né tránh (Evasion), thể lực (Vitality), nhanh nhẹn (Agility), kháng phép (Magic Res) và sát thương vật lý cơ bản (Damage). Bên phải là các chỉ số thiên về tấn công và năng lực phép thuật, bao gồm sức mạnh (Strength), trí tuệ (Intelligence), tỉ lệ và sát thương chí mạng (Crit Rate, Crit Dame), cùng ba hệ sát thương nguyên tố: lửa, băng và sét (Fire, Ice, Lighting Damage). Ở thời điểm hiện tại, nhân vật đang trong trạng thái khởi đầu, với 100 máu và 5 sát thương cơ bản, chưa có trang bị hay điểm cộng thêm nào. Giao diện được nhóm thiết kế đơn giản để người chơi dễ dàng theo dõi, tạo điều kiện để người chơi tập trung vào hành trình phát triển nhân vật theo lối chơi riêng của mình.

### 3.3.5 Giao diện skill tree (cây kỹ năng)



Hình 3. 7. Giao diện cây kỹ năng

Hệ thống cây kỹ năng trong game là một phần không thể thiếu đối với game thuộc thể loại phiêu lưu nhập vai. Tại giao diện này người chơi có thể nâng cấp kỹ năng theo từng nhánh giúp định hình phong cách chiến đấu và phát triển nhân vật theo hướng cá nhân hóa bằng cách sử dụng điểm Soul kiếm được thông qua việc giết quái vật để mở khóa các kỹ năng.

### 3.3.6 Giao diện túi đồ



Hình 3. 8. Giao diện túi đồ

Giao diện túi đồ của nhân vật là nơi lưu trữ toàn bộ vật phẩm, nguyên liệu, và tài nguyên mà người chơi thu thập được trong suốt hành trình. Trong tab Material, người chơi sẽ có thể quan sát toàn bộ các loại nguyên liệu hiện có. Mỗi ô thể hiện biểu tượng vật phẩm và số lượng tương ứng, ví dụ như da thú, sắt, đá quý, vàng, gỗ,... Những nguyên liệu này đóng vai trò quan trọng trong việc chế tạo trong game. Bên phải là khung chi tiết nguyên liệu, nơi người chơi có thể chọn một vật phẩm để xem thông tin mô tả chi tiết của vật phẩm đó.

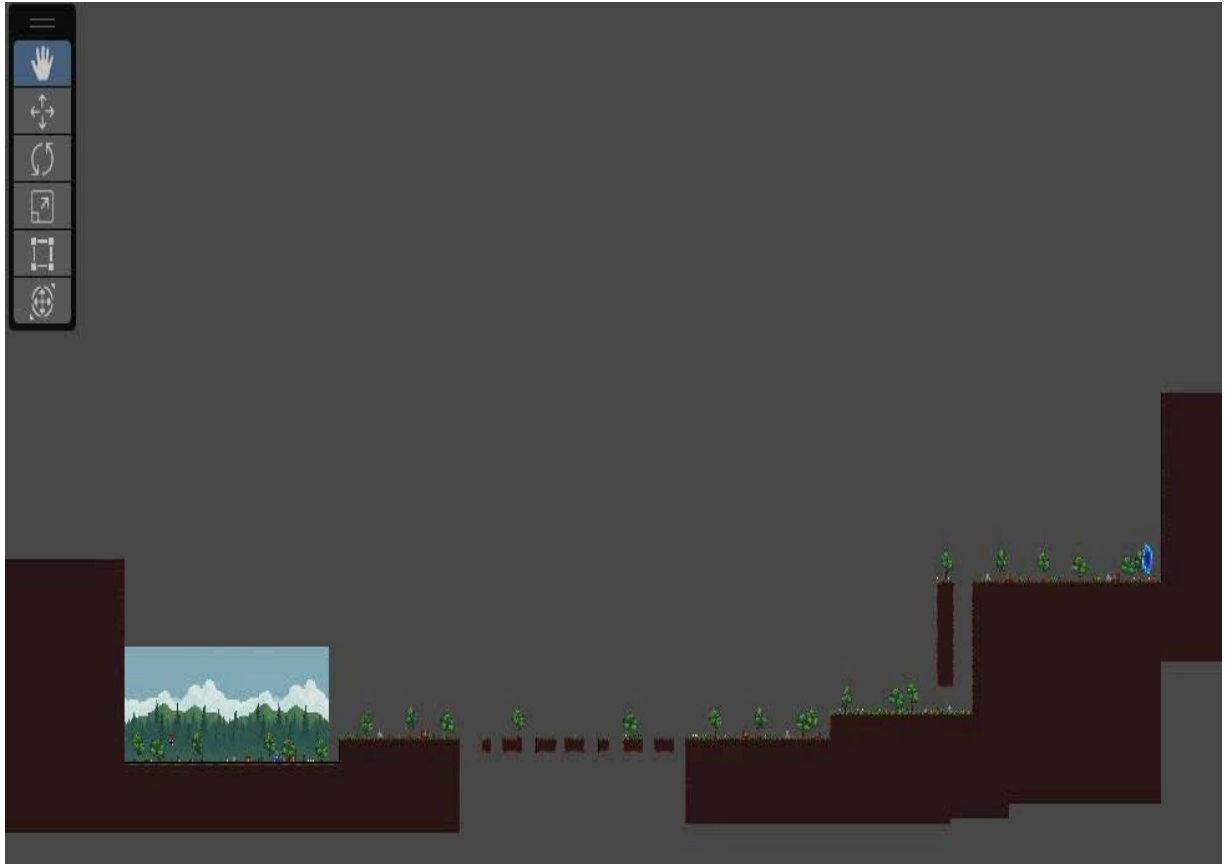
### 3.3.7 Giao diện craft vật phẩm trang bị



Hình 3. 9. Giao diện craft vật phẩm trang bị

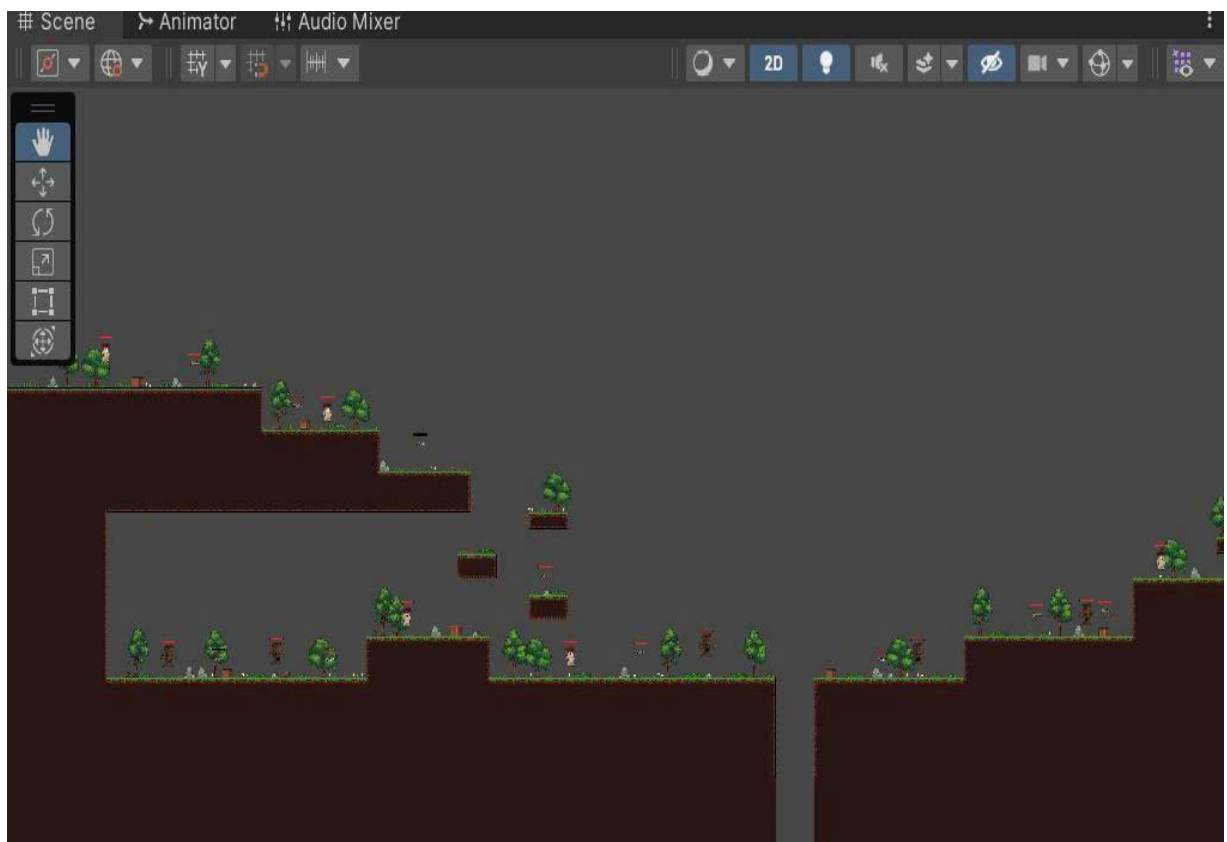
Giao diện này thể hiện hệ thống chế tạo vật phẩm trong game. Tại giao diện này nơi người chơi có thể sử dụng các nguyên liệu thu thập được để tạo ra vũ khí, trang bị hoặc vật phẩm hỗ trợ. Bên trái giao diện sẽ hiển thị danh sách các bản công thức có thể chế tạo. Mỗi ô tương ứng với một vật phẩm cụ thể, như: Wooden Sword, Iron Sword, Thunder Sword, hay Dagger,... Các biểu tượng được thể hiện giúp người chơi dễ dàng nhận diện từng loại vật phẩm và có thể nhấn vào từng vật phẩm để xem chi tiết yêu cầu nguyên liệu. Khu vực bên phải là khu vực chi tiết chế tạo. Khi người chơi chọn một vật phẩm, hệ thống sẽ hiển thị hình ảnh vật phẩm và thông tin chỉ số của vật phẩm đó cũng như hiển thị danh sách vật phẩm nguyên liệu. Nếu đủ vật phẩm nguyên liệu, người chơi có thể bấm “Craft” để hoàn tất tạo ra vật phẩm mong muốn.

### 3.4 Một số giao diện khác trong game

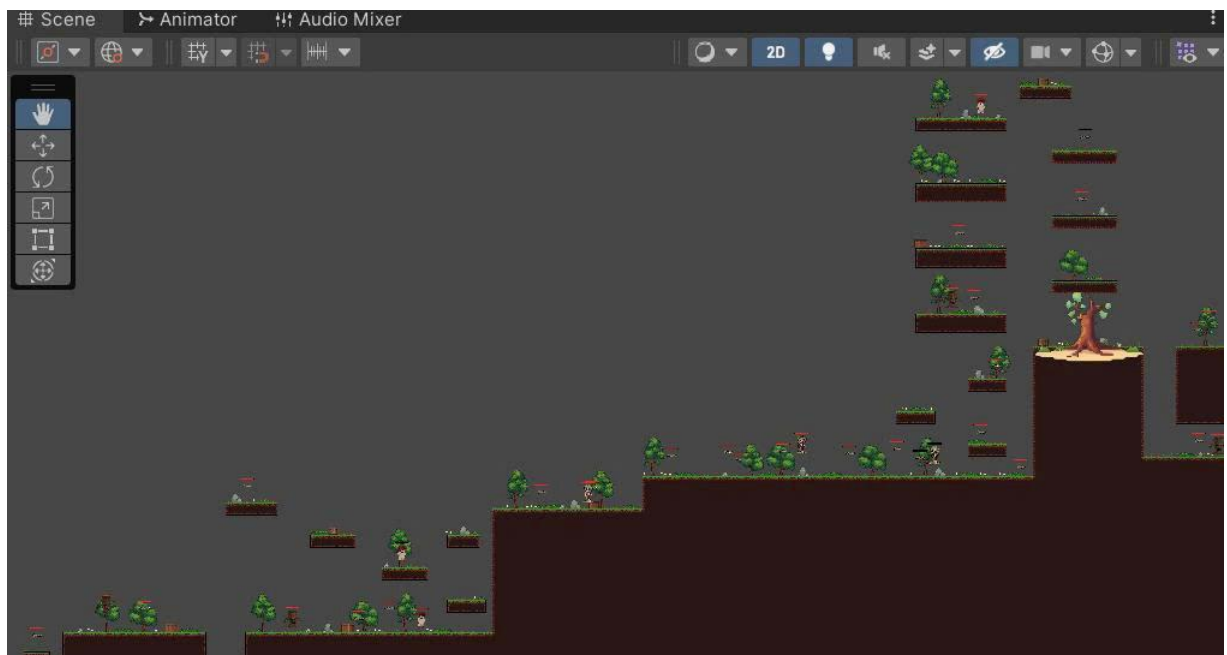


*Hình 3. 10. Giao diện map hướng dẫn*





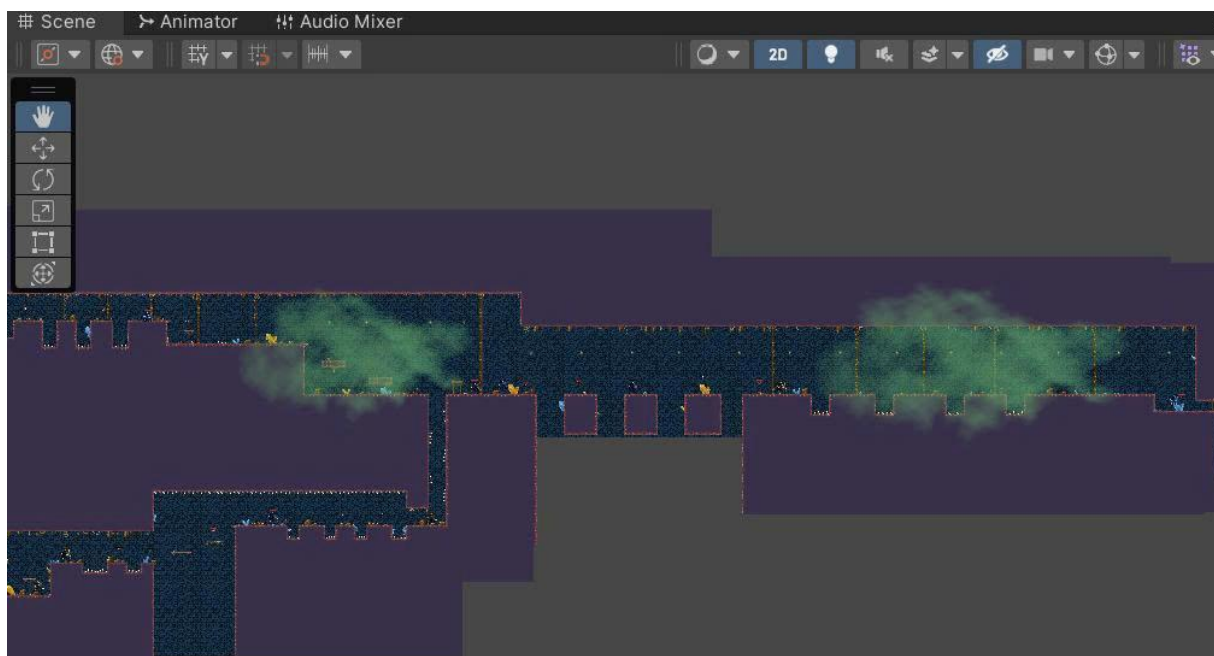
*Hình 3. 11. Giao diện map rừng 1*



*Hình 3. 12. Giao diện map rừng 2*

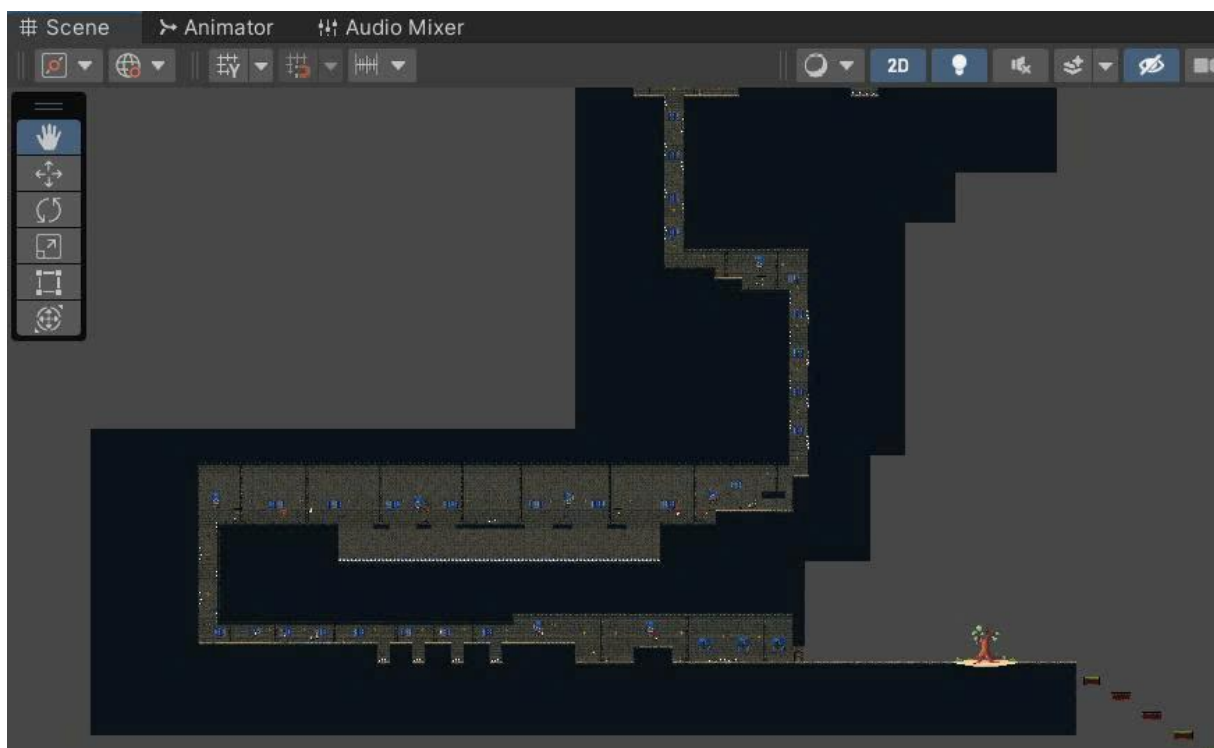


*Hình 3. 13. Giao diện map hàng 1*

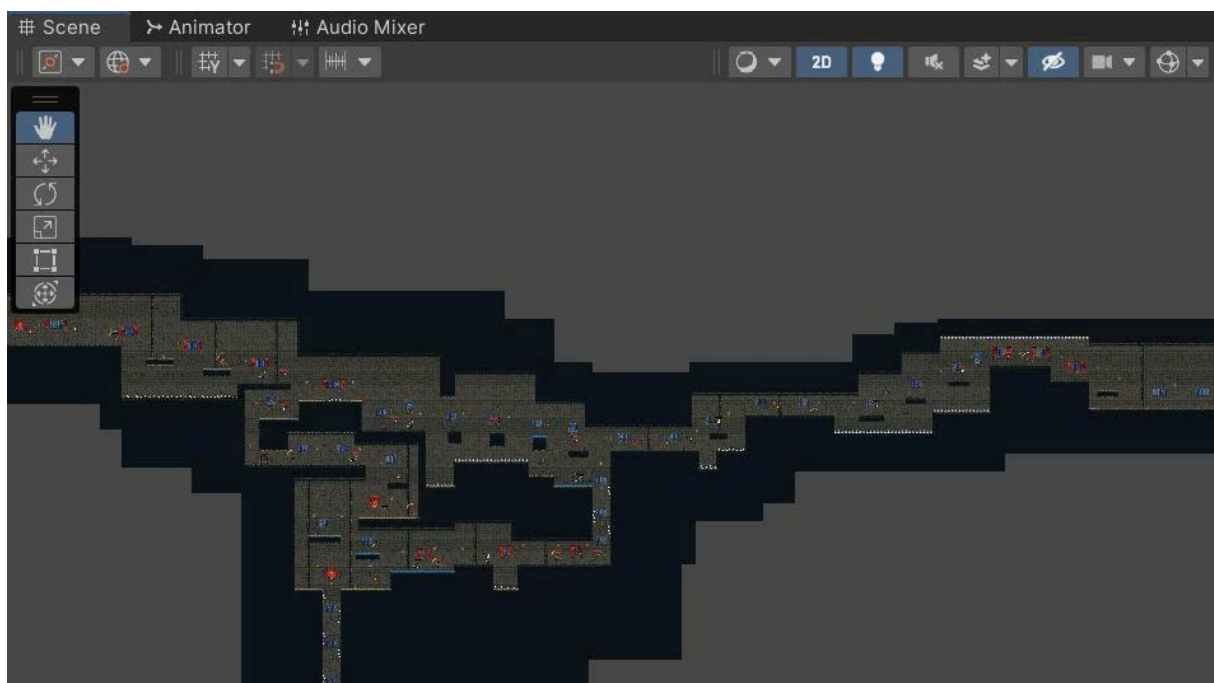


*Hình 3. 14. Giao diện map hàng 2*





*Hình 3. 15. Giao diện map lâu đài 1*



*Hình 3. 16. Giao diện map lâu đài 2*

### 3.5 Các State xử lý logic của nhân vật chính – Ember



Hình 3. 17. Các State xử lý logic của nhân vật chính

Việc thiết kế các State như thế này giúp dễ quản lý và chuyển đổi hơn, tránh được các trường hợp quá tải bộ nhớ trong khi chơi, hay là mất khung hình do thiết bị không thể Render nổi hình ảnh của nhân vật, đồng thời hạn chế việc kẹt khung hình đến mức tối thiểu nhất do sự rõ ràng khi xử lý thay đổi các hình ảnh trước khi chuyển đổi State.

## 3.6 Source code một số chức năng của nhân vật chính – Ember

### 3.6.1 Di chuyển

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

3 references
public class PlayerMoveState : PlayerGroundedState
{
    1 reference
    public PlayerMoveState(Player _player, PlayerStateMachine _stateMachine, string _animBoolName) : base(_player, _stateMachine, _animBoolName)
    {
    }

    14 references
    public override void Enter()
    {
        base.Enter();
        //
        AudioManager.instance.PlaySFX(39, null);
    }

    13 references
    public override void Exit()
    {
        base.Exit();
        //
        AudioManager.instance.StopSFX(39);
    }

    13 references
    public override void Update()
    {
        base.Update();

        player.SetVelocity(xInput * player.moveSpeed, rb.velocity.y);

        if (xInput == 0 || player.IsWallDetected())
            stateMachine.ChangeState(player.idleState);
    }
}
```

Hình 3. 18. Code di chuyển trái, phải của nhân vật chính

Người chơi có thể điều khiển nhân vật di chuyển thông qua các phím:

- Bấm A để di chuyển sang trái.
- Bấm D để di chuyển sang phải.

### 3.6.2 Lướt

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

3 references
public class PlayerDashState : PlayerState
{
    1 reference
    public PlayerDashState(Player _player, PlayerStateMachine _stateMachine, string _animBoolName) : base(_player, _stateMachine, _animBoolName)
    {
    }

    11 references
    public override void Enter()
    {
        base.Enter();
        AudioManager.instance.PlaySFX(41, null);
        stateTimer = player.dashDuration;
    }

    10 references
    public override void Exit()
    {
        base.Exit();
        player.SetVelocity(0, rb.velocity.y);
    }

    10 references
    public override void Update()
    {
        base.Update();

        if(!player.IsGroundDetected() && player.IsWallDetected()) //giup nhan vat vao trang thai bam tuong nhanh hon khi vua thuc hien dash vao tuong
            stateMachine.ChangeState(player.wallSlide);

        player.SetVelocity(player.dashSpeed * player.dashDir, 0);

        if(stateTimer < 0)
            stateMachine.ChangeState(player.idleState);
    }
}
```

Hình 3. 19. Code hành động lướt của nhân vật chính

Người chơi bấm Shift để lướt, điều này không chỉ giúp nhân vật di chuyển nhanh hơn mà còn có thể né được các đòn tấn công của quái vật trên đường.

### 3.6.3 Nhảy

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

3 references
public class PlayerJumpState : PlayerState
{
    1 reference
    public PlayerJumpState(Player _player, PlayerStateMachine _stateMachine, string _animBoolName) : base(_player, _stateMachine, _animBoolName)
    {
    }

    11 references
    public override void Enter()
    {
        base.Enter();
        player.jumpCounter++;
        rb.velocity = new Vector2(rb.velocity.x, player.jumpForce);
    }

    10 references
    public override void Exit()
    {
        base.Exit();
    }

    10 references
    public override void Update()
    {
        base.Update();

        if (rb.velocity.y < 0)
            stateMachine.ChangeState(player.airState);
    }
}
```

Hình 3. 20. Code hành động nhảy của nhân vật chính

Người chơi có thể thực hiện thao tác nhảy thông qua việc nhấn phím Space trên bàn phím, là một trong những chức năng cốt lõi trong trải nghiệm điều khiển nhân vật, cho phép nhân vật vượt qua các chướng ngại vật, hoặc tiếp cận những vị trí cao hơn trong màn chơi.

### 3.6.4 Tấn công

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

3 references
public class PlayerPrimaryAttackState : PlayerState
{
    private int comboCounter;
    private float lastTimeAttacked;
    private float comboWindow = 2;
    1 reference
    public PlayerPrimaryAttackState(Player _player, PlayerStateMachine _stateMachine, string _animBoolName) : base(_player, _stateMachine, _animBoolName)
    {
    }

    11 references
    public override void Enter()
    {
        base.Enter();
        xInput = 0; //de sua loi huong tan cong

        if (comboCounter > 2 || Time.time >= lastTimeAttacked + comboWindow)
            comboCounter = 0;

        player.anim.SetInteger("ComboCounter", comboCounter);

        float attackDir = player.facingDir;
        if(xInput != 0)
            attackDir = xInput;

        player.SetVelocity(player.attackMoment[comboCounter].x * attackDir, player.attackMoment[comboCounter].y);

        stateTimer = .1f;
    }

    10 references
    public override void Exit()
    {
        base.Exit();
        player.StartCoroutine("BusyFor", .15f);
        comboCounter++;
        lastTimeAttacked = Time.time;
    }
}
```

```
public override void Update()
{
    base.Update();

    if (stateTimer < 0)
        player.SetZeroVelocity();

    if(triggerCalled)
        stateMachine.ChangeState(player.idleState);
}
}
```

Hình 3. 21. Code hành động tấn công của nhân vật chính

Với thao tác đơn giản là nhấn chuột trái liên tục, nhân vật sẽ tung ra đòn đánh tạo thành chuỗi combo đẹp mắt nhằm tấn công quái vật hoặc phá hủy vật phẩm trên đường đi.

### 3.6.5 Trượt tường (Wall Slide)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

3 references
public class PlayerWallSlideState : PlayerState
{
    1 reference
    public PlayerWallSlideState(Player _player, PlayerStateMachine _stateMachine, string _animBoolName) : base(_player, _stateMachine, _animBoolName)
    {
    }

    11 references
    public override void Enter()
    {
        base.Enter();
    }

    10 references
    public override void Exit()
    {
        base.Exit();
    }

    10 references
    public override void Update()
    {
        base.Update();

        if(Input.GetKeyDown(KeyCode.Space))
        {
            stateMachine.ChangeState(player.wallJump);
            return;
        }

        if(xInput != 0 && player.facingDir != xInput)
            stateMachine.ChangeState(player.idleState);

        if(yInput < 0)
            rb.velocity = new Vector2(0, rb.velocity.y);
        else
            rb.velocity = new Vector2(0, rb.velocity.y * .7f);

        if (player.IsGroundDetected())
            stateMachine.ChangeState(player.idleState);
        //test fix bug
        if (!player.IsWallDetected())
        {
            stateMachine.ChangeState(player.airState);
        }
    }
}
```

Hình 3. 22. Code hành động trượt tường của nhân vật chính

Khi nhân vật tiếp xúc với bề mặt tường theo phương đứng nhân vật sẽ tự động bám tường và trượt chậm xuống thay vì rơi tự do. Điều này giúp người chơi kiểm soát tốc độ rơi, tạo điểm dừng tạm thời để định hướng nhảy hoặc thực hiện Wall Jump.

### 3.6.6 Nhảy tường (Wall Jump)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

3 references
public class PlayerWallJumpState : PlayerState
{
    1 reference
    public PlayerWallJumpState(Player _player, PlayerStateMachine _stateMachine, string _animBoolName) : base(_player, _stateMachine, _animBoolName)
    {
    }

    11 references
    public override void Enter()
    {
        base.Enter();
        stateTimer = 1f;
        player.SetVelocity(5 * -player.facingDir, player.jumpForce);
    }

    10 references
    public override void Exit()
    {
        base.Exit();
    }

    10 references
    public override void Update()
    {
        base.Update();
        if (stateTimer < 0)
            stateMachine.ChangeState(player.airState);

        if (player.IsGroundDetected())
            stateMachine.ChangeState(player.idleState);
    }
}
```

Hình 3. 23. Code hành động nhảy tường của nhân vật chính

Chức năng cho phép nhân vật bật nhảy khỏi tường khi đang bám/trượt, tạo một cú nhảy hướng ngược lại bề mặt tiếp xúc. Giúp tăng tính linh hoạt khi di chuyển trên bản đồ, vượt qua các chướng ngại gai dọc hoặc tiếp cận vị trí cao hơn.



## CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM

### 4.1 Kết quả thực nghiệm

Sau khi hoàn thiện phiên bản demo của game The Last Gleam, nhóm chúng em đã tiến hành kiểm thử trên nền tảng Windows PC để đánh giá các tính năng cốt lõi cũng như hiệu năng vận hành của game. Kết quả cho thấy game có khả năng chạy ổn định, không xuất hiện lỗi nghiêm trọng, các cơ chế chính như điều khiển nhân vật, hệ thống chiến đấu, boss fight, hiệu ứng kỹ năng, craft trang bị và trang bị lên nhân vật chạy ổn định và các UI đều hoạt động đúng theo như thiết kế.

Về mặt gameplay, game đã tái hiện được trải nghiệm đặc trưng của thể loại platformer 2D: Người chơi cần quan sát địa hình, điều khiển chính xác các hành động như nhảy, lướt và tấn công, đồng thời phải học cách đối phó với các loại quái vật khác nhau. Boss cuối Morveth được thiết kế có nhiều pha tấn công khác nhau, yêu cầu người chơi phải thích nghi từ từ sau nhiều lần thất bại để giành chiến thắng.

Phong cách đồ họa pixel art, kết hợp với âm thanh nền tối và hiệu ứng ánh sáng động, tạo nên không khí u tối và bí ẩn, làm nổi bật chủ đề của game.

### 4.2 So sánh với các game cùng thể loại

*Bảng 4. 1. Bảng so sánh với game cùng thể loại*

Tiêu chí	The Last Gleam	Hollow Knight	Dead Cells
Phong cách đồ họa	Pixel art, tối giản	Vẽ tay, chi tiết	Pixel art hiện đại
Cách kể chuyện	Ẩn dụ	Gợi mở, nhiều tầng	Gợi ý qua gameplay
Cơ chế game	Platformer, combat	Platformer, Metroidvania	Roguelike, combat
Boss	Có kịch bản tuyến tính	Nhiều, theo nhánh	Nhiều, ngẫu nhiên
Mức độ khó	Trung bình – cao	Khó, có độ sâu	Khó, phản xạ cao
Quy mô game	Nhỏ, tuyến tính	Lớn, thế giới mở	Trung, ngẫu nhiên
Cảm hứng chính	Cá nhân, fantasy	Gothic fantasy	Combat, gameplay

=> Qua bảng so sánh nhóm chúng em thấy, mặc dù The Last Gleam có quy mô nhỏ hơn và ít tính năng hơn so với các tựa game cùng thể loại, nhưng game vẫn thể hiện rõ nét cá tính riêng qua phong cách nghệ thuật pixel và hệ thống boss được đầu tư thiết kế. Game tập trung vào trải nghiệm tuyến tính, giàu cảm xúc và dễ tiếp cận.

## CHƯƠNG 5: KẾT LUẬN

Sau quá trình thực hiện, đề tài “Game 2D thể loại phiêu lưu – nhập vai hành động sử dụng Unity với đồ họa pixel” của nhóm chúng em đã hoàn thành đúng theo mục tiêu và phạm vi đặt ra ban đầu. Đây là sản phẩm đầu tay có sự kết hợp giữa kỹ năng lập trình, tư duy thiết kế gameplay và khả năng sáng tạo nghệ thuật thông qua phong cách pixel art. Đây là một con game phiêu lưu đi cảnh mang yếu tố hành động, chiến đấu với quái vật và boss, đồng thời mang chiều sâu về nội dung.

Sản phẩm đã hoàn thiện các tính năng cốt lõi: hệ thống điều khiển nhân vật, combat mượt mà và ổn định, AI đơn giản cho quái vật, boss được thiết kế theo từng giai đoạn (phase), âm thanh và hiệu ứng ánh sáng tạo không khí riêng biệt. Các địa hình, khu vực màn chơi được thiết kế với độ khó riêng biệt, có bố cục rõ ràng và tạo được thử thách cho người chơi. Nhân vật chính và boss Morveth đều có phong cách hình ảnh đặc trưng, truyền tải được thông điệp chủ đề xoay quanh “ánh sáng – bóng tối”.

Nhưng bên cạnh đó, chúng em nhận thấy rằng đề tài vẫn còn một số hạn chế về mặt nội dung và tính năng, chẳng hạn như chưa hỗ trợ chơi trên nhiều nền tảng, và nội dung cốt truyện tuy có chiều sâu nhưng vẫn còn ngắn gọn, cần mở rộng thêm để tạo sự gắn kết mạnh hơn với người chơi hơn.

### 5.1 Kết quả đạt được

- Hiểu được cấu trúc, cách thức hoạt động, cách thức phát triển một ứng dụng trên Unity.
- Hiểu biết nhiều hơn về ngôn ngữ C#.
- Nâng cao kiến thức lập trình về game.
- Chèn âm thanh vào game cũng như một số chức năng cơ bản khác mà một game thể loại phiêu lưu 2D platformer cần có.

### 5.2 Những điểm chưa đạt được

- Game còn sơ sài, chưa thực sự thú vị và cuốn hút.
- Các Script trong Unity tổ chức chưa thực sự rõ ràng.
- Sắp xếp quái vật trong màn chơi chưa thực sự logic.

### 5.3 Định hướng phát triển

- Mở rộng số lượng màn chơi, bổ sung thêm boss phụ, quái vật và môi trường đa dạng hơn.
- Mở rộng hệ thống nâng cấp kỹ năng và vật phẩm hỗ trợ.
- Tích hợp thêm cutscene ngắn hoặc đoạn phim pixel để câu chuyện trong game có chiều sâu hơn.
- Nâng cấp UI, hệ thống HUD và thêm phần tùy chỉnh cho người chơi như chọn độ khó, thiết lập phím để nâng cao trải nghiệm người chơi.
- Phát hành game trên các nền tảng như Android hoặc WebGL để tiếp cận người chơi rộng rãi hơn.
- Nếu có thời gian, trong tương lai nhóm chúng em sẽ thử phát triển thêm chế độ Multiplayer.

# GAME 2D THỂ LOẠI PHIÊU LƯU - NHẬP VAI HÀNH ĐỘNG SỬ DỤNG UNITY VỚI ĐỒ HỌA PIXEL



\The Last Gleam/

**Ngành: Công Nghệ Thông Tin**

**Lớp: 21DTHJA2**

**Nhóm: TMP**

**GV: Trịnh Đồng Thạch Trúc**

## NEED

Bối cảnh: Những năm gần đây, làn sóng game indie – đặc biệt là thể loại 2D platformer kết hợp pixel art đã trở nên phổ biến trên toàn cầu. Sự thành công vang dội của thể loại này đã truyền cảm hứng cho hàng ngàn studio nhỏ theo đuổi phong cách nghệ thuật retro và lối chơi đi cảnh cổ điển.

Vấn đề: Người chơi yêu thích thể loại 2D platformer đang cần trải nghiệm thêm những tựa game có cùng thể loại có chiều sâu nội dung, kết hợp giữa hành động, phiêu lưu và nhập vai. Họ cần một trải nghiệm vừa mang tính thử thách, vừa giàu cảm xúc, có cốt truyện lôi cuốn để thỏa mãn đam mê.

## COMPETITION



Điểm nổi bật riêng biệt của The Last Gleam:

- The Last Gleam đi theo hướng màn chơi tuyến tính, tập trung vào trình diễn câu chuyện ngắn gọn nhưng đủ ấn tượng trong từng khu vực địa hình.
- Thiết kế môi trường có chiều sâu, liên kết chặt chẽ giữa gameplay và cốt truyện.
- Pha trộn yếu tố nhập vai, nơi người chơi tự do phát triển nhân vật theo lối chơi của mình.
- Tái tạo cảm giác phiêu lưu cổ điển trong lớp áo pixel art, tối giản nhưng chính chu.



## APPROACH

The Last Gleam là game 2D platformer pixel art phong cách phiêu lưu hành động – nhập vai, đưa người chơi vào hành trình khám phá một thế giới u ám, bí ẩn đầy thử thách.

Nội dung và tính năng chính:

- Hệ thống bản đồ nhiều chia nhiều khu vực địa hình.
- Hệ thống quái vật đa dạng.
- Cơ chế chiến đấu thời gian thực: né đòn, tấn công tạo thành chuỗi combo đẹp mắt.
- Cốt truyện lôi cuốn.
- Hệ thống nâng cấp nhân vật đa hướng: kỹ năng chủ động, trang bị và chỉ số.
- Pixel art theo phong cách gothic fantasy, âm nhạc u ám.



## Giao diện craft vật phẩm



**S**

- Đồ họa pixel art đẹp mắt, mang phong cách hoài cổ.
- Gameplay kết hợp giữa hành động, phiêu lưu nhập vai, tạo chiều sâu và độ cuốn hút.

**W**

- Đội ngũ phát triển nhỏ, nguồn lực hạn chế ảnh hưởng tiến độ phát triển.
- Khả năng marketing ban đầu còn yếu, khó tiếp cận được số đông nếu không có chiến lược cụ thể.
- Chưa có hệ thống đa ngôn ngữ.

**O**

- Thị trường game indie tại Việt Nam đang phát triển mạnh.
- Người chơi ngày càng ưa chuộng các game 2D có chiều sâu.
- Nhu cầu trải nghiệm game gợi cảm xúc, nghệ thuật đang tăng cao trong cộng đồng yêu game.

**T**

- Cạnh tranh gay gắt từ các tựa game indie quốc tế.
- Khó gây chú ý nếu không có điểm khác biệt rõ ràng hoặc nội dung tiếp thị hiệu quả.
- Rủi ro bị "chết yếu" nếu không đạt đủ doanh thu hoặc thiếu vốn đầu tư.

## COST

- Chi phí phát triển game.
- Chi phí xây dựng hệ thống gameplay, bản đồ và cốt truyện.
- Chi phí duy trì, sửa lỗi và cập nhật nội dung sau phát hành.

## BENEFITS

- Tăng trải nghiệm giải trí kết hợp tư duy khám phá.
- Góp phần phát triển ngành game indie tại Việt Nam.
- Tạo nền tảng mở rộng IP: Truyện tranh, âm nhạc, dự án game tiếp theo.

## TÀI LIỆU THAM KHẢO

- [1]. Trịnh Đức Huy, Unity là gì? Unity có quan trọng với sự nghiệp của Game Developer không?, 29/3/2022, <https://itviec.com/blog/unity-la-gi/>
- [2]. ITNavi, Lập trình Unity là gì và những kiến thức tổng quan cần nắm vững, 14/10/2020, <https://itnavi.com.vn/blog/lap-trinh-unity>
- [3]. Thanh Nguyen, DATN – đồ án tốt nghiệp game, 2018, <https://www.studocu.vn/vn/document/truong-dai-hoc-bach-khoa-dai-hoc-quoc-gia-thanh-pho-ho-chi-minh/khoa-luan-tot-nghiep/dath-do-an-tot-nghiep-game/33548200>
- [4]. Chipa, Lập trình game với Unity như thế nào?, 10/3/2020, <https://codelearn.io/sharing/lap-trinh-game-voi-unity-nhu-the-nao-phan-1>
- [5]. Alex Dev, The Ultimate Guide to Creating an RPG Game in Unity 6, 2024, <https://www.udemy.com/course/2d-rpg-alexdev/?couponCode=KEEPLEARNING>
- [6]. Clembod, Warrior-Free Animation set V1.3, 5/2020, <https://clembod.itch.io/warrior-free-animation-set>
- [7]. Clembod, Bringer Of Death (free), 6/2020, <https://clembod.itch.io/bringer-of-death-free>
- [8]. LuizMelo, Monster Creatures Fantasy, 2020, <https://luizmelo.itch.io/monsters-creatures-fantasy>
- [9]. Thúy Diễm, C# là gì? Tất tần tật những thông tin về C# mà các lập trình viên mới nên biết, 2024, <https://fptshop.com.vn/tin-tuc/danh-gia/c-la-gi-175870>