

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

----------

**BÁO CÁO**

**THỰC TẬP CƠ SỞ**

***Giảng viên hướng dẫn:***

***Sinh viên thực hiện : Vũ Ngọc Hùng***

***Mã sinh viên: B22DCCN373***

***Lớp: D22CQCN01-B***

**Hà Nội, 2025**

# MỞ ĐẦU: Đặt vấn đề

**1. Lý do chọn đề tài/Tính cấp thiết của đề tài:**

Game nhập vai (RPG) là một trong những thể loại game lâu đời và được yêu thích nhất, với khả năng kể chuyện sâu sắc và cho phép người chơi hóa thân vào nhân vật. Trong khuôn khổ thực tập cơ sở, em mong muốn vận dụng những kiến thức đã học tại Học viện Công nghệ Bưu chính Viễn thông, đặc biệt là về lập trình hướng đối tượng và phát triển ứng dụng, để xây dựng một tựa game RPG đơn giản. Đây không chỉ là cơ hội để em củng cố kiến thức mà còn là bước đầu để tìm hiểu sâu hơn về ngành công nghiệp game.

**2. Mục tiêu của đợt thực tập và báo cáo:**

* + Mục tiêu của bài làm : Tìm hiểu quy trình phát triển game cơ bản, làm quen với Unity Engine, phát triển các kỹ năng lập trình C#, thiết kế và triển khai các tính năng cốt lõi của game RPG.
  + Mục tiêu của báo cáo: Ghi lại quá trình nghiên cứu, thiết kế và phát triển game. Phân tích các công nghệ và kỹ thuật đã sử dụng. Đánh giá kết quả đạt được và rút kinh nghiệm.

**3. Đối tượng và phạm vi nghiên cứu:**

* + Đối tượng nghiên cứu: Các kỹ thuật, công nghệ và phương pháp phát triển game RPG trên nền tảng Unity Engine.
  + Phạm vi nghiên cứu/phát triển game:
    - Xây dựng một game RPG 2D đơn giản cho nền tảng PC.
    - Các tính năng chính dự kiến:
      * Hệ thống di chuyển và điều khiển nhân vật.
      * Hệ thống chiến đấu cơ bản (đánh thường, kỹ năng đơn giản nếu có).
      * Hệ thống quản lý chỉ số nhân vật (HP, MP, ATK, DEF, EXP, Level).
      * Hệ thống vật phẩm và túi đồ đơn giản.
      * Lưu/tải game.

**4. Phương pháp nghiên cứu/thực hiện:**

* + Nghiên cứu tài liệu: Tìm hiểu về thiết kế game RPG, lập trình Unity C#, các kiến trúc phần mềm trong game.
  + Phân tích: Phân tích các game RPG tương tự để học hỏi.
  + Thiết kế: Thiết kế gameplay, giao diện, cơ sở dữ liệu (ScriptableObjects).
  + Thực nghiệm: Lập trình, triển khai các tính năng trên Unity.
  + Kiểm thử và đánh giá: Chơi thử, gỡ lỗi và cải tiến game.

**5. Cấu trúc của báo cáo:**

* + Nêu ngắn gọn các chương:
    - Chương 1: Tổng quan về game RPG và Unity Engine.
    - Chương 2: Nghiên cứu các phương pháp, kỹ thuật và công nghệ phát triển game RPG trên Unity.
    - Chương 3: Phân tích, thiết kế và triển khai hệ thống game RPG đơn giản
    - Kết luận và kiến nghị.

# LỜI CẢM ƠN

Để hoàn thành được báo cáo thực tập tốt nghiệp này thì em xin cảm ơn tới phía ban lãnh đạo --------------- đã tạo điều kiện cho em được thực tập tại đây. Ngoài ra, em cảm ơn ------- đã luôn nhiệt tình chỉ dẫn, giảng dạy cho em những kiến thức về ------ để em có thể hoàn thiện hơn.

Em xin gửi lời cảm ơn tới ban lãnh đạo Học viện, Khoa đào tạo đã thiết lập khung chương trình có môn học “Thực tập” để em có thể vận dụng những kiến thức học được trên giảng đường áp dụng vào công việc và được tham quan, tìm hiểu về môi trường doanh nghiệp.

Em xin gửi lời cảm ơn sâu sắc và chân thành tới giảng viên hướng dẫn -----. Nhờ---- luôn dìu dắt, giúp đỡ và chỉ bảo tận tình cho em để hoàn thành tốt Thực tập.

Em xin chân thành cảm ơn!

Hà Nội, ngày ---- tháng --- năm ---

**MỤC LỤC**

# DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Viết đầy đủ** | **Giải thích nghĩa** |

# DANH MỤC HÌNH ẢNH

# CHƯƠNG 1: Tổng quan về bài toán / lĩnh vực nghiên cứu

## 1.1. Giới thiệu về game RPG

### + Định nghĩa game RPG:

Game RPG (Role-Playing Game) là một thể loại game trong đó người chơi sẽ nhập vai vào một nhân vật hoặc nhóm nhân vật và tham gia vào một câu chuyện, thường là một cuộc phiêu lưu hoặc chiến đấu. Trong các game RPG, người chơi thường được phép phát triển và tùy chỉnh nhân vật của mình, tham gia vào các nhiệm vụ, giải đố, và giao tiếp với các nhân vật khác trong một thế giới giả tưởng. Cốt truyện, các quyết định và sự tiến triển của nhân vật là trọng tâm trong thể loại này.

### Các loại game RPG (turn-based, action RPG, MMO, v.v***.)***

#### +Turn-based RPG (RPG theo lượt):

**Đặc điểm**: Trong các game này, các nhân vật hành động theo lượt, nghĩa là mỗi bên (người chơi hoặc đối thủ) sẽ có một khoảng thời gian để thực hiện hành động trước khi lượt tiếp theo bắt đầu.

**Ví dụ nổi bật**: *Final Fantasy*, *Dragon Quest*, *Pokémon*

**Lý do người chơi yêu thích**: Các trận đấu theo lượt giúp người chơi có thời gian suy nghĩ và lên kế hoạch chiến đấu.

#### + Action RPG (RPG hành động)

**Đặc điểm**: Thể loại này kết hợp yếu tố chiến đấu hành động trực tiếp với các yếu tố RPG. Người chơi có thể điều khiển nhân vật để tham gia vào các trận đấu thời gian thực.

**Ví dụ nổi bật**: *The Elder Scrolls V: Skyrim*, *Dark Souls*, *The Witcher 3: Wild Hunt*

**Lý do người chơi yêu thích**: Game này đem đến cảm giác chiến đấu mạnh mẽ và tốc độ, giúp người chơi tham gia trực tiếp vào các trận chiến.

#### + MMORPG (Massively Multiplayer Online Role-Playing Game)

**Đặc điểm**: MMORPG là một thể loại game RPG trực tuyến, nơi hàng nghìn người chơi có thể tương tác với nhau trong cùng một thế giới ảo. Các game MMORPG cho phép người chơi tạo nhân vật, làm nhiệm vụ, chiến đấu và hợp tác với người chơi khác trong một thế giới rộng lớn.

**Ví dụ nổi bật**: *World of Warcraft*, *Final Fantasy XIV*, *Guild Wars2*

**Lý do người chơi yêu thích**: MMORPG mang đến một thế giới mở rộng lớn và đầy cơ hội để khám phá, kết bạn và hợp tác với hàng ngàn người chơi khác.

#### + Action-Adventure RPG (RPG phiêu lưu hành động)

**Đặc điểm**: Đây là sự kết hợp giữa game hành động, phiêu lưu và RPG. Người chơi sẽ tham gia vào các cuộc phiêu lưu, khám phá thế giới mở và giải quyết các câu đố, đồng thời phát triển nhân vật qua việc nâng cấp kỹ năng, trang bị.

**Ví dụ nổi bật**: *The Legend of Zelda: Breath of the Wild*, *Horizon Zero Dawn*

**Lý do người chơi yêu thích**: Thế giới mở và tự do khám phá cùng với yếu tố hành động hấp dẫn.

## 1.2. Lý do chọn thể loại RPG

Lựa chọn thể loại **RPG** (Game nhập vai) làm đối tượng nghiên cứu hoặc phát triển có nhiều lý do hấp dẫn và hợp lý, dưới đây là những lý do nổi bật khiến thể loại RPG trở thành một lựa chọn ưa thích :

### Tính Linh Hoạt và Tự Do Cao

**Lý do:** RPG mang lại cho người chơi sự tự do tuyệt vời trong việc tùy chỉnh và phát triển nhân vật. Người chơi có thể lựa chọn kỹ năng, vũ khí, hoặc cách phát triển cốt truyện, tạo nên một trải nghiệm cá nhân hóa sâu sắc. Điều này tạo ra sự kết nối mạnh mẽ giữa người chơi và nhân vật trong game, khiến họ cảm thấy họ thực sự "là" nhân vật đó trong một thế giới ảo.

**Ví dụ:** Trong *The Elder Scrolls V: Skyrim*, người chơi có thể chọn lựa từ một loạt các nhánh kỹ năng khác nhau như chiến đấu, phép thuật hay đánh cắp, từ đó ảnh hưởng đến cách chơi và trải nghiệm game của họ.

### Khả Năng Kể Chuyện Mạnh Mẽ

**Lý do:** Cốt truyện là một yếu tố quan trọng trong RPG, mang đến cho game một chiều sâu và lý do để người chơi gắn bó lâu dài. Các game RPG thường có các câu chuyện phức tạp, đầy sự kiện và nhân vật đa chiều, khiến người chơi muốn khám phá tiếp câu chuyện cho đến khi hoàn thành.

**Ví dụ:** Các game như *Final Fantasy* hay *The Witcher 3: Wild Hunt* không chỉ là những trò chơi giải trí, mà còn là những câu chuyện sử thi, nơi người chơi bị cuốn vào một thế giới giả tưởng và một cốt truyện đậm chất điện ảnh.

### Tăng Cường Kỹ Năng Chiến Thuật và Quyết Định

**Lý do:** Các trận chiến trong RPG yêu cầu người chơi phải suy nghĩ chiến thuật và quyết định khôn ngoan. Đây là nơi mà người chơi có thể áp dụng các kỹ năng lập kế hoạch và tư duy chiến lược. Tùy thuộc vào cách phát triển nhân vật và lựa chọn của người chơi, kết quả trận đấu có thể thay đổi, đem lại sự thử thách thú vị và hài lòng.

**Ví dụ:** Trong các trò chơi *Turn-based RPG* như *Dragon Quest*, người chơi không chỉ cần sức mạnh mà còn cần suy nghĩ chiến thuật để vượt qua các thử thách và kẻ thù mạnh.

### Cảm Giác Tiến Bộ và Phát Triển Nhân Vật

**Lý do:** Một trong những yếu tố chính trong RPG là việc người chơi có thể tiến bộ qua các cấp độ, phát triển nhân vật, mở khóa kỹ năng mới và nhận phần thưởng. Điều này tạo ra một cảm giác thỏa mãn mạnh mẽ khi người chơi chứng kiến sự thay đổi và trưởng thành của nhân vật mà mình đang điều khiển.

**Ví dụ:** Trong *Diablo* hay *Path of Exile*, cảm giác "grinding" (cày cuốc) và "leveling up" (nâng cấp cấp độ) tạo nên sự thú vị và cảm giác thành tựu.

## 1.3. Lý do chọn Unity Engine

### 1.3.1. Lợi ích của Unity trong phát triển game :

Unity Engine đã trở thành một trong những công cụ phát triển game phổ biến nhất trên thế giới, đặc biệt là đối với các nhà phát triển game Indie và các dự án nhỏ đến trung bình. Dưới đây là một số lý do vì sao Unity được lựa chọn để phát triển game, đặc biệt là game RPG :

#### Đa nền tảng (Cross-Platform)

* **Lý do:** Unity hỗ trợ phát triển game trên rất nhiều nền tảng khác nhau, bao gồm PC, Mac, Linux, iOS, Android, PlayStation, Xbox, Nintendo Switch, và cả VR/AR. Điều này giúp các nhà phát triển game có thể dễ dàng xuất bản sản phẩm của mình lên nhiều thiết bị khác nhau mà không cần phải viết lại mã nguồn cho từng nền tảng.
* **Ví dụ:** Ta có thể phát triển một game RPG cho PC, sau đó dễ dàng xuất bản game đó lên điện thoại di động hay thậm chí các nền tảng thực tế ảo như Oculus Rift.

#### Giao diện trực quan và dễ sử dụng

* **Lý do:** Unity có giao diện người dùng (UI) thân thiện và dễ tiếp cận, đặc biệt đối với những người mới bắt đầu. Các tính năng của Unity như **Scene View**, **Hierarchy**, **Inspector** giúp các nhà phát triển dễ dàng tổ chức và quản lý các đối tượng trong game. Điều này giúp giảm thời gian học hỏi và tập trung vào việc sáng tạo game hơn là phải lo lắng về các công cụ phát triển.
* **Ví dụ:** Với Unity, Ta có thể nhanh chóng kéo và thả các đối tượng vào Scene, kiểm tra hiệu ứng vật lý hay thay đổi các thuộc tính của nhân vật mà không cần phải biết quá nhiều về lập trình.

#### Hỗ trợ mạnh mẽ về đồ họa 2D và 3D

* **Lý do:** Unity hỗ trợ cả đồ họa 2D và 3D, cho phép các nhà phát triển dễ dàng tạo ra các game RPG với môi trường 3D rộng lớn hoặc các trò chơi 2D đơn giản. Các tính năng đồ họa mạnh mẽ, như hỗ trợ ánh sáng, vật liệu, và hiệu ứng đặc biệt, giúp tăng cường chất lượng hình ảnh cho game mà không cần phải sử dụng công cụ đồ họa bên ngoài.
* **Ví dụ:** Unity giúp chúng ta tạo các nhân vật 3D, cảnh vật và ánh sáng phức tạp cho một game RPG, đồng thời cũng hỗ trợ dễ dàng các yếu tố 2D như bản đồ, giao diện người chơi (UI) hay các trò chơi RPG theo phong cách cổ điển.

#### Cộng đồng và tài nguyên hỗ trợ mạnh mẽ

* **Lý do:** Một trong những điểm mạnh lớn nhất của Unity là cộng đồng người dùng rộng lớn và tài nguyên hỗ trợ phong phú. Unity Asset Store là nơi bạn có thể tìm kiếm và mua các tài nguyên có sẵn như mô hình 3D, âm thanh, hệ thống AI, giao diện người dùng (UI) và nhiều hơn nữa. Điều này giúp các nhà phát triển tiết kiệm thời gian và công sức, đồng thời giảm bớt chi phí phát triển game.
* **Ví dụ:** Thay vì tự thiết kế một bộ vũ khí 3D từ đầu, chúng ta có thể mua các bộ mô hình vũ khí trong Unity Asset Store và dễ dàng tích hợp vào game của mình.

#### Hỗ trợ phát triển AI và hệ thống game phức tạp

* **Lý do:** Unity cung cấp nhiều công cụ và thư viện hỗ trợ việc phát triển các hệ thống phức tạp như AI, điều khiển nhân vật, và hệ thống chiến đấu. Bạn có thể dễ dàng tích hợp các thuật toán AI vào game RPG, giúp NPCs trở nên thông minh hơn và nâng cao chất lượng gameplay.
* **Ví dụ:** Unity có **NavMesh** cho AI di chuyển và **Animator** cho các chuyển động của nhân vật, giúp tạo ra một hệ thống chiến đấu hoặc hệ thống nhiệm vụ phong phú và tương tác.

#### Công cụ lập trình mạnh mẽ với C#

* **Lý do:** Unity sử dụng ngôn ngữ lập trình C#, một trong những ngôn ngữ phổ biến và dễ học trong ngành phát triển game. C# giúp việc lập trình trở nên dễ dàng hơn, đồng thời cung cấp một hệ thống mạnh mẽ để quản lý mã nguồn và các tương tác trong game.
* **Ví dụ:** Trong Unity, ta có thể lập trình các hành động như di chuyển nhân vật, tương tác với vật thể, tạo ra các kỹ năng đặc biệt cho nhân vật, và phát triển các tính năng phức tạp trong game RPG.

### 1.3.2. Unity và cộng đồng hỗ trợ game phát triển

#### Cộng đồng người dùng đông đảo

* + **Lý do:** Unity có một cộng đồng phát triển game cực kỳ lớn và năng động trên toàn cầu. Các nhà phát triển, từ những người mới bắt đầu đến các chuyên gia, đều tham gia vào các diễn đàn, nhóm trên mạng xã hội, và sự kiện để chia sẻ kinh nghiệm, giải đáp thắc mắc và cung cấp các tài nguyên hỗ trợ.
  + **Ví dụ:** Các diễn đàn như *Unity Forum* và các nhóm trên *Reddit*, *Discord* đều là nơi tuyệt vời để bạn hỏi đáp và trao đổi với các lập trình viên khác.

#### Tài liệu học tập và khóa học miễn phí

* + **Lý do:** Unity cung cấp rất nhiều tài liệu học tập, khóa học và video hướng dẫn miễn phí cho người mới bắt đầu, cũng như các chuyên gia muốn nâng cao kỹ năng. Các khóa học này bao gồm các chủ đề từ cơ bản đến nâng cao như phát triển game 2D, game 3D, AR/VR, và lập trình AI.
  + **Ví dụ:** Unity Learn là nền tảng học tập miễn phí của Unity cung cấp các khóa học và bài học trực tuyến giúp người học từ mọi trình độ có thể nắm vững cách sử dụng Unity trong phát triển game.

#### Hỗ trợ từ Unity Asset Store

* + **Lý do:** Asset Store của Unity cung cấp hàng nghìn tài nguyên có sẵn, từ mô hình 3D, âm thanh, mã nguồn, cho đến các công cụ phát triển game chuyên nghiệp. Điều này giúp các nhà phát triển tiết kiệm thời gian trong việc tạo ra các tài nguyên cần thiết cho game, đồng thời có thể áp dụng chúng vào dự án của mình một cách nhanh chóng.
  + **Ví dụ:** Ta có thể mua các gói mô hình nhân vật, môi trường, hệ thống chiến đấu, hoặc AI để tích hợp vào game RPG của mình mà không phải tự phát triển từ đầu.

# CHƯƠNG 2: Nghiên cứu phương pháp tiếp cận và kỹ thuật/công nghệ để giải quyết bài toán trong đề tài

## 2.1. Phương pháp phát triển game trong Unity

### Phương pháp phát triển game trong Unity (Sơ Lược)

#### Lập kế hoạch và thiết kế:

Trước tiên, cần xác định ý tưởng, cốt truyện và thiết kế game (bố cục, giao diện, nhân vật, cơ chế chơi). Việc lập kế hoạch chi tiết giúp tạo ra một lộ trình rõ ràng cho quá trình phát triển.

#### Xây dựng môi trường và các scene:

Sử dụng **Scene Editor** của Unity để tạo ra các cảnh (scene) tương ứng với các cấp độ, màn chơi hoặc màn hình giao diện trong game.

#### Tạo đối tượng và cấu hình bằng GameObject:

Mọi thứ trong Unity đều bắt đầu từ các **GameObject**. Các GameObject này sẽ được bổ sung các thành phần (components) để hiện thực hóa chức năng và giao diện mong muốn.

#### Lập trình và logic game:

Viết script bằng ngôn ngữ C# để điều khiển hành vi của các GameObject, xử lý tương tác người chơi, va chạm và các sự kiện trong game.

#### Kiểm thử và gỡ lỗi:

Sử dụng **Console** và các công cụ debug tích hợp của Unity để theo dõi lỗi, kiểm thử tính năng và tối ưu hóa hiệu năng.

#### Tối ưu hóa và xuất bản:

Sau khi hoàn thiện, game được tối ưu hóa cho từng nền tảng (PC, mobile, console, …) và xuất bản thông qua các cửa hàng ứng dụng hoặc nền tảng phân phối.

## 2.2. Các kỹ thuật phát triển cốt lõi trong game RPG đơn giản.

Khi phát triển Game RPG thì hai vấn đề về cấu trúc map và quản lí nhân vật là hai vấn đề cần giải quyết để định hình ra được tựa game RPG mà chúng ta muốn làm .

### 2.2.1. Cấu trúc thế giới game (Map, khu vực, NPCs)

#### Map (Bản đồ):

**Thiết kế địa hình:** Em đã ứng dụng Tilemap 2D trong unity trong việc tạo map và đã tạo ra các map như :

**+**Map khởi đầu (làng tân thủ ) ,

**+**Map nghĩa địa (nơi quái vật trú ngụ ) .

**Hệ thống kết nối:** Để kết nối các map lại với nhau em có sử dụng các cổng dịch chuyển được bố trị ở các vị trí cuối map cho người chơi di chuyển và khi người chơi di chuyển đến thì em sẽ chuyển người chơi tới một map khác (Scene khác ).

#### Khu vực:

**Phân chia vùng:** Ở các map em tạo bên trên em phân chia ra thành 2 vùng :

**+ Vùng 1 :** Đây là vùng an toàn : ở vùng này người chơi sẽ được tự do đi lại , mua sắm đồ từ các NPC và tập luyện để xem sức mạnh của nhân vật của họ đến đâu rồi . Các map là vùng an toàn như : Map khởi đầu .

**+ Vùng 2 :** Đây là vùng không an toàn đây là vùng mà em bố trí quái vật để đuỏi theo và tấn công người chơi khi người chơi đi vào vùng phát hiện và tấn công của quái vật.Và cũng ở vùng này khi người chơi giết quái vật thì sẽ nhận được kinh nghiệm để tăng cấp cũng như vật phẩm để từ đố gia tăng sức mạnh của nhân vật.

#### NPCs (Nhân vật không phải người chơi):

**Vai trò cốt lõi:** NPCs cung cấp thông tin , cũng như là có các tư vấn về các khu vực với người chơi đồng thười một số NPC sẽ bán và thu mua vật phẩm.

**Tương tác và hành vi:** Phát triển hệ thống giao tiếp với NPCs thông qua các kịch bản (script) và trí tuệ nhân tạo (AI) cơ bản để tạo nên sự sống động cho thế giới game.

### 2.2.2. Quản lý nhân vật

#### Chỉ số nhân vật

***Thuộc tính cơ bản:***

Nhân vật trong game được em định nghĩa và sử dụng các chỉ số như :Sức mạnh, Tốc độ , Mana, Sức chịu đựng .Những chỉ số này ảnh hưởng trực tiếp đến khả năng chiến đấu, phòng thủ và các hành động khác của nhân vật.

***Cân bằng trò chơi:***

Để tránh sức mạnh của người chơi tăng cường quá nhiều dẫn đến giảm độ khó của trò chơi thì mỗi khi lên cấp em sẽ cho người chơi nhận được một thuộc tính tự do dể tự do phân phối vào các thuộc tính nhân vật mà họ muốn và điểm thuộc tính tự do nhận được tăng theo cấp số cộng mỗi cấp người chơi sẽ nhận thêm một điểm và tối đa là 10 điểm mỗi cấp.

#### Kỹ năng

***Hệ thống cây kỹ năng:***

Một game RPG hấp dẫn thường có hệ thống kỹ năng phong phú, cho phép người chơi mở khóa và nâng cấp các khả năng đặc biệt. Cây kỹ năng (skill tree) giúp người chơi tùy chỉnh theo phong cách chiến đấu của riêng mình, từ chiến binh mạnh mẽ, pháp sư tài ba cho đến những nhân vật hỗ trợ linh hoạt.

Ứng dụng vào game : Em đã tiến hành xây dựng một hệ thống để triển khai Skill của nhân vật và em có chia Skill của nhân vật thành các loại sau : BasicSkill(đây là đòn đành thường ) , MagicSkill , HealSkill, AOESkill

***Phát triển và tùy biến:***

Các kỹ năng có thể được cải tiến qua từng cấp độ, giúp nhân vật ngày càng mạnh mẽ hơn. Sự đa dạng của các kỹ năng không chỉ tạo nên nhiều chiến thuật khác nhau mà còn giúp người chơi cảm nhận được quá trình tiến hóa và cá tính độc đáo của nhân vật.

#### Cấp độ (Level)

***Hệ thống tích lũy kinh nghiệm:***

Em triển khai hệ thống cấp độ và tích lũy kinh nghiệm cho người chơi thong qua việc người chơi đánh quái vật . và khi lên cấp thì số kinh nghiệm cần để lên cấp tiếp theo sẽ tăng theo cấp số nhân với công bội là 1.2.

**Tiến trình phát triển nhân vật:**  
 Cấp độ không chỉ đơn thuần là con số mà còn biểu thị quá trình trưởng thành của nhân vật. Mỗi lần lên cấp có thể đi kèm với việc cải thiện ngoại hình, trang bị mới, hay thậm chí thay đổi cốt truyện cá nhân, giúp người chơi cảm thấy nhân vật mình điều khiển luôn tiến bộ và phát triển theo thời gian.

### 2.2.3. Hệ thống chiến đấu (Combat System)

* + Phát hiện va chạm và tính sát thương:
    - Trong game để phát hiện kẻ địch em dùng Raycast để phát hiện kẻ địch và lấy đối tượng kẻ địch sau đó tiến hành trừ máu sau khi tấn công.
    - Công thức để tính Sát thương nhận vào của kẻ địch hoặc của người chơi em dùng là Sát thương nhận vào – Phòng thủ của bản thân = Sát thương cuối cùng.
  + Trạng thái nhân vật/quái vật (FSM):
    - Em sử dụng State Machine (FSM) để quản lý các trạng thái như Idle, Move, Attack, Die cho nhân vật và quái vật.
  + Hiệu ứng và Animation:
    - Các animation (đánh, bị đánh, chết) và hiệu ứng (máu, skill) được em tích vào vào game thông qua các State của nhân vật (em đã thiết lập bằng FSM) và thông qua các Event được phát ra .

### 2.2.4. Hệ thống vật phẩm và túi đồ (Inventory System)

* + Em đã thiết lập hệ thống IS nhằm chứa đồ cho nhân vật và em có phân loại các vật phẩm trong game như sau :
    - Vật phẩm thuộc loại trang bị : Vũ khí , Áo giáp , Giày , Mũ
    - Vật phẩm thuộc loại phi trang bị (có thể sử dụng ) : Bình hồi máu , bình hồi sức bền , bình tăng tốc độ , bình tăng lượng Exp nhận vào .
  + Dữ liệu vật phẩm được lưu trữ ở đâu (ScriptableObject)? Cấu trúc dữ liệu của một vật phẩm gồm những gì (tên, mô tả, icon, loại, chỉ số cộng thêm, giá bán)?
  + Giao diện túi đồ được thiết kế như thế nào? Người chơi tương tác với túi đồ ra sao (kéo thả, click để dùng)?

## **2.3. Các công nghệ sử dụng trong Unity**

### 2.3.1. Ngôn ngữ lập trình C#

C# là ngôn ngữ chính dùng trong Unity để viết logic game, bao gồm:

* Xử lý chuyển động, chiến đấu, tương tác của nhân vật.
* Tạo hệ thống nhiệm vụ, kỹ năng, trạng thái nhân vật.
* Điều khiển UI, âm thanh, lưu và tải dữ liệu.

### 2.3.2. Hệ thống vật lý (Physics Engine)

### Em sử dụng Physics 2D để tiến hành xây dựng vật lý cho game đồng thời em có sử dụng các Colider 2D như BoxColider2D , MeshColider2D , ... để tạo các va chạm cho các đối tượng trong game như quái vật , người chơi , vật phẩm rơi trên mặt đất.

### 2.3.3. Giao diện người dùng (UI System)

Em đã ứng dụng hệ thống UI của Unity vào việc xây dựng:

* Thanh máu (HP), độ bền (Sta) , kinh nghiệm (EXP).
* Menu nhân vật, cửa hàng ,..
* Hệ thống túi đồ (Inventory)
* Các popup tương tác như hội thoại, thông báo,...

Dùng các thành phần như Canvas, Text, Button, Image, Slider, Layout Group.

A screenshot of a video game

AI-generated content may be incorrect.

### 2.3.4. Hệ thống hoạt ảnh (Animation System)

Em sử dụng hệ thống hoạt ảnh được unity hỗ trợ như :

* Animator Controller: để điều khiển các hoạt ảnh như chết , chạy , nhảy , tấn công ,.. của cả người chơi và quái vật.

Ngoài ra em cũng sử dụng thêm các công cụ sau của Unity để tối ưu hóa hoạt ảnh.

* Blend Tree: Trộn nhiều trạng thái chuyển động mượt mà.
* Sprite Animation (2D) hoặc Mecanim (3D).

A screenshot of a video game

AI-generated content may be incorrect.

### 2.3.5. Tilemap System (cho RPG 2D)

Unity có một công cụ rất mạch giúp em thiết kế và triển khai map đó là công cụ TileMap 2D :

* Em đã ứng dụng nó vào việc vẽ map khời đầu và các map khác của game
* A screenshot of a video game

  AI-generated content may be incorrect.

### 2.3.6. ScriptableObject

Là loại dữ liệu đặc biệt mà em dùng để:

* Lưu trữ thông tin vật phẩm, kỹ năng, kẻ địch, NPC,...trong game của em
* Ngoài ra em còn ứng dụng SC vào việc tạo ra database nội bộ cho game mà không cần database ngoài.

### 2.3.7. Event System (Hệ thống sự kiện)

Em có sử dụng song song hệ thống sự kiện như :C# event, delegate hoặc UnityEvent cùng với State Machine Pattern vào trong việc thực hiện các hành động của các đối tượng khi chuyển sang các trạng thái khác nhau ví dụ như :

+ Khi nhân vật lên cấp em sẽ phát ra sự kiện nhân vật lên cấp

+ Khi nhân vật công thuộc tính thành công em cũng phát ra

sự kiện công thuộc tính thành công.

### 2.3.8. Hệ thống âm thanh (Audio System)

Unity có hỗ trợ:

* Phát nhạc nền (BGM), hiệu ứng âm thanh (SFX).
* Điều khiển âm lượng, fade in/out, lặp lại,...
* AudioMixer để nhóm và kiểm soát nhiều nguồn âm thanh.

Do đó em có ứng dụng vào việc phát ra âm thanh khi nhân vật chết , chạy , tấn công , nhận sát thương ,..

# CHƯƠNG 3: Phát triển hệ thống theo phương pháp đưa ra ở chương 2

## 3.1 Xác định yêu cầu

* **Yêu cầu chức năng (Functional Requirements):**
  + FR1: Người chơi có thể điều khiển nhân vật di chuyển trên bản đồ bằng [phím WASD].
  + FR2: Nhân vật có thể tấn công kẻ địch bằng [click chuột/nút bấm].
  + FR3: Kẻ địch có thể di chuyển và tấn công lại người chơi.
  + FR4: Nhân vật và kẻ địch có chỉ số HP, khi HP về 0 sẽ chết.
  + FR5: Người chơi có thể mở túi đồ để xem và sử dụng vật phẩm (ví dụ: bình máu , bình tốc độ , bình hồi thể lực).
  + FR6: Game có thể được lưu lại và tải lại.
  + FR7: Người chơi có thể tăng cấp bằng cách đánh quái vật và tự tăng điểm thuộc tính tự do.
  + FR8: Người chơi có thể mua đồ thông qua NPC.
* **Yêu cầu phi chức năng (Non-Functional Requirements):**
  + NFR1: Game có thể chạy ổn định trên máy tính ở mức trên 50 fps .
  + NFR2: Giao diện người dùng trực quan, dễ hiểu.
  + NFR3: Âm thanh, hình ảnh phù hợp với bối cảnh game.

## 3.2 Thiết kế tương tác cho game RPG đơn giản.

Mô tả cách người chơi tương tác với hệ thống: Di chuyển nhân vật, chiến đấu, tương tác với NPCs.

### 3.2.1. Sơ đồ luồng người dùng .

**Luồng bắt đầu :**

A diagram of a game

AI-generated content may be incorrect.

**Luồng chiến đấu :**

A diagram of a flowchart

AI-generated content may be incorrect.

### 3.2.2. Thiết kế giao diện cho game RPG đơn giản (UI Design)

**Giao diện khởi đầu :**

A screenshot of a video game

AI-generated content may be incorrect.

**Giao diện trong game :**

A video game with a person holding a sword

AI-generated content may be incorrect.

**Giao diện trong kho đồ:**

A screenshot of a computer

AI-generated content may be incorrect.

## 3.3 Mô hình tổng quát hệ thống

### Mô hình tổng quan hệ thống của em :



## 3.4 Mô hình phát triển phần mềm và phương pháp xây dựng:

Em sử dụng 2 phương pháp xây dựng phần mềm chính là :

**1. Lập Trình Hướng Đối Tượng (Object-Oriented Programming)**

**Phổ biến nhất** trong ngành game AAA

**Ứng dụng**: Unity (C#), Unreal Engine (C++)

**Studio áp dụng**: Naughty Dog, Blizzard, Rockstar Games

**Ưu điểm**: Dễ mở rộng, tái sử dụng code cao, quản lý trạng thái game tốt

**2. Lập Trình Hướng Thành Phần (Component-Based Programming)**

**Phương pháp chính** trong các engine hiện đại

**Ứng dụng**: Unity's GameObject-Component system, Unreal Engine's Actor-Component system

**Studio áp dụng**: Epic Games, CD Projekt RED, Ubisoft

**Ưu điểm**: Linh hoạt trong việc tạo game object, dễ mở rộng tính năng

Ngoài ra em sử dụng mô hình phát triển phần mềm là Agile & Scrum vì phương pháp này thường được sử dụng rộng rãi trong ngành game cùng với đó nó có những đặc điểm nổi trội như:

**Phổ biến nhất** trong ngành game hiện nay

**Đặc điểm**: Sprint 2-4 tuần, Daily Standup, Product Backlog, Sprint Review

**Studio áp dụng**: EA, Ubisoft, Bethesda, gần như mọi studio lớn

**Ưu điểm**: Thích ứng nhanh với thay đổi, liên tục cải tiến, demo sớm

**Thực tế áp dụng**: Thường được điều chỉnh cho phù hợp với đặc thù game

## 3.5 Kiến trúc phần mềm

Khi làm một tựa game RPG trên Unity, các studio thường “vá” nhiều kiểu kiến trúc cùng lúc để tận dụng điểm mạnh của từng phần, đồng thời đảm bảo tính mở rộng, bảo trì và hiệu năng.

*Do đó em sử dụng những kiến trúc phần mềm sau :*

+ Component-Based Architecture (Kiến trúc dựa trên Component) : đây là kiến trúc của unity tự nhiên và là cốt lõi của Unity. Với kiểu kiến trúc như vậy thì em đã áp dụng vào game như sau :

**Ví dụ :** "Nhân vật Player là một GameObject, được gắn các component như PlayerMovement (xử lý di chuyển), PlayerCombat (xử lý chiến đấu), PlayerStats (quản lý chỉ số), Inventory (quản lý túi đồ). Mỗi component độc lập và có thể tái sử dụng."

A screenshot of a computer program

AI-generated content may be incorrect.

+  **Service Locator Pattern (Hệ thống quản lý) để** Quản lý các hệ thống toàn cục của game (GameManager, QuestManager, etc.)Truy cập dễ dàng đến các service từ bất kỳ nơi nào.

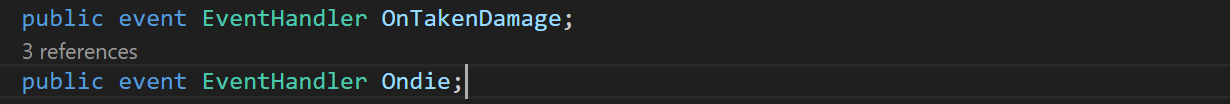
Với kiểu kiến trúc phần mềm như vậy em đã áp dụng vào sản phẩm để tạo ra các thành phần giúp quản lí như : Quản lí tạo ra đối tượng game (Object Pooling ) , Quản lí tạo vật phẩm (Item Pooling ) , Quản lí hiệu ứng âm thanh SFX (SFX Manager) ,..

A grey sign with white text

AI-generated content may be incorrect.

+ **Event System (Giao tiếp giữa các hệ thống) để** Giảm sự phụ thuộc lẫn nhau giữa các thành phần .Đơn giản hóa luồng dữ liệu và thông báo trong game.

Đối với kiểu triển khai game như vây em ứng ựng dụng vào việc tạo ra các sự kiện cục bộ giúp thộng báo tới những lớp kế thừa hoặc sở hữu liên kết đến đối tượng phát sự kiện thông báo chết để các đối tượng muốn xử lí tiến hành bắt và sự lí sự kiện chết :



Hàm xử lí khi nhận dame :

A computer screen with text on it

AI-generated content may be incorrect.

+ **Layered (Tiered) Architecture** để phân chia dự án thành các lớp và việc tương tác giữa các lớp thông qua interface , abstraction , và liên kết giữa các Component .Em chia thành các lớp như sau .

1.**Presentation Layer**: UI, HUD, dialog, menu…

**2.Gameplay Layer**: combat system, inventory system , quest System, Move System,..

**3.Data Layer**: ScriptableObject, JSON, database , ...

**4.Infrastructure Layer**: AudioService, NetworkManager, PoolManager, Logging…

Ví dụ : Sự tương tác giữa các lớp: "UI ở Presentation Layer chỉ đọc dữ liệu từ Gameplay Layer (ví dụ: HealthBar đọc HP từ PlayerStats) và gửi input xuống (ví dụ: click nút dùng Item). Gameplay Layer xử lý logic và có thể yêu cầu Data Layer lưu/tải dữ liệu."

+ **Data‑Driven Architecture** để tạo **ScriptableObject** hoặc **JSON/XML** cho mọi dữ liệu (item, skill, enemy stats, dialog tree…) và Logic engine (code) chỉ đọc “data” để khởi tạo và chạy ngoài ra designer có thể chỉnh sửa asset mà không đụng code.

## 3.6 Phân tích thiết kế hệ thống

Sử dụng ngôn ngữ mô hình hóa hợp nhất (UML) để tạo sơ đồ lớp, sơ đồ tuần tự, sơ đồ hoạt động.

### 3.6.1. Sơ đồ Use Case cho hệ thống :

* **Mục đích:** Xác định các chức năng chính mà hệ thống game cung cấp cho người chơi (Actor). Cho cái nhìn tổng quan về những gì game có thể làm.
* **Thành phần:**
  + **Actor (Tác nhân):** Người hoặc hệ thống tương tác với game. Trong trường hợp của bạn, Actor chính là Người Chơi (Player).
  + **Use Case (UC sử dụng):** Một chức năng hoặc một chuỗi hành động mà game thực hiện, mang lại kết quả có giá trị cho Actor. Được biểu diễn bằng hình elip.
  + **Relationship (Quan hệ):**
    - **Association (Kết hợp):** Đường nối giữa Actor và Use Case (Actor thực hiện Use Case đó).
    - **Include (Bao gồm):** Một Use Case này bắt buộc phải thực hiện một Use Case khác. (Ít dùng trong game đơn giản).
    - **Extend (Mở rộng):** Một Use Case có thể được mở rộng bởi một Use Case khác trong một điều kiện nhất định. (Ít dùng trong game đơn giản).

A blue square with black text

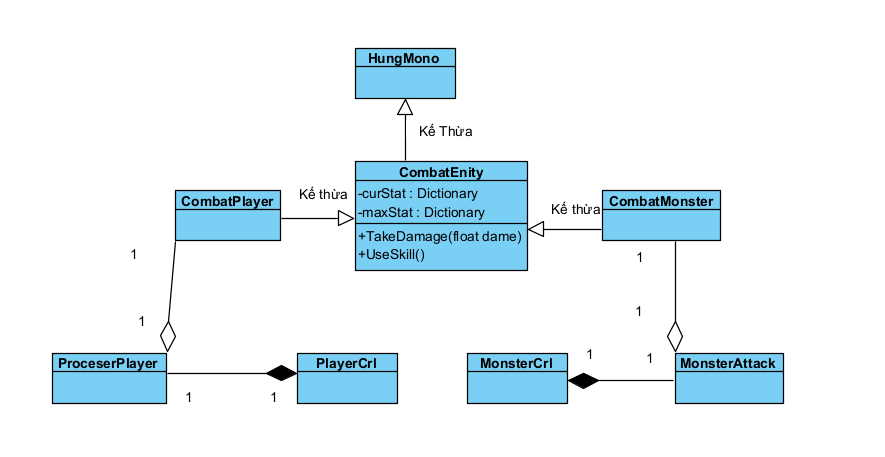
AI-generated content may be incorrect.

### 3.6.2. Sơ đồ lớp :

**Mục đích:** Mô tả cấu trúc tĩnh của hệ thống game: các lớp (classes) quan trọng, thuộc tính (attributes) và phương thức (methods) của chúng, cũng như mối quan hệ giữa các lớp. Đây là sơ đồ quan trọng nhất cho lập trình viên.

Dưới đây em đã viết sơ đồ lớp cho một số hệ thống tiêu biểu và quan trọng trong game của em .

### A . Sơ đồ lớp hệ thống chiến đấu :



**Mô tả về sơ đồ:**

Ở sơ đồ hệ thống chiến đấu trên em đã tạo một lớp là CombatEnity là lớp abstract để :

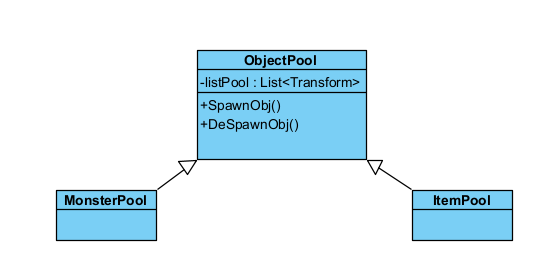
+Định nghĩa các các thuộc tính để lưu trữ các giá trị tối đa của các thuộc tính cơ bản của đối tượng , tiếp đến em định nghĩa các thuộc tính để lưu các giá trị động của các thuộc tính như HP , Saitama

+Định nghĩa các phương thức để nhận sát thương khi bị tấn công (TakeDame) , định nghĩa các phương thức để tấn công (Use Skill)

Đối với các đối tượng muốn sử dụng thì chỉ cần kế thừa từ phương thức trên, sửa đổi đi để sử dụng.Ở hệ thống trên em có 2 lớp là CombatPlayer và CombatMonster kế thừa lớp trên.

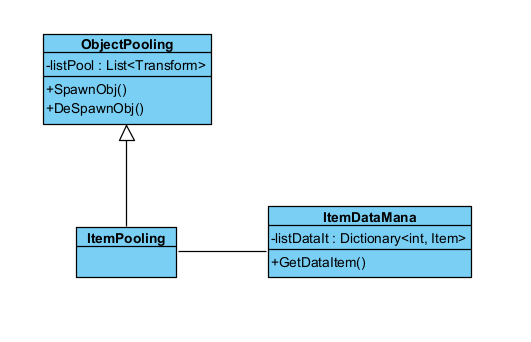
Các lớp ProceserPlayer và MonsterAttack sẽ chứa các lớp trên để tiến hành sử dụng.Khi đối tượng Player tiến hành sử dụng hàm UseSkill của lớp CombatPlayer thì hàm trên sẽ quét toàn bộ đối tượng Monster và tìm ra lớp CombatMonster để gọi hàm TakeDame của lớp đó.

### B . Sơ đồ lớp hệ thống Pooling Object



Lớp ObjectPool là một lớp abtract sẽ định nghĩa các thuộc tính cơ và phương thức để phục vụ cho việc tạo và huy đối tượng bằng Pooling Pattern . Các lớp khác muốn sử dụng chỉ cần kế thừa ví dụ như MonsterPool , ItemPool và tiến hành override nếu cần để sử dụng chức năng của lớp Object Pooling.

### C . Sơ đồ lớp hệ thống tạo Item



Khi có yêu cầu tạo Item thì ItemPooling sẽ lấy dữ liệu về Item như : tên , icon , các giá trị thuộc tính kèm theo ,... thông qua hàm GetDataItem của lớp ItemDataMana sau khi có dữ liệu về Item thì ItemPooling sẽ tiến hành gọi hàm SpawnObj() để tạo đối tượng khuôn cho Item và truyền dữ liệu về Item vào khuân của Item vừa tạo .

### 3.6.3 Sơ đồ tuần tự

Sơ đồ tuần tự mô tả cách hoạt động của hệ thống Pooling ITem:

A diagram of a diagram

AI-generated content may be incorrect.

Sơ đồ tuần tự của hệ thống combat :

A diagram of a computer program

AI-generated content may be incorrect.

# KẾT LUẬN, KIẾN NGHỊ

**1. Kết quả đạt được:**

* + Đã cơ bản xây dựng được một game Unity thể loại RPG cơ bản có các tính năng như di chuyển , tấn công , nhặt vật phẩm ,sử dụng vật phẩm ,...
  + Sau môn học em đã học thêm về kĩ năng quản lí thời gian , kĩ năng lên các timeline cho từng đầu công việc một cách hợp lí nhất , biết cách phân chia các công việc lớn thành các công việc nhỏ hơn.
  + Học thêm và được thực hành thêm về tư duy xây dựng và phát triển một hệ thống phần mềm có thể sử dụng được.

**2. Hạn chế và Khó khăn gặp phải:**

* + Những chức năng như nâng cấp vật phẩm , bang hội , chơi online , tính năng hoạt ảnh cho từng skill một cách đẹp mắt và mượt mà , ...

**3. Đề xuất hướng phát triển thêm cho game:**

* + Cải tiến đồ họa, âm thanh.
  + Thêm nhiều loại nhân vật, quái vật, kỹ năng, vật phẩm, nhiệm vụ.
  + Phát triển AI thông minh hơn cho NPC và kẻ địch.
  + Xây dựng cốt truyện sâu sắc hơn.
  + Mở rộng bản đồ, thêm các khu vực mới.
  + Port game lên nền tảng mobile .

# TÀI LIỆU THAM KHẢO

* Unity Technologies. (2023). *Unity User Manual*. [https://docs.unity3d.com/Manual/index.html](https://www.google.com/url?sa=E&q=https%3A%2F%2Fdocs.unity3d.com%2FManual%2Findex.html)
* Unity Technologies. (2023). *Unity Scripting API*. [https://docs.unity3d.com/ScriptReference/index.html](https://www.google.com/url?sa=E&q=https%3A%2F%2Fdocs.unity3d.com%2FScriptReference%2Findex.html)
* Schell, J. (2019). *The Art of Game Design: A Book of Lenses*. CRC Press.
* Nystrom, R. (2014). *Game Programming Patterns*. Genever Benning. (Nếu bạn có đọc về các pattern)

# PHỤ LỤC CÀI ĐẶT VÀ TRIỂN KHAI

## Phụ lục 1: Thiết lập môi trường

Hướng dẫn cài đặt Unity

Cài đặt các phần mềm hỗ trợ như Visual Studio cho lập trình C#

## Phụ lục 2: Cài đặt triển khai hệ thống

Hướng dẫn cài đặt game trên PC hoặc mobile

# KẾT LUẬN CHUNG

# TÀI LIỆU THAM KHẢO