**LỜI CẢM ƠN**

Em xin chân thành cảm ơn Khoa Công Nghệ Thông Tin, trường Đại Duy Tân đã tạo điều kiện tốt cho em thực hiện đề tài này.

Em xin chân thành cảm ơn Thầy Trịnh Hiệp Hòa, là người đã định hướng và giúp đỡ em trong suốt thời gian thực hiện đề tài. Trong quá trình thực hiện đề tài thầy đã tận tình chỉ dẫn, trao đổi giúp em giải quyết các vấn đề để hoàn thiện đề tài. Em cũng xin gửi lời cảm ơn sâu sắc đến quý thầy cô trong khoa đã tận tình giảng dạy và trang bị cho em vốn kiến thức vô cùng quí báu trong những năm học vừa qua. Cha, Mẹ, anh chị đã quan tâm, chăm sóc, động viên. Bạn bè đã ủng hộ, giúp đỡ chúng em trong những lúc khó khăn cũng như trong suốt thời gian học tập và nghiên cứu. Mặc dù em đã cố gắng hoàn thành luận văn trong phạm vi và khả năng cho phép, nhưng chắc chắn sẽ không tránh khỏi những thiếu sót, kính mong sự cảm thông và chỉ bảo của quý thầy cô và các bạn.

Đà Nẵng, ngày 16 tháng 3 năm 2025.

Sinh viện thực hiện

Đoàn Quang Đăng

LỜI CAM ĐOAN

Tôi xin cam đoan :

1. Những nội dung trong luận văn này là do tôi thực hiện dưới sự hướng dẫn trực tiếp của thầy Trịnh Hiệp Hòa.
2. Mọi tham khảo dùng trong luận văn đều được trích dẫn rõ ràng và trung thực tên tác giả, tên công trình, thời gian, địa điểm công bố.
3. Mọi sao chép không hợp lệ, vi phạm quy chế đào tạo, hay gian trá,  
   tôi xin chịu hoàn toàn trách nhiệm.

Đà Nẵng, ngày 22 tháng 3 năm 2025.

Sinh viện thực hiện

Đoàn Quang Đăng

# MỞ ĐẦU

**1. Lý do chọn đề tài**

**2. Mục đích và ý nghĩa của đề tài**

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

**4. Phương pháp nghiên cứu**

**5. Kết quả dự kiến**

**6. Bố cục đề tài**

*Đà Nẵng, ngày 16 tháng 3, năm 2025.*

# CHƯƠNG I

# TỔNG QUAN VỀ CÔNG CỤ VÀ MỖI TRƯỜNG PHÁT TRIỂN

## **GIỚI THIỆU NGÔN NGỮ LẬP TRÌNH C#**

### Lịch sử phát triển của ngôn ngữ lập trình C#

### C# là gì?

* C# (hay C sharp) là một ngôn ngữ lập trình đơn giản, được phát triển bởi đội ngũ kỹ sư của Microsoft vào năm 2000, trong đó người dẫn đầu là Anders Hejlsberg và Scott Wiltamuth.
* C# là ngôn ngữ lập trình hiện đại, hướng đối tượng và nó được xây dựng trên nền tảng của hai ngôn ngữ mạnh nhất là C++ và Java.
* C# được thiết kế cho Common Language Infrastructure (CLI), mà gồm Executable Code và Runtime Environment, cho phép chúng ta sử dụng các ngôn ngữ high-level đa dạng trên các nền tảng và cấu trúc máy tính khác nhau.
* C# với sự hỗ trợ mạnh mẽ của .NET Framework giúp cho việc tạo một ứng dụng Windows Forms hay WPF (Windows Presentation Foundation), . . . trở nên rất dễ dàng.

### Một số đặc điểm nổi bậc của ngôn ngữ lập trình C#.

### 1.1.2 Đặc điểm ngôn ngữ:

* C# theo một hướng nào đó, là ngôn ngữ lập trình phản ánh trực tiếp nhất đến [.NET Framework](https://vi.wikipedia.org/wiki/.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

### 1.1.3 Đặc trưng của ngôn ngữ C#.

* C# là ngôn ngữ đơn giản, mạnh mẽ
  + C# được dựng trên nền tảng C++ và Java, ảnh hưởng bởi Delphi, VisualBasic nên ngôn ngữ C# được thừa hưởng các ưu điểm vào loại bỏ các yếu điểm của các ngôn ngữ trên, vì vậy nó khá đơn giản, đồng thời loại bỏ các cú pháp dư thừa và thêm vào đó các cú pháp cải tiến hơn
  + C# là [ngôn ngữ lập trình bậc cao](https://vi.wikipedia.org/wiki/Ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh_b%E1%BA%ADc_cao#:~:text=Trong_khoa_h%E1%BB%8Dc_m%C3%A1y_t%C3%ADnh,chi_ti%E1%BA%BFt_c%E1%BB%A7a_m%C3%A1y_t%C3%ADnh.), [đa nền tảng](https://vi.wikipedia.org/wiki/%C4%90a_n%E1%BB%81n_t%E1%BA%A3ng) vì vậy nó **dễ dàng tiếp cận** và **phù hợp cho người mới bắt đầu học**, ví dụ câu lệnh kinh điển dành cho người mới bắt đầu học là in ra dòng chữ "Hello world", với C# ta chỉ cần 1 câu lệnh: **System.Console.WriteLine("Hello world");**
* C# là ngôn ngữ đa năng và hiện đại
  + C# phù hợp cho việc phát triển trong thời đại 4.0, bao gồm việc phát triển web, mobile app, game, học máy và trí tuệ nhân tạo, phát triển đám mây, IoT, blockchain, microservices...
* C# là một ngôn ngữ lập trình hướng đối tượng đồng thời hỗ trợ lập trình chức năng
  + C# hỗ trợ mạnh mẽ cho phương pháp lập trình hướng đối tượng, ngoài ra C# còn hỗ trợ các phương pháp lập trình chức năng thông qua các biểu thức lamba, khớp mẫu, functions, các thuộc tính bất biến.
* C# là ngôn ngữ gõ tĩnh, định kiểu mạnh, hỗ trợ gõ động.
  + C# được gõ tĩnh nên nó mang đầy đủ các ưu việt của phương pháp gõ tĩnh như bảo đảm an toàn kiểu, tự động phân tích và nhận biết lỗi cú pháp ngay trong quá trình viết mã...
  + Ngoài ra khi sử dụng C# kết hợp với IDE Visual Studio, C# được hỗ trợ gợi ý code bởi [Visual Studio IntelliCode](https://visualstudio.microsoft.com/services/intellicode/) sử dụng trí tuệ nhân tạo giúp cho việc viết code trở nên nhanh chóng và dễ dàng hơn
* C# là một ngôn ngữ ít [từ khóa](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/)
  + C# có khoảng hơn 80 từ khóa
* C# là một trong các ngôn ngữ lập trình phổ biến và phát triển nhất
  + Theo [TIOBE Index](https://www.tiobe.com/tiobe-index/), tính đến tháng 10/2020, C# là ngôn ngữ phổ biến thứ 5 thế giới.
  + Theo [PYPL](http://pypl.github.io/PYPL.html), tính đến tháng 10/2020, C# là ngôn ngữ được cộng đồng quan tâm và chia sẻ nhiều thứ 4 thế giới.
  + Theo Github (Kho lưu trữ mã nguồn lớn nhất thế giới), tính đến tháng 10/2020, C# là ngôn ngữ hoạt động nhiều thứ 4.
  + Cộng đồng phát triển và số người theo học ngôn ngữ C# tăng khônng ngừng theo mỗi năm.Theo ước tính 10/2020, cộng đồng phát triển C# là hơn 6 triệu người
* C# kết hợp chặt chẽ với nền tảng .NET - một khung nền tảng được đầu tư rất mạnh của Microsoft.
* Ngoài ra C# còn có những ưu điểm:
  + [C# là ngôn ngữ lập trình mã nguồn mở](https://github.com/dotnet/csharplang),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](https://vi.wikipedia.org/wiki/N%E1%BB%81n_t%E1%BA%A3ng_m%C3%A1y_t%C3%ADnh) khác nhau (Windows, Linux, [MacOS](https://vi.wikipedia.org/wiki/MacOS))
  + C# có hiệu suất cao và tốc độ thực thi nhanh do sử dụng trình biên dich trung gian ([CLR](https://docs.microsoft.com/en-us/dotnet/standard/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](https://docs.microsoft.com/en-us/dotnet/csharp/) 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.

### Các ứng dụng C#.

* Phát triển web [backend](https://vi.wikipedia.org/wiki/Front-end_v%C3%A0_back-end) ([ASP.NET](https://vi.wikipedia.org/wiki/ASP.NET) MVC, ASP.NET core, Web API,Graph API,gPRC, Blazor sevver,Uno platform, Mono)
* Phát triển web [front end](https://vi.wikipedia.org/wiki/Front-end_v%C3%A0_back-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](https://vi.wikipedia.org/wiki/Game_engine):[Unity](https://vi.wikipedia.org/wiki/Unity_(game_engine)), Monogame, Godot, Stride, CryEngine, Flax Engine, Evergine, 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](https://vi.wikipedia.org/wiki/%E1%BB%A8ng_d%E1%BB%A5ng_tr%C3%AAn_thi%E1%BA%BFt_b%E1%BB%8B_di_%C4%91%E1%BB%99ng), IOS native, [Android](https://vi.wikipedia.org/wiki/Android_(h%E1%BB%87_%C4%91i%E1%BB%81u_h%C3%A0nh)) native (Xamarin,MAUI,Uno platform)
* Phát triển [đám mây](https://vi.wikipedia.org/wiki/%C4%90i%E1%BB%87n_to%C3%A1n_%C4%91%C3%A1m_m%C3%A2y) ([Azure](https://azure.microsoft.com/en-us/),AWS,Google Cloud...)
* Học máy và [trí tuệ nhân tạo](https://vi.wikipedia.org/wiki/Tr%C3%AD_tu%E1%BB%87_nh%C3%A2n_t%E1%BA%A1o) ([ML.Net](https://dotnet.microsoft.com/apps/machinelearning-ai/ml-dotnet), TensorFlow, csiSharp..)
* Data science, bigdata (csiSharp, Apache Spark)
* [Blockchain](https://vi.wikipedia.org/wiki/Blockchain) ([NEO, Stratis](https://www.stratisplatform.com/))
* Microservices and containers
* Internet of thing (IoT,5G)
* Hệ thống nhúng ([Raspberry pi](https://vi.wikipedia.org/w/index.php?title=Raspberry_pi&action=edit&redlink=1), PLC)

## **GIỚI THIỆU ENGINE UNITY 6**

### Lịch sử phát triển

* Unity là một game engine đa nền tảng được phát triển bởi Unity Technologies, chủ yếu để phát triển video game cho máy tính, consoles và thiết bị di động. Ban đầu, Unity được công bố tại Apple's Worldwide Developers Conference vào năm 2005, chỉ hỗ trợ hệ điều hành Mac OS X. Kể từ đó, Unity đã mở rộng để hỗ trợ nhiều nền tảng khác như Windows, Linux, iOS, Android, PlayStation, Xbox, và Nintendo Switch.
* Các phiên bản chính của Unity đã lần lượt được phát hành:
* Unity 1.0 (2005): Hỗ trợ Mac OS X, tập trung vào phát triển game 3D.
* Unity 2.0 (2007): Mở rộng sang Windows, hỗ trợ trình duyệt web với Unity Web Player.
* Unity 3.0 (2010): Bổ sung hỗ trợ PlayStation 3, Xbox 360 và Android.
* Unity 4.0 (2012): Cải thiện đồ họa với DirectX 11, Mecanim animation và hỗ trợ Linux.
* Unity 5.0 (2015): Cập nhật Global Illumination, PBR (Physically-Based Rendering) và hỗ trợ WebGL.
* Unity 2017 - 2019: Giới thiệu Timeline, Cinemachine, và công nghệ DOTS (Data-Oriented Technology Stack).
* Unity 2020 - 2023: Nâng cấp HDRP, URP và hỗ trợ ray tracing.
* Unity 6 (2024): Tập trung vào tối ưu hiệu suất, hỗ trợ AI và tích hợp công nghệ hiện đại cho game và metaverse.
  + Hiện nay, Unity là một trong những game engine phổ biến nhất, cạnh tranh trực tiếp với Unreal Engine.

### Các module trong Unity

* Unity bao gồm nhiều module quan trọng giúp lập trình viên phát triển game dễ dàng:
* **Unity Editor:** Giao diện đồ họa chính để xây dựng, chỉnh sửa game.
* **Rendering Engine:** Hỗ trợ HDRP (High Definition Render Pipeline) và URP (Universal Render Pipeline) để tối ưu đồ họa.
* **Physics Engine:** Sử dụng NVIDIA PhysX và DOTS-based physics cho mô phỏng vật lý.
* **Animation System:** Mecanim hỗ trợ tạo và quản lý animation chuyên nghiệp.
* **AI & Navigation:** Công cụ điều hướng NavMesh và tích hợp AI giúp phát triển NPC thông minh hơn.
* **Networking:** Hỗ trợ Multiplayer với Netcode for GameObjects và Unity Transport.
* **AR/VR Support:** Tích hợp công cụ phát triển cho AR (Augmented Reality) và VR (Virtual Reality).
* **Scripting:** Sử dụng ngôn ngữ C# để phát triển gameplay và logic game.

### Các dự án trong Unity

* Unity được sử dụng rộng rãi trong ngành công nghiệp game và nhiều lĩnh vực khác:
* **Game nổi bật:**
  + **Hollow Knight:** Game phiêu lưu hành động với đồ họa đẹp mắt.
  + **Genshin Impact:** Game thế giới mở với đồ họa ấn tượng và hệ thống chiến đấu hấp dẫn.
  + **Among Us:** Game multiplayer nổi tiếng với lối chơi sáng tạo.
  + **Monument Valley:** Game giải đố với thiết kế nghệ thuật đặc sắc.
* **Ứng dụng ngoài game:**
  + **Kiến trúc & Xây dựng:** Unity giúp mô phỏng thiết kế nội thất và công trình kiến trúc.
  + **Giáo dục & Y tế:** Sử dụng để tạo mô phỏng phẫu thuật, hướng dẫn thực tế ảo.
  + **Phim & Hoạt hình:** Công nghệ real-time rendering của Unity giúp tạo hiệu ứng phim nhanh chóng.
* Nhờ sự phát triển mạnh mẽ và cộng đồng hỗ trợ rộng lớn, Unity 6 tiếp tục là một nền tảng lý tưởng để phát triển game và ứng dụng tương tác đa lĩnh vực.

## **GIỚI THIỆU GAME TOWER DEFENSE**

1. **Lịch sử phát triển của Tower Defense.**

* Lịch sử phát triển game tower defense bắt đầu từ những năm 1990 với các trò chơi chiến thuật thời gian thực. Những trò chơi này thường yêu cầu người chơi xây dựng và nâng cấp các tháp để ngăn chặn kẻ thù tiến vào căn cứ. Một trong những trò chơi đầu tiên có thể được coi là tiền thân của thể loại này là "Rampart". Tuy nhiên, thể loại tower defense thực sự bùng nổ vào giữa những năm 2000 với sự ra đời của "Desktop Tower Defense". Trò chơi này đã thu hút hàng triệu người chơi và mở ra một kỷ nguyên mới cho thể loại game này. Kể từ đó, nhiều trò chơi tower defense khác đã được phát triển, bao gồm "Bloons Tower Defense" và "Kingdom Rush", mỗi trò chơi đều mang đến những cơ chế và chiến lược độc đáo, thu hút người chơi trên toàn thế giới.

1. **Tower Defense là gì? Thế nào thì được gọi là game Tower Defense?**
   * Tower Defense (TD) là một thể loại game chiến thuật, trong đó người chơi phải bảo vệ căn cứ của mình khỏi các đợt tấn công của kẻ địch bằng cách xây dựng và nâng cấp các trụ phòng thủ (tower). Những trụ này tự động tấn công kẻ địch khi chúng di chuyển qua bản đồ theo một tuyến đường cố định hoặc mở.
   * **Đặc điểm của game Tower Defense:**
     + **Xây dựng trụ phòng thủ:** Người chơi có thể đặt các trụ ở những vị trí chiến lược để ngăn chặn kẻ địch.
     + **Làn sóng kẻ địch (Wave):** Kẻ địch thường tấn công theo từng đợt, với độ khó tăng dần theo thời gian.
     + **Nâng cấp và chiến thuật:** Người chơi có thể nâng cấp trụ để tăng sức mạnh, tốc độ bắn hoặc thêm hiệu ứng đặc biệt. Việc chọn vị trí đặt trụ và cách nâng cấp ảnh hưởng lớn đến chiến thắng.
     + **Tài nguyên và quản lý:** Game thường yêu cầu người chơi thu thập và quản lý tài nguyên để xây dựng và nâng cấp trụ.
     + **Thế nào thì được gọi là game Tower Defense?**
     + Một trò chơi được gọi là Tower Defense khi có những yếu tố cốt lõi sau:
     + **Hệ thống phòng thủ bằng trụ:** Trò chơi tập trung vào việc xây dựng các trụ để ngăn chặn kẻ địch.
     + **Lối chơi chiến thuật:** Người chơi phải đưa ra quyết định hợp lý về việc đặt trụ và sử dụng tài nguyên để tối ưu hóa khả năng phòng thủ.
     + **Mục tiêu bảo vệ căn cứ:** Trò chơi kết thúc khi căn cứ của người chơi bị phá hủy hoặc khi người chơi đánh bại tất cả các làn sóng kẻ địch.
   * Các trò chơi nổi bật thuộc thể loại này bao gồm *Bloons TD*, *Plants vs. Zombies*, *Kingdom Rush* và nhiều tựa game khác.

## **GIỚI THIỆU MYSQL**

### 1.4.1 MySQL là gì?

* MySQL là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) mã nguồn mở, được sử dụng rộng rãi để lưu trữ và quản lý dữ liệu. Nó hỗ trợ **SQL (Structured Query Language)** để truy vấn và thao tác dữ liệu.

### 1.4.2 MySQL hoạt động như thế nào?

* **Cấu trúc bảng (Table-Based)**: Dữ liệu được lưu trữ dưới dạng bảng có hàng và cột.
* **Hệ thống truy vấn SQL**: Sử dụng lệnh như SELECT, INSERT, UPDATE, DELETE để thao tác dữ liệu.
* **Hỗ trợ ACID**: Đảm bảo tính toàn vẹn dữ liệu qua Atomicity, Consistency, Isolation, Durability.
* **Kết nối với ứng dụng**: MySQL có thể kết nối với nhiều ngôn ngữ lập trình như **C#, Java, Python, PHP**.

### 1.4.3 Tại sao sử dụng MySQL?

* **Hiệu suất cao**: Tốc độ truy vấn nhanh, tối ưu cho web và ứng dụng lớn.
* **Bảo mật tốt**: Hỗ trợ mã hóa dữ liệu và quyền truy cập theo user.
* **Hỗ trợ đa nền tảng**: Chạy được trên Windows, Linux, macOS.
* **Tích hợp dễ dàng**: Hỗ trợ nhiều framework và công nghệ như **Laravel, Django, .NET**.
* **Cộng đồng mạnh mẽ**: Được sử dụng rộng rãi với tài liệu phong phú.

# CHƯƠNG II

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## **KHẢO SÁT THỰC TRẠNG**

**2.1.1. Tổng quan về thể loại Tower Defense**

* Tower Defense (TD) là một nhánh của thể loại game chiến thuật, trong đó người chơi phải xây dựng hệ thống phòng thủ để ngăn chặn các đợt tấn công của kẻ địch. Dòng game này phổ biến trên cả nền tảng PC, console và mobile nhờ lối chơi đơn giản nhưng mang tính chiến thuật cao.
* Hiện nay, có nhiều tựa game Tower Defense nổi bật như **Bloons TD, Kingdom Rush, Plants vs. Zombies, Defense Grid**, v.v. Những trò chơi này thường có các yếu tố chính như:
* **Xây dựng và nâng cấp trụ phòng thủ** để chặn kẻ địch.
* **Hệ thống tài nguyên** giúp người chơi xây dựng chiến lược hợp lý.
* **Cấp độ khó tăng dần** qua từng màn chơi.
* **Cơ chế quái vật theo từng đợt (Wave System)** tạo sự thử thách cho người chơi.

**2.1.2. Thực trạng game Tower Defense trên thị trường**

* Mặc dù dòng game Tower Defense đã tồn tại từ lâu, nhưng vẫn có sự đổi mới về lối chơi và đồ họa để phù hợp với xu hướng hiện đại. Một số xu hướng hiện nay bao gồm:
* **Kết hợp với yếu tố RPG**: Một số game cho phép người chơi nâng cấp nhân vật hoặc có các anh hùng hỗ trợ chiến đấu.
* **Lối chơi đa dạng hơn**: Bên cạnh cơ chế truyền thống, một số game cho phép di chuyển tháp hoặc thay đổi chiến thuật linh hoạt hơn.
* **Đồ họa và âm thanh nâng cao**: Sự cải tiến về mặt hình ảnh giúp game trở nên hấp dẫn hơn, đặc biệt trên nền tảng mobile.
* **Mô hình kinh doanh miễn phí (Free-to-Play)**: Nhiều game TD hiện nay áp dụng mô hình này, kết hợp với quảng cáo hoặc mua vật phẩm trong game.

**2.1.3. Hạn chế của các game Tower Defense hiện nay**

* Mặc dù thể loại Tower Defense có nhiều ưu điểm nhưng vẫn còn một số hạn chế:
* **Thiếu sự sáng tạo**: Nhiều game có lối chơi lặp lại, không có điểm khác biệt rõ ràng.
* **Độ khó mất cân bằng**: Một số game quá dễ hoặc quá khó, không có sự điều chỉnh phù hợp theo kỹ năng người chơi.
* **Phụ thuộc vào giao dịch trong game**: Một số trò chơi Free-to-Play buộc người chơi phải mua vật phẩm để có trải nghiệm tốt hơn, gây mất cân bằng.

**2.1.4. Hướng phát triển của game Tower Defense**

* Với những xu hướng và hạn chế trên, các game Tower Defense trong tương lai có thể tập trung vào:
* **Tạo ra cơ chế gameplay mới mẻ**, kết hợp yếu tố chiến thuật khác để tăng sự đa dạng.
* **Phát triển AI thông minh hơn**, giúp quái vật có thể phản ứng linh hoạt thay vì chỉ đi theo một lộ trình cố định.
* **Cải thiện trải nghiệm người chơi**, như hệ thống nhiệm vụ đa dạng, đồ họa sống động và khả năng chơi nhiều người.

## **MỘT SỐ TỰA GAME TOWER DEFENSE NỔI BẬT TRÊN THỊ TRƯỜNG**

* Hiện nay, có nhiều tựa game Tower Defense thành công và được cộng đồng game thủ yêu thích. Dưới đây là một số trò chơi tiêu biểu:

**1. Bloons TD Series (Bloons TD 6)**



* **Nhà phát triển**: Ninja Kiwi
* **Nền tảng**: PC, Mobile
* **Mô tả**: Đây là một trong những series Tower Defense phổ biến nhất với lối chơi chiến thuật hấp dẫn. Người chơi đặt các trụ phòng thủ là những con khỉ có khả năng đặc biệt để ngăn chặn bong bóng (bloons) đi qua bản đồ. Bloons TD 6 được đánh giá cao nhờ đồ họa sinh động, hệ thống nâng cấp phong phú và chế độ co-op nhiều người chơi.

**2. Kingdom Rush Series**



* **Nhà phát triển**: Ironhide Game Studio
* **Nền tảng**: PC, Mobile
* **Mô tả**: Kingdom Rush là một trong những tựa game Tower Defense thành công nhất, nổi bật với lối chơi chiến thuật đặc sắc và phong cách đồ họa vẽ tay. Game có nhiều loại tháp phòng thủ, anh hùng hỗ trợ chiến đấu, và hệ thống kẻ địch đa dạng.

**3. Plants vs. Zombies**



* **Nhà phát triển**: PopCap Games (sau này thuộc EA)
* **Nền tảng**: PC, Mobile, Console
* **Mô tả**: Dù không hoàn toàn giống với Tower Defense truyền thống, nhưng Plants vs. Zombies vẫn có cơ chế phòng thủ bằng việc đặt cây cối để chống lại zombie. Lối chơi sáng tạo, vui nhộn cùng với sự đa dạng về chiến thuật đã giúp game trở thành một trong những tựa game được yêu thích nhất trên toàn thế giới.

## **CÁC ĐỐI TƯỢNG TƯƠNG TÁC VỚI HỆ THỐNG**

Hệ thống có các tác nhân: người dùng chưa đăng ký, người dùng đã đăng ký và người quản trị. Vai trò của các tác nhân được thể hiện trong bảng như sau:

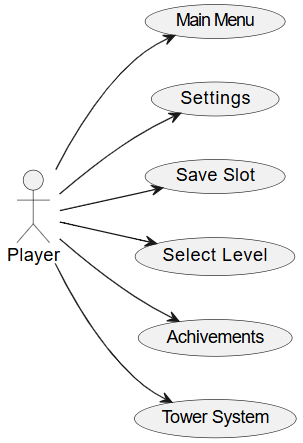
*Bảng 2.1 Các tác nhân của hệ thống*

|  |  |  |
| --- | --- | --- |
| STT | Tên tác nhân | Mô tả |
| 1 | Người chơi | **Tương tác với Main Menu**:   * Bắt đầu trò chơi (Start Game). * Xem thông tin nhóm phát triển (Credits). * Cài đặt âm thanh trong phần Settings. * **Chọn Save Slot**: * Chọn một trong ba slot để lưu tiến trình hoặc tiếp tục chơi.   **Tương tác với bản đồ**:   * Chọn level để chơi. * Xem thông tin về các thực thể trong game (Book). * Kiểm tra thành tích (Achievements).   **Chơi trong Level**:   * Bắt đầu màn chơi. * Xây dựng, nâng cấp, hoặc bán tháp phòng thủ. * Điều chỉnh âm lượng hoặc thoát về Main Menu thông qua Settings. |
| 2 | Admin | **Quản lý level và gameplay**:   * Thêm, chỉnh sửa hoặc xóa các màn chơi. * Điều chỉnh thông số của kẻ địch, tháp phòng thủ, tài nguyên để cân bằng game.   **Quản lý nội dung trong game**:   * Cập nhật thông tin các thực thể trong mục “Book”. * Quản lý phần Achievements, thiết lập điều kiện mở khóa.   **Quản lý hệ thống**:   * Kiểm tra và xử lý lỗi trong game. * Cập nhật hoặc điều chỉnh các thiết lập chung của trò chơi. |

## **SƠ ĐỒ USECASE SỬ DỤNG (USECASE DIAGRAM)**

1. **Use case tổng quát**

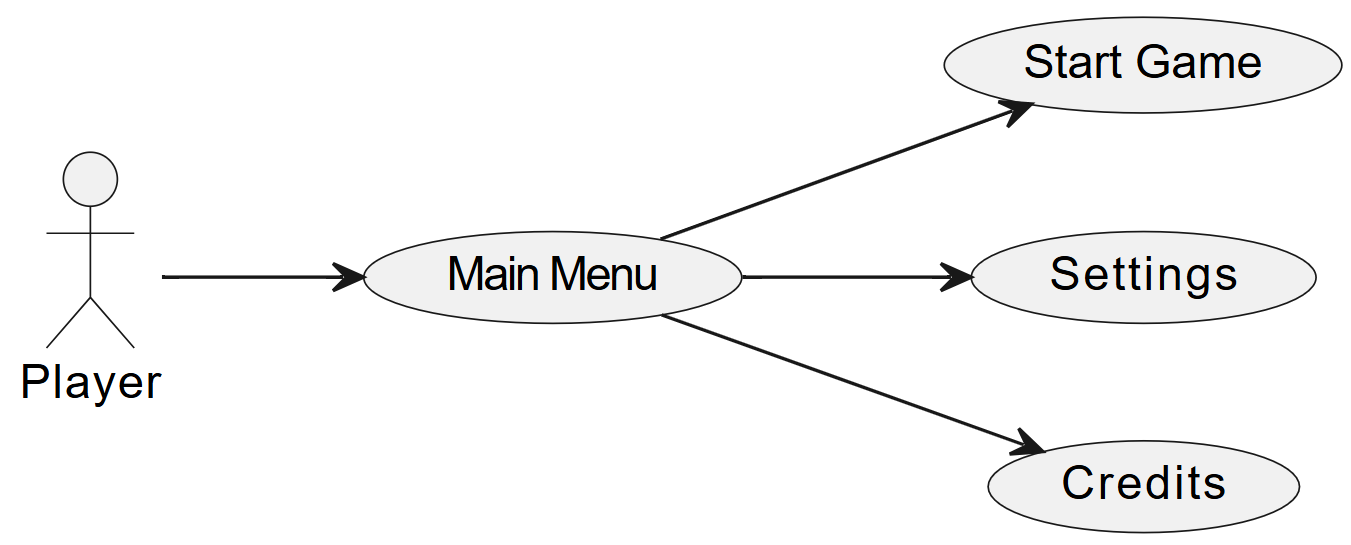
* Biểu đồ Use case người chơi



*Hình 2.4.1.a Use case người chơi*

1. **Use case chi tiết**

#### *Use case Main Menu:*



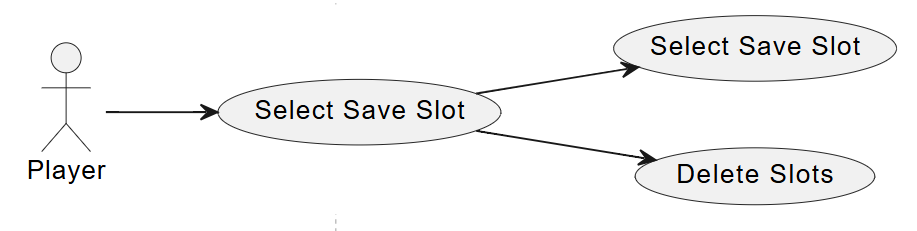
*Hình 2.4.2.1 Use case Main menu*

* **Mục đích:** Cho phép người chơi tương tác với main menu
* **Tác nhân, mô tả chung:**
  + **Tác nhân:** Người chơi
  + **Mô tả chung:** Mô tả việc người chơi thực hiện tương tác với main meu
* **Luồng sự kiện chính:**

Bảng 2.4.2.1 Luồng sự kiện chính use case main menu

|  |  |
| --- | --- |
| Hành động của tác nhân | Phản ứng của hệ thống |
| 1. Người dùng chọn vào Start game.  2. Người dùng chọn vào Settings  3. Người dùng chọn vào Credits | 1. Hiển thị màn hình save slots  2. Hiển thị một panel tắt bật âm thanh  3. Hiển thị credits của game |

#### *Use case Save Slots*



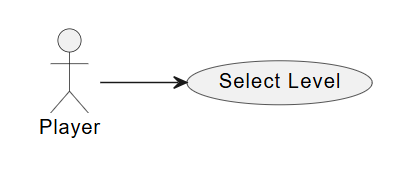
*Hình 2.4.2.2 Use case Main menu*

* **Mục đích:** Cho phép người chơi quản lý và tải dữ liệu lưu trữ
* **Tác nhân, mô tả chung:**
  + **Tác nhân:** Người chơi
* **Mô tả chung:** Cho phép người chơi quản lý và tải dữ liệu lưu trữ
* **Luồng sự kiện chính:**

Bảng 2. 4.2.2 Luồng sự kiện chính use case Save Slots

|  |  |
| --- | --- |
| Hành động của tác nhân | Phản ứng của hệ thống |
| 1. Người dùng chọn vào Save Slots  2. Người dùng chọn vào Delete Slots | 1. khi Player chọn save slots thì sẽ vào save system check xem data có tồn tại hay ko, khi data tồn tại thì sẽ load data lên cho actor, còn khi data rỗng thì sẽ create data mới và load data lên cho Player  2. Delete thì xóa đi data rồi trả lại data rỗng cho Player |

#### *Use case Select Level*



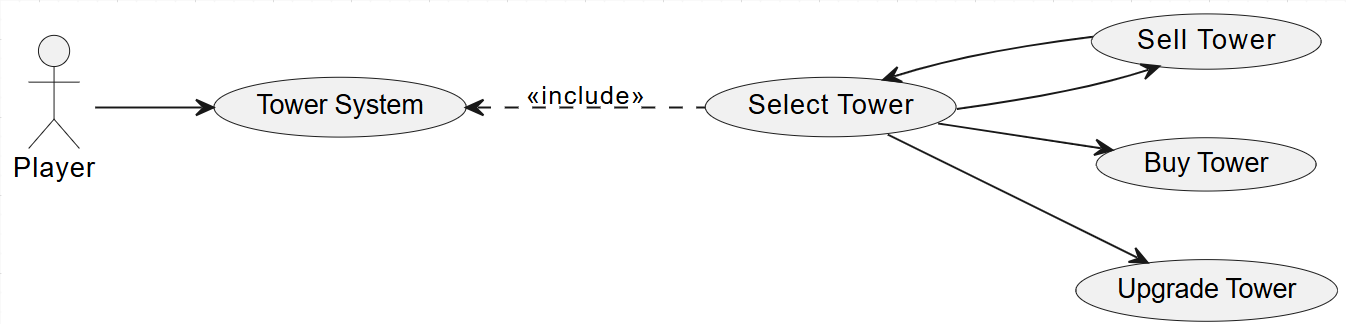
*Hình 2.4.2.3 Use case Select Level*

* **Mục đích:** Cho phép người chơi chọn Level
* **Tác nhân, mô tả chung:**
  + **Tác nhân:** Người chơi
* **Mô tả chung:** Cho phép người chơi chọn Level
* **Luồng sự kiện chính:**

Bảng 2. 4.2.3 Luồng sự kiện chính use case chọn Level

|  |  |
| --- | --- |
| Hành động của tác nhân | Phản ứng của hệ thống |
| 1. Người dùng chọn Level | 1. Hiển thị màn hình game của Level mà người chơi đã chọn |

#### *Use case Tower System*



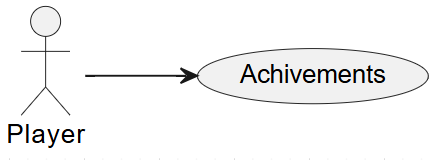
*Hình 2.4.2.3 Use case Select Level*

* **Mục đích:** Cho phép người chơi sử dụng hệ thống tower của game
* **Tác nhân, mô tả chung:**
  + **Tác nhân:** Người chơi
* **Mô tả chung:** Cho phép người chơi chọn tower và mua, bán hoặc nâng cấp
* **Luồng sự kiện chính:**

Bảng 2. 4.2.3 Luồng sự kiện chính use case chọn Level

|  |  |
| --- | --- |
| Hành động của tác nhân | Phản ứng của hệ thống |
| 1. Người dùng chọn Select Tower  2. Người dùng chọn Buy Tower  3. Người dùng chọn Upgrade Tower  4. Người dùng chọn Sell Tower | 1. Hiển thị thông tin của các Tower  2. Mua Tower mà người dùng select trước đó  3. Nâng cấp Tower  4. Bán Tower và quay lại việc select tower |

#### *Use case Achivements*



*Hình 2.4.2.5 Use case Achivements*

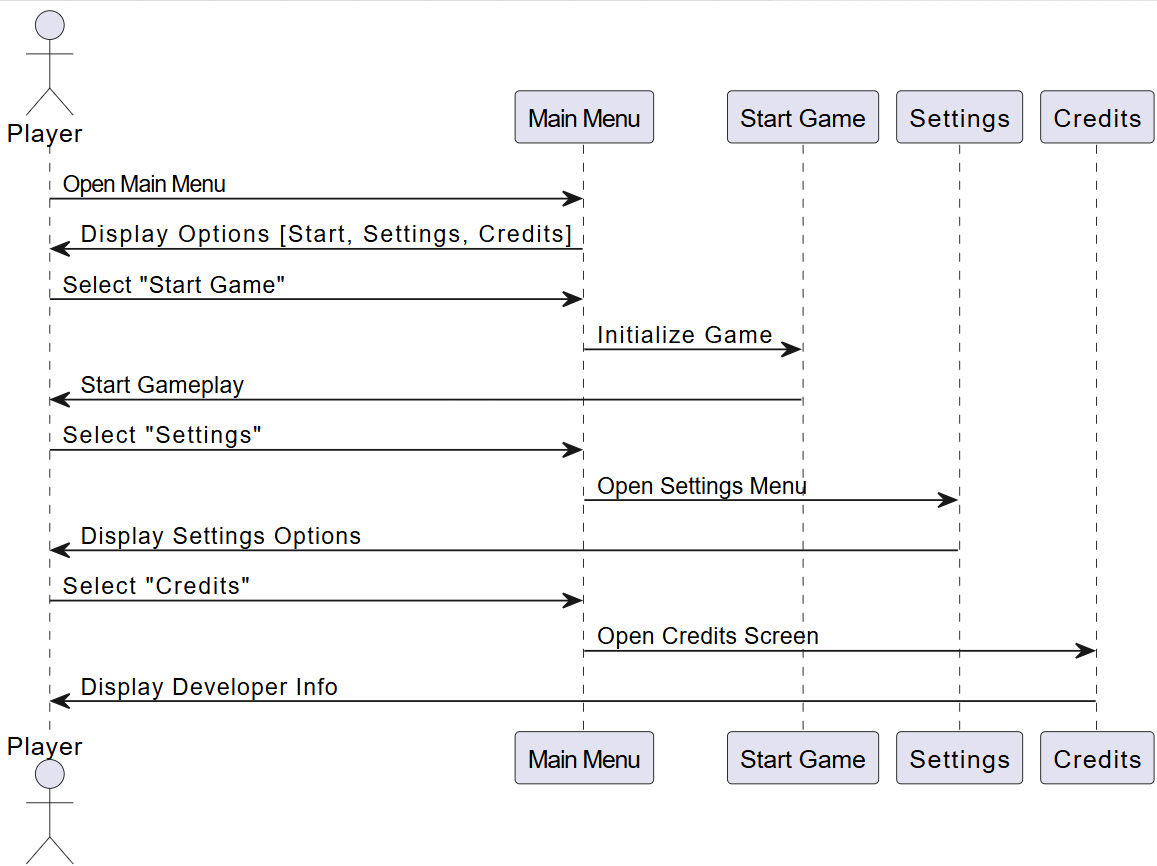
* **Mục đích:** Cho phép người chơi xem Achivements
* **Tác nhân, mô tả chung:**
  + **Tác nhân:** Người chơi
* **Mô tả chung:** Cho phép người chơi xem Achivements
* **Luồng sự kiện chính:**

Bảng 2. 4.2.5 Luồng sự kiện chính use case chọn Level

|  |  |
| --- | --- |
| Hành động của tác nhân | Phản ứng của hệ thống |
| 1. Người dùng chọn Level | 1. Hiển thị màn hình xem Achivements người chơi có thể xem các thành tựu mà mình đã đạt được trong game |

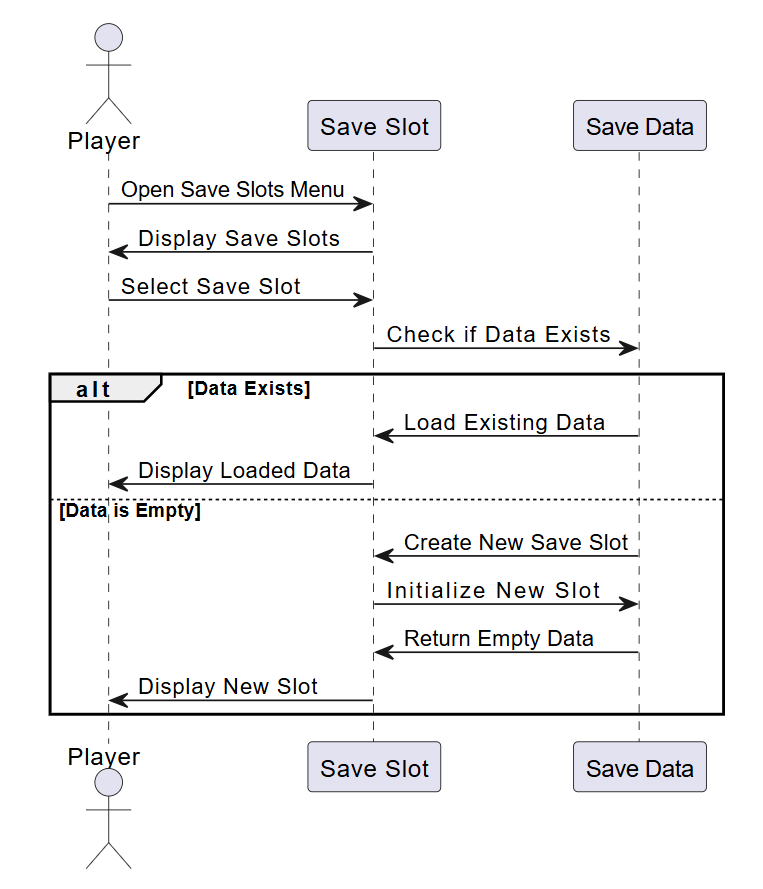
## **SƠ ĐỒ TUẦN TỰ (SEQUENSE DIAGRAM)**

### Sơ đồ tuần tự Main meu



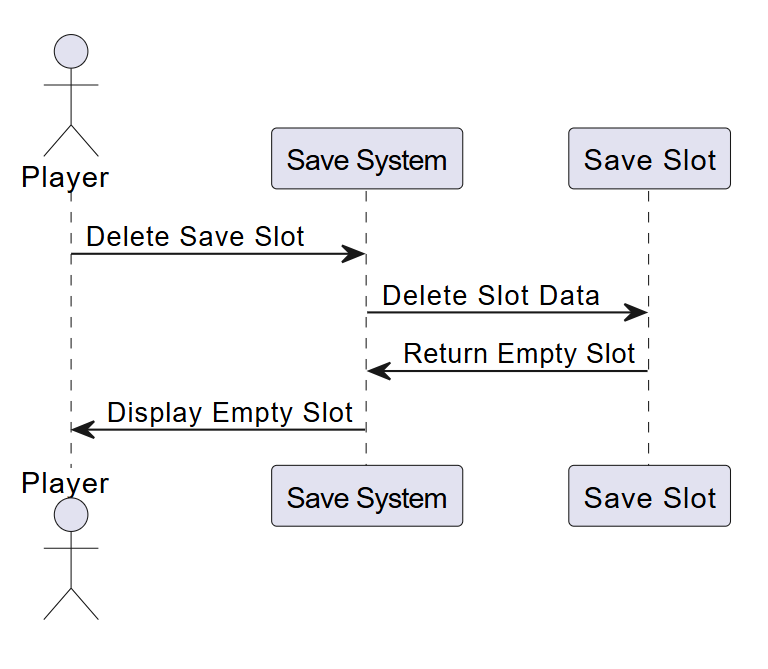
*Hình 2.5.1 Sơ đồ tuần tự Main menu*

### Sơ đồ tuần tự Save slots



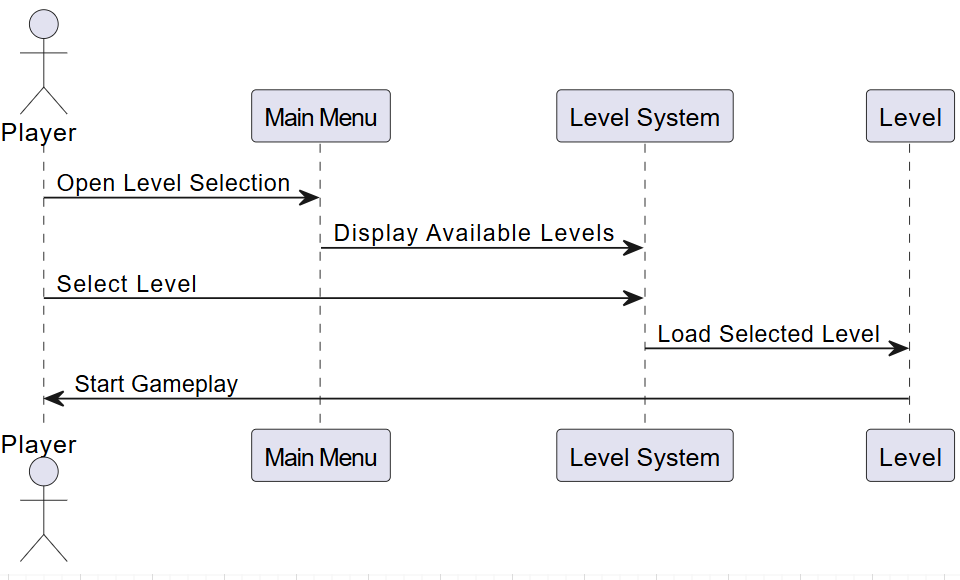
*Hình 2.5.2 Sơ đồ tuần tự Save slots*

### Sơ đồ tuần tự Delete Slots



*Hình 2.5.3 Sơ đồ tuần tự Delete Slots*

### Sơ đồ tuần tự Select level



*Hình 2.5.4 Sơ đồ tuần tự Select level*

### Sơ đồ tuần tự Buy Tower

*Hình 2.5.5 Sơ đồ tuần tự Tower System*

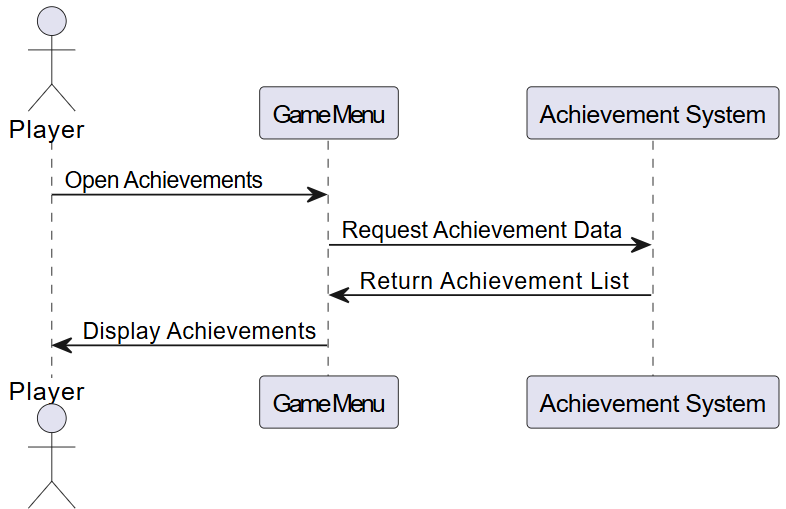
### Sơ đồ tuần tự Upgrade Tower

*Hình 2.5.6 Sơ đồ tuần tự Tower System*

### Sơ đồ tuần tự Sell Tower

*Hình 2.5.7 Sơ đồ tuần tự Tower System*

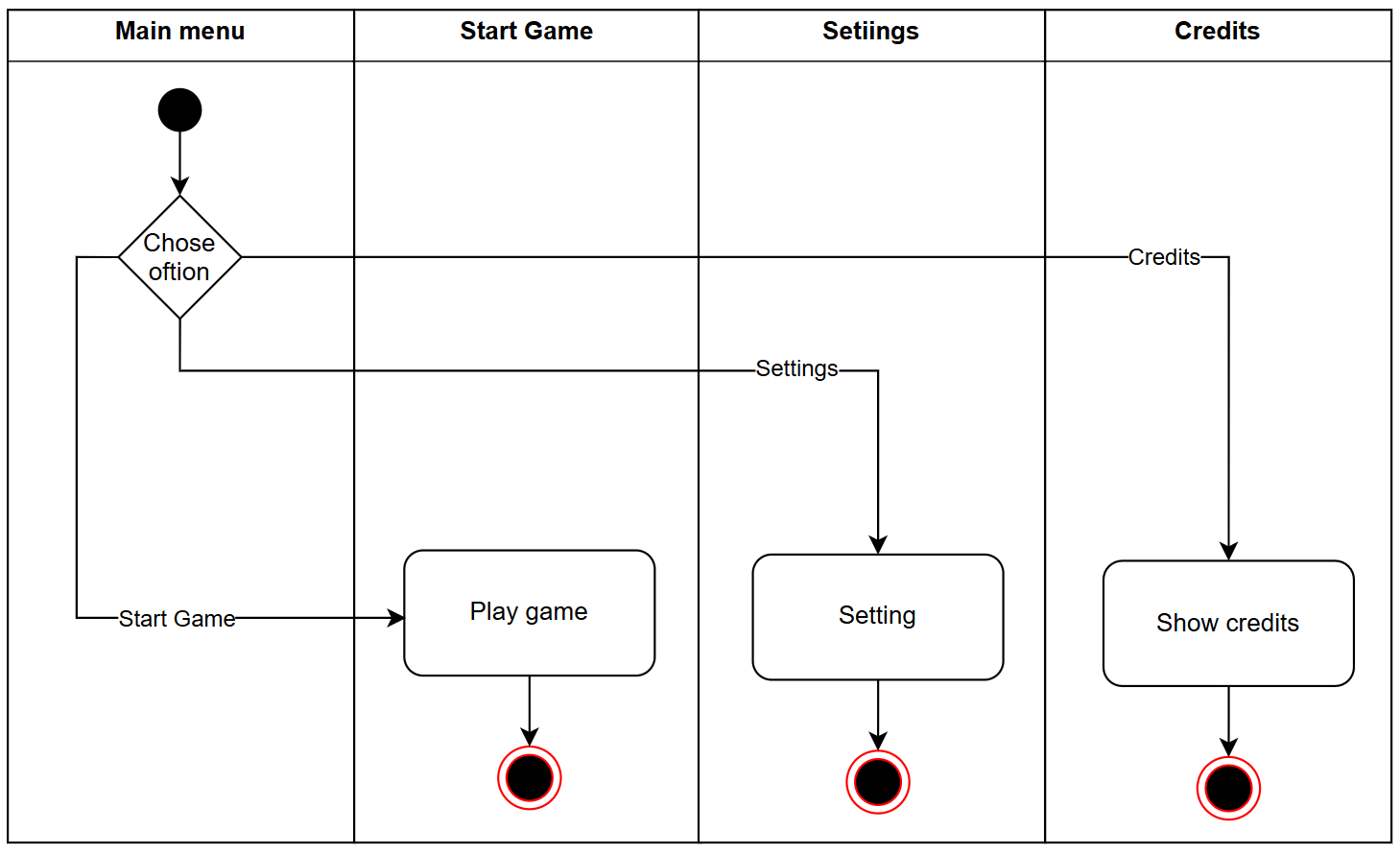
### Sơ đồ tuần tự Achivements:



*Hình 2.5.8 Sơ đồ tuần tự Achivements*

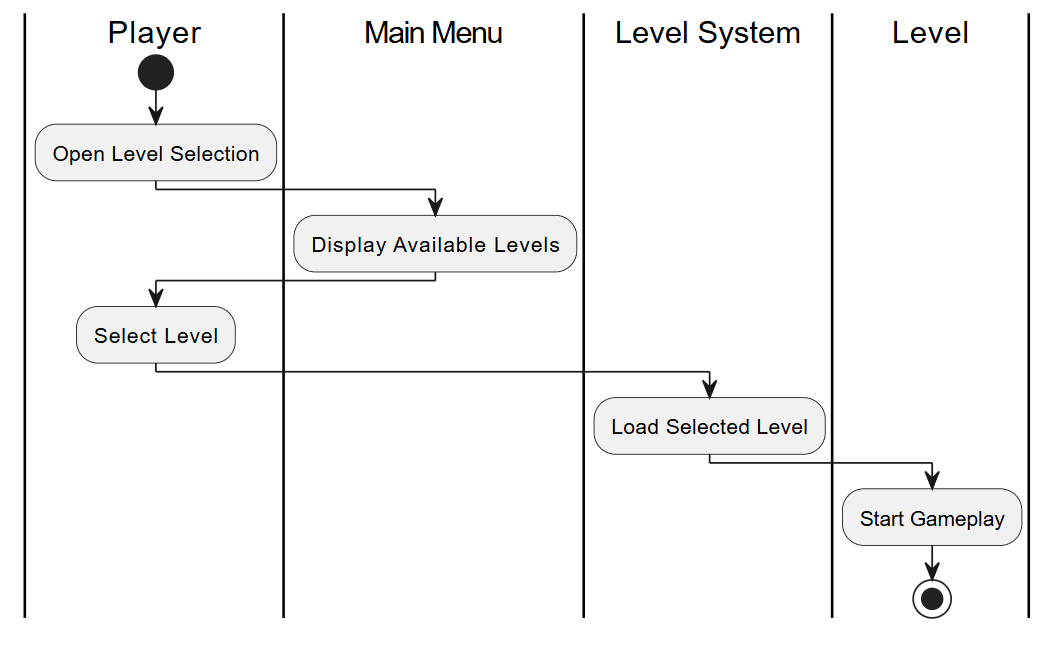
## **SƠ ĐỒ HOẠT ĐỘNG (ACTIVITY DIAGRAM)**

### Sơ đồ hoạt động đăng nhập



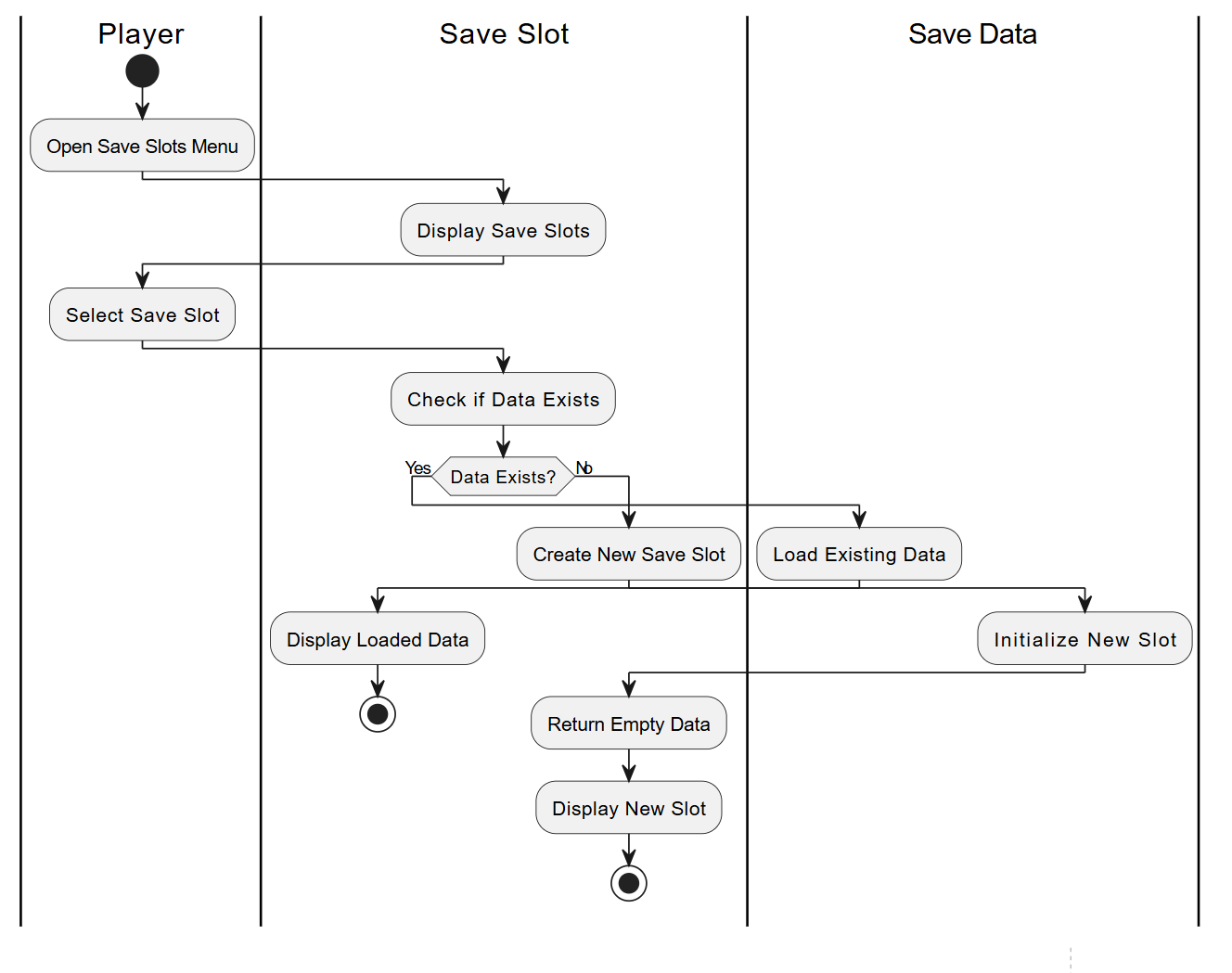
*Hình 2.6.1 Sơ đồ hoạt động main menu*

### Sơ đồ hoạt động Select Level



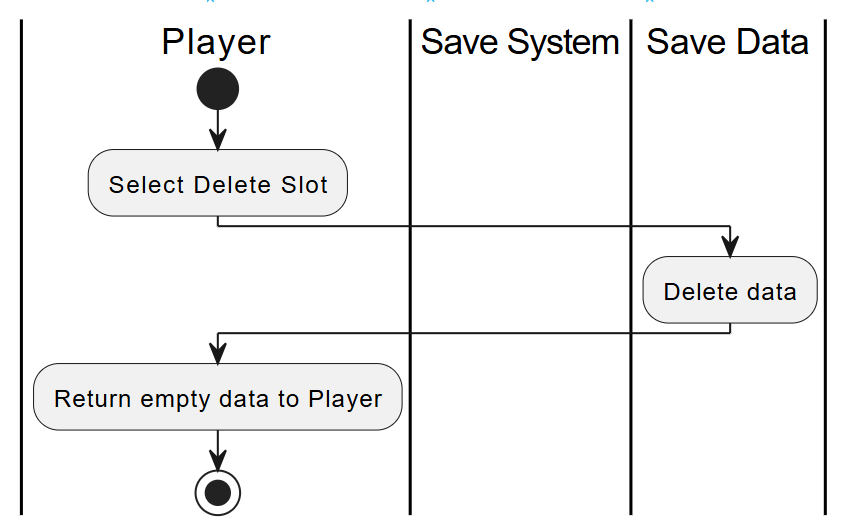
*Hình 2.6.2 Sơ đồ hoạt động Select Level*

### Sơ đồ hoạt động Save Slots



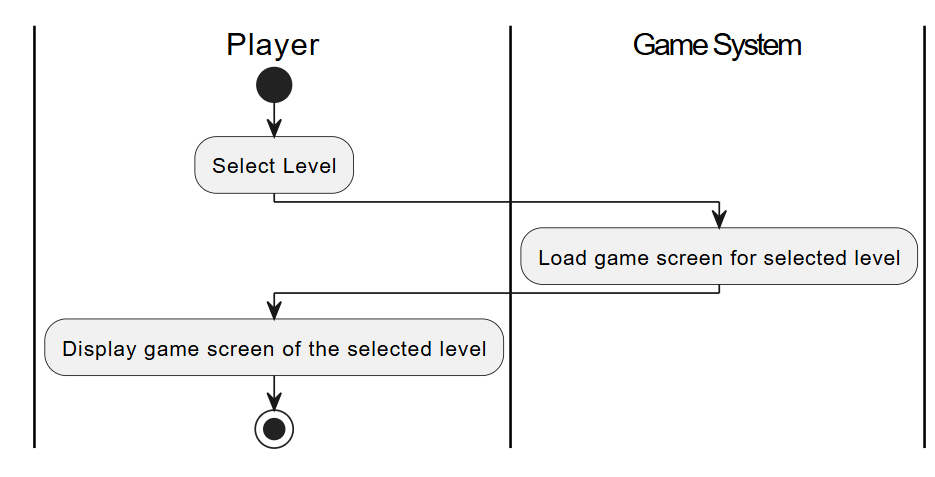
*Hình 2.6.3 Sơ đồ hoạt động Save slots*

### Sơ đồ hoạt động Delete Slots



*Hình 2.6.4 Sơ đồ hoạt động Delete Slots*

### Sơ đồ hoạt động Select Level

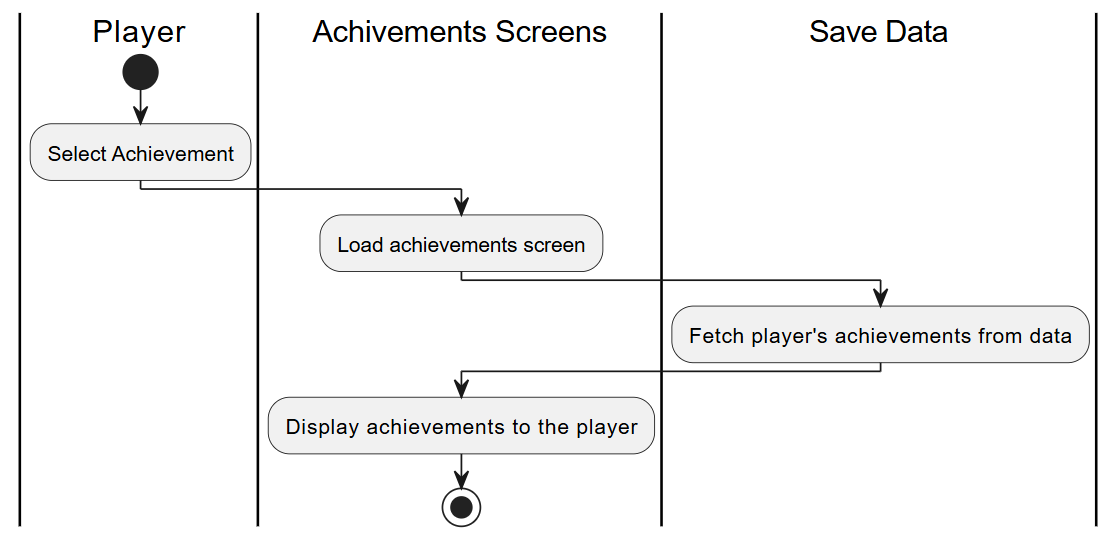
 *Hình 2.6.5 Sơ đồ hoạt động Select Level*

### Sơ đồ hoạt động Buy Tower

### Sơ đồ hoạt động Upgrade Tower

### Sơ đồ hoạt động Sell Tower

### Sơ đồ hoạt động Achivements



*Hình 2.6.7 Sơ đồ hoạt động Achivements*

## **SƠ ĐỒ LỚP (CLASS DIAGRAM)**

## **THIẾT KẾ CƠ SỞ DỮ LIỆU**

### Mô hình quan hệ

### Mô hình vật lý

# 

# CHƯƠNG III

# CÀI ĐẶT VÀ KIỂM THỬ

## **KẾT QUẢ THỰC HIỆN**

1. **Trang chủ**
2. **Danh sách bài viết**
3. **Chi tiết bài viết**
4. **Đăng ký**
5. **Đăng nhập**
6. **Danh sách bài thi**
7. **Thi trắc nghiệm**
8. **Xem kết quả thi**
9. **Lịch sử thi**
10. **Quản lý người dùng**
11. **Thêm, cập nhật tài khoản**
12. **Xóa tài khoản**
13. **Quản lý bài viết**
14. **Thêm, cập nhật bài viết**
15. **Xóa bài viết**
16. **Quản lý bài thi**
17. **Thêm, cập nhật bài thi**
18. **Xóa bài thi**
19. **Chi tiết bài thi**
20. **Thêm, cập nhật câu hỏi**
21. **Xóa câu hỏi**
22. **Cập nhật thông tin người dùng**
23. **Cập nhật mật khẩu**

# KẾT LUẬN

Qua quá trình thực hiện khóa luận này, đã đạt được những kết quả sau:

Ưu điểm:

**Điểm hạn chế:**

**Hướng phát triển**

# TÀI LIỆU THAM KHẢO

**Tài liệu tiếng Việt**

**Tài liệu tiếng Anh**

**Tài liệu Internet**