

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN CÔNG NGHỆ PHẦN MỀM**

**\*\*\*\*\***

Icon

Description automatically generated

**ĐỒ ÁN MÔN HỌC**

**THIẾT KẾ KIẾN TRÚC PHẦN MỀM**

**Đề tài**

Ứng dụng: FLAPPY BIRD

Nhóm sinh viên thực hiện: Nhóm 40

Danh sách sinh viên:

**Phạm Danh Hiển 0188966**

**Nguyễn Thạc Trung Hiếu 0189866**

Lớp học phần: **66PM1**

Giảng viên hướng dẫn:Ths. **Phạm Hữu Tùng**

Hà Nội 4/2025

**MỤC LỤC**

[**1. Giới thiệu đề tài** 2](#_Toc199195556)

[**2. Phân tích hệ thống** 5](#_Toc199195557)

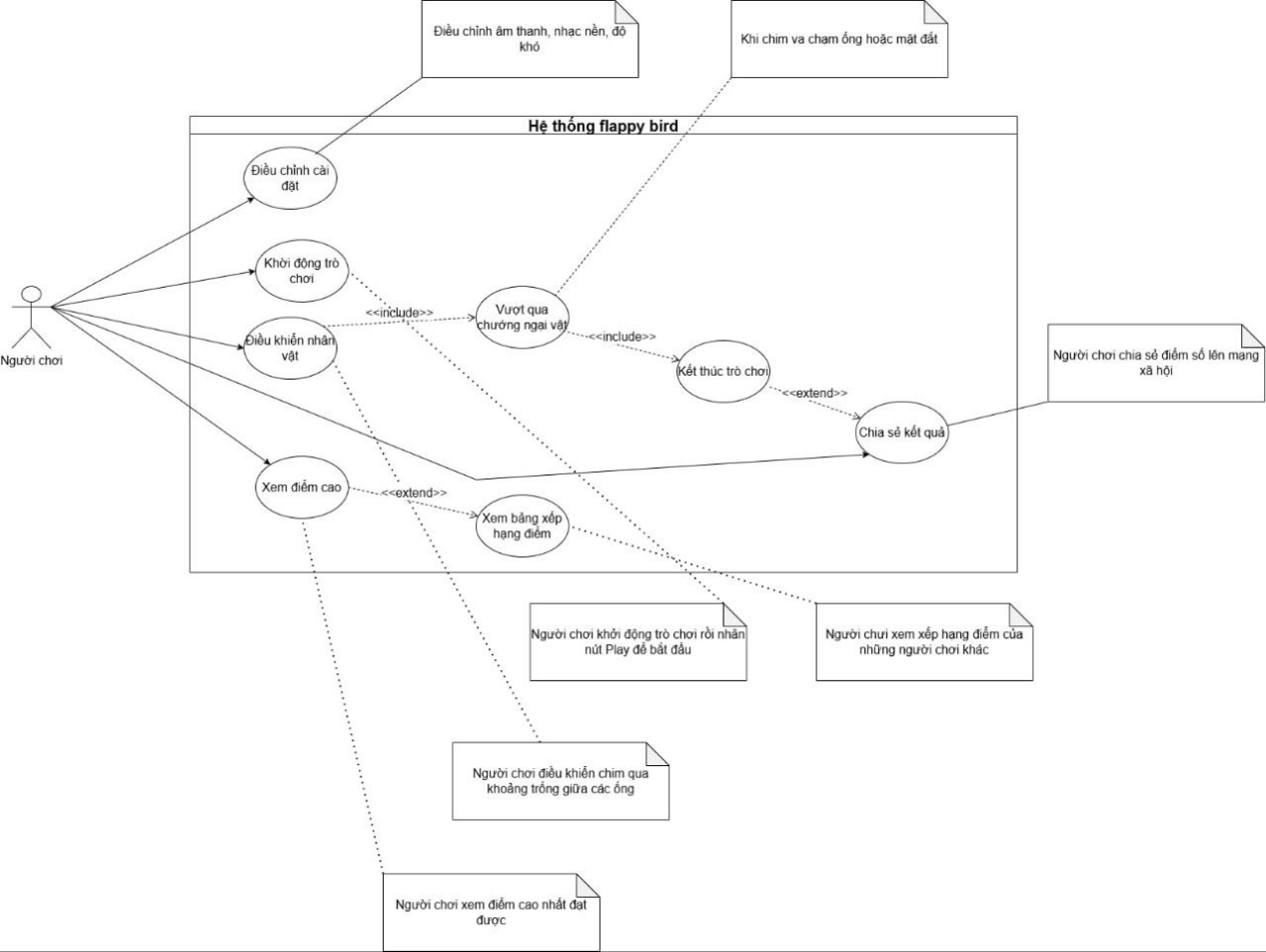
[**3. Thiết kế hệ thống** 11](#_Toc199195558)

[**4. Cài đặt và triển khai hệ thống** 13](#_Toc199195559)

# **Giới thiệu đề tài**

* 1. **Lý do chọn đề tài**
* Trong những năm gần đây, xu hướng giải trí trên thiết bị di động đã bùng nổ mạnh mẽ. Giữa hàng ngàn tựa game có đồ họa phức tạp, gameplay cầu kỳ, thì những trò chơi đơn giản, dễ tiếp cận nhưng đầy thử thách lại bất ngờ chiếm trọn cảm tình của người dùng. Flappy Bird chính là một ví dụ điển hình: một trò chơi với cơ chế chỉ cần một chạm nhưng đã từng gây sốt toàn cầu, thu hút hàng triệu lượt tải và người chơi mỗi ngày.
* Tuy nhiên, sau khi bị gỡ khỏi các kho ứng dụng, Flappy Bird để lại một khoảng trống đáng kể cho những ai yêu thích thể loại game casual này. Rất nhiều người chơi – từ học sinh, sinh viên đến người đi làm – mong muốn được trải nghiệm lại cảm giác “vừa chơi vừa tức” nhưng không thể dứt ra. Điều này mở ra cơ hội để phát triển lại một phiên bản Flappy Bird mới, hiện đại hơn, có thể mở rộng tính năng, và mang tính cá nhân hóa cao hơn.
  1. **Mục tiêu chọn đề tài**
* Một ứng dụng trò chơi Flappy Bird được làm lại không chỉ đơn thuần là một trò chơi giải trí mà còn có thể phục vụ nhiều mục đích khác như: thử nghiệm công nghệ mới, khai thác quảng cáo tạo doanh thu, xây dựng cộng đồng người chơi hoặc thậm chí là nền tảng để học lập trình game cơ bản.
* Nếu trò chơi có nhiều lượt người dùng có thể cân nhắc thu phí
* Ngôn ngữ lập trình: Godot C#
* Mô hình : MVVM + DDD
  1. **Phạm vi và giới hạn**
     1. **Phạm vi**
* Ứng dụng mô phỏng trò chơi Flappy Bird đơn giản dưới dạng 2D.
* Cung cấp các tính năng cơ bản:
* Khởi động trò chơi.
* Nhấn phím để điều khiển chim bay.
* Tính điểm dựa trên số ống vượt qua.
* Hiển thị điểm cao nhất.
* Được phát triển trên nền tảng GODOT
* Sử dụng đồ họa đơn giản, âm thanh cơ bản, hướng đến mục tiêu học thuật và giải trí nhẹ nhàng.
  + 1. **Giới hạn**
* Không hỗ trợ chơi đa người hoặc chơi online.
* Không có các tính năng nâng cao như lưu trạng thái, bảng xếp hạng online hay tuỳ biến nhân vật.
* Chỉ được thiết kế cho thiết bị di động.
* Thiết kế và hiệu ứng chưa phong phú như bản thương mại.
  1. **Phương pháp thực hiện**
* Quá trình xây dựng ứng dụng Flappy Bird được chia thành các bước chính:
  + 1. **Bước 1: Phân tích yêu cầu**
* Xác định chức năng cần có: chuyển động nhân vật, tạo ống, tính điểm, va chạm.
* Thiết kế giao diện cơ bản của trò chơi.
  + 1. **Bước 2: Thiết kế game**
* Xây dựng sơ đồ use case, sequence và activity diagram.
* Thiết kế tài nguyên hình ảnh: chim, ống nước, nền, v.v.
* Quy định các tham số như tốc độ rơi, độ cao nhảy, khoảng cách giữa các ống.
  + 1. **Bước 3: Lập trình**
* Cài đặt các đối tượng (class) như BirdEntity, PipeEntity, GameLevel.
* Xử lý vòng lặp trò chơi (Game loop): cập nhật chuyển động, kiểm tra va chạm, render khung hình.
* Cập nhật điểm và reset trò chơi khi kết thúc.
  + 1. **Bước 4: Kiểm thử và tinh chỉnh**
* Kiểm tra va chạm và logic tính điểm.
* Tinh chỉnh độ khó bằng việc thay đổi tốc độ hoặc vị trí sinh ống.
* Xử lý lỗi, tối ưu hóa hiệu suất.
  + 1. **Bước 5: Hoàn thiện**
* Thêm âm thanh (nếu có).
* Đóng gói và triển khai ứng dụng.

# **Phân tích hệ thống**

* 1. **Mô tả bài toán**
* Bài toán đặt ra là xây dựng một ứng dụng trò chơi mô phỏng Flappy Bird – một trò chơi điện tử dạng arcade 2D đơn giản nhưng gây nghiện, trong đó người chơi điều khiển một chú chim bay qua các chướng ngại vật (ống nước) mà không được va chạm vào chúng.
* Mục tiêu của trò chơi là giúp chú chim bay càng xa càng tốt bằng cách nhấn phím (hoặc chạm màn hình) để giữ nó bay lên trong khi trọng lực kéo nó rơi xuống. Mỗi lần chim vượt qua một cặp ống nước thành công, người chơi sẽ nhận được một điểm.
* Yêu cầu đặt ra:
  + Chim có thể bay lên mỗi khi người chơi nhấn phím.
  + Chim bị ảnh hưởng bởi trọng lực và rơi dần nếu không được điều khiển.
  + Các ống nước sinh ra liên tục, có khoảng cách ngẫu nhiên, và di chuyển từ phải qua trái.
  + Va chạm giữa chim và ống hoặc rơi xuống đất sẽ kết thúc trò chơi.
  + Giao diện hiển thị điểm số hiện tại và điểm cao nhất (high score).
  + Sau khi thua, người chơi có thể chơi lại từ đầu.
  1. **Yêu cầu chức năng**
     1. **Chức năng khởi động trò chơi**
* Mô tả: Khi mở ứng dụng, người chơi nhấn vào bất kỳ vị trí nào trên màn hình để bắt đầu.
* Luồng xử lý:
* Tải tài nguyên game (hình ảnh, âm thanh).
* Hiển thị màn hình khởi động với logo và hướng dẫn.
* Khi người chơi nhấn → chuyển sang chế độ chơi.
* Vai trò: Tạo ấn tượng ban đầu và là điểm khởi đầu của gameplay.
  + 1. **Chức năng điều khiển chim bay**
* Mô tả: Người chơi chạm vào màn hình để làm chim bay lên; khi không chạm, chim rơi tự do vì trọng lực.
* Luồng xử lý:
* Nhận sự kiện chạm màn hình.
* Thay đổi vận tốc theo trục Y của chim (hướng lên).
* Áp dụng trọng lực theo thời gian để kéo chim xuống.
* Vai trò: Đây là chức năng cốt lõi giúp người chơi tương tác với game.
  + 1. **Chức năng tạo ống nước (Pipe Generation)**
* Mô tả: Các cặp ống nước xuất hiện liên tục từ bên phải màn hình, với độ cao thay đổi. Luồng xử lý:
* Định kỳ tạo ra cặp ống với khoảng trống ngẫu nhiên.
* Di chuyển từ phải sang trái.
* Xóa ống cũ khi ra khỏi màn hình.
* Vai trò: Tạo chướng ngại vật, tăng độ khó và thử thách cho người chơi. 4. Chức năng xử lý va chạm
  + 1. **Chức năng quản lý trạng thái chim bay**
* Mô tả: Khi chim va vào ống nước hoặc mặt đất → kết thúc trò chơi.
* Luồng xử lý:
* So sánh vị trí của chim với các ống nước và mặt đất.
* Nếu xảy ra va chạm → hiển thị màn hình kết thúc (Game Over).
* Vai trò: Kiểm soát điều kiện kết thúc game, tạo động lực cải thiện kỹ năng. 5. Chức năng tính điểm
* Mô tả: Mỗi khi chim bay qua một cặp ống thành công, người chơi được cộng 1 điểm.
* Luồng xử lý:
* Kiểm tra nếu chim vượt qua ống mà chưa va chạm.
* Tăng biến đếm điểm.
* Cập nhật điểm số hiện tại và điểm cao nhất.
* Vai trò: Khuyến khích người chơi cải thiện, tạo yếu tố thi đua. 6. Chức năng hiển thị điểm và Game Over
  + 1. **Chức năng tổng kết**
* Mô tả: Khi trò chơi kết thúc, màn hình sẽ hiển thị điểm hiện tại và điểm cao nhất.
* Luồng xử lý:
* Gọi giao diện “Game Over”.
* Hiển thị điểm.
* Cung cấp nút "Chơi lại" hoặc "Thoát".
* Vai trò: Thông báo kết quả, thúc đẩy người chơi tiếp tục thử lại. 7. Chức năng chơi lại (Restart)
* Mô tả: Sau khi Game Over, người chơi có thể nhấn nút để bắt đầu lại trò chơi.
* Luồng xử lý:
* Reset tất cả vị trí đối tượng (chim, ống).
* Reset điểm số về 0.
* Quay lại vòng chơi mới.
* Vai trò: Duy trì tính lặp lại và gây nghiện của trò chơi. 8. Chức năng âm thanh
  1. **Yêu cầu phi chức năng**
* Giao diện thân thiện, dễ sử dụng với người dùng
* Ổn định, xử lí hiệu quả
* Khả năng lưu trữ dữ liệu và phục hồi khi gặp sự cố
* Đủ khả năng tiếp nhận số lượng truy cập lớn với định mức mong muốn
  1. **USE CASE**
     1. **USE CASE hệ thống**
     2. **USE CASE chính**
* Bắt đầu trò chơi

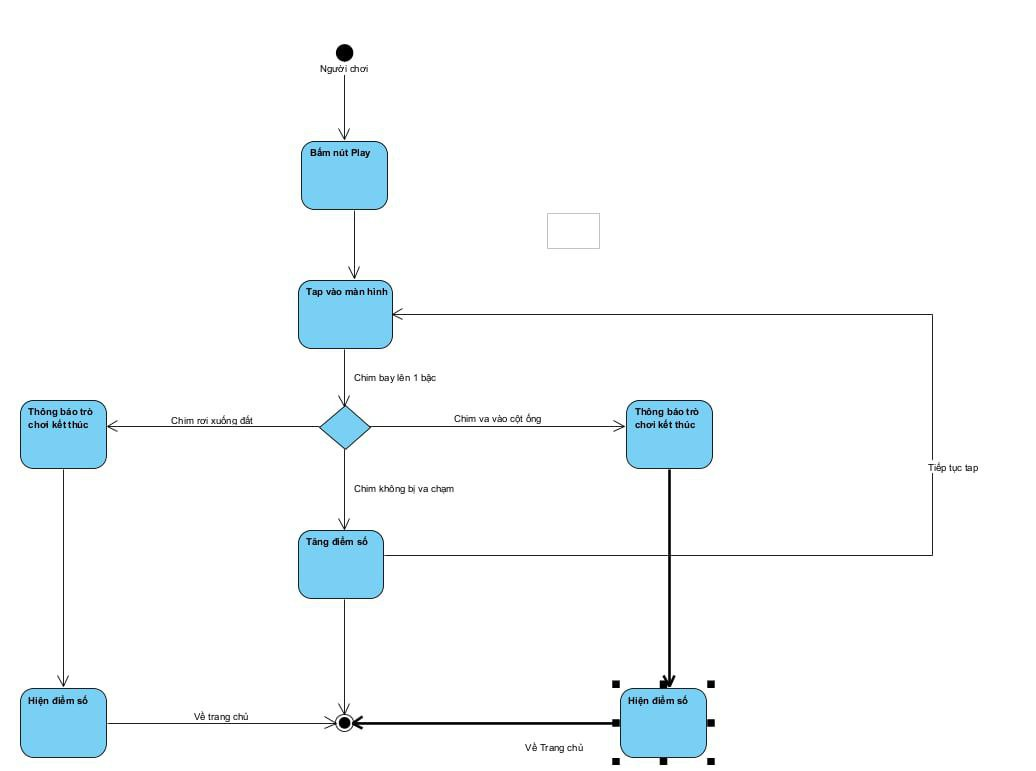
|  |  |
| --- | --- |
| **ID** | 0 |
| **Name** | Bắt đầu trò chơi |
| **Brief description** | Cho phép người dùng bắt đầu trò chơi |
| **Actors** | Người dùng |
| **Pre-conditions** |  |
| **Basic flow** | 1. Người dùng khởi động trò chơi 2. Hệ thống hiện giao diện trò chơi 3. Người dùng bấm phím space 4. Hệ thống chuyển sang màn hình chơi |
| **Alternative flow** |  |
| **Result** | Người dùng khởi động được trò chơi |
| **Exceptions** |  |

* Chơi trò chơi

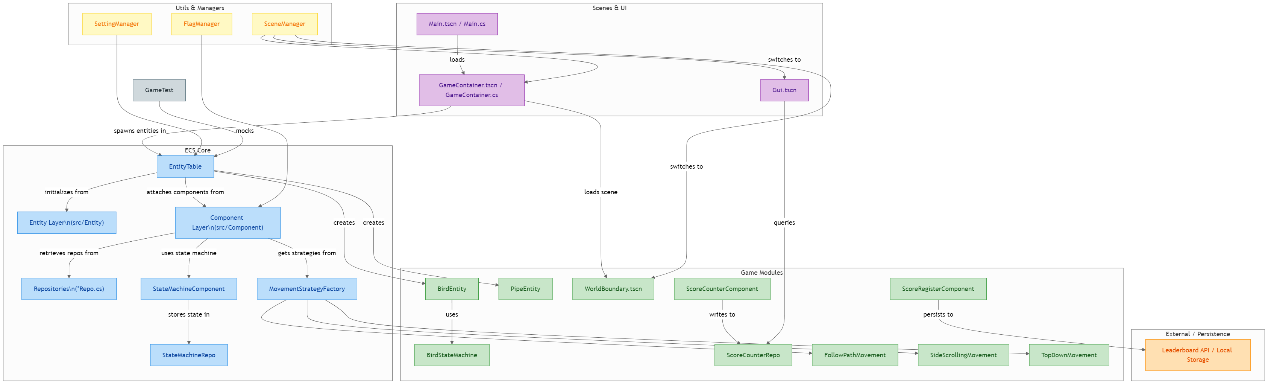
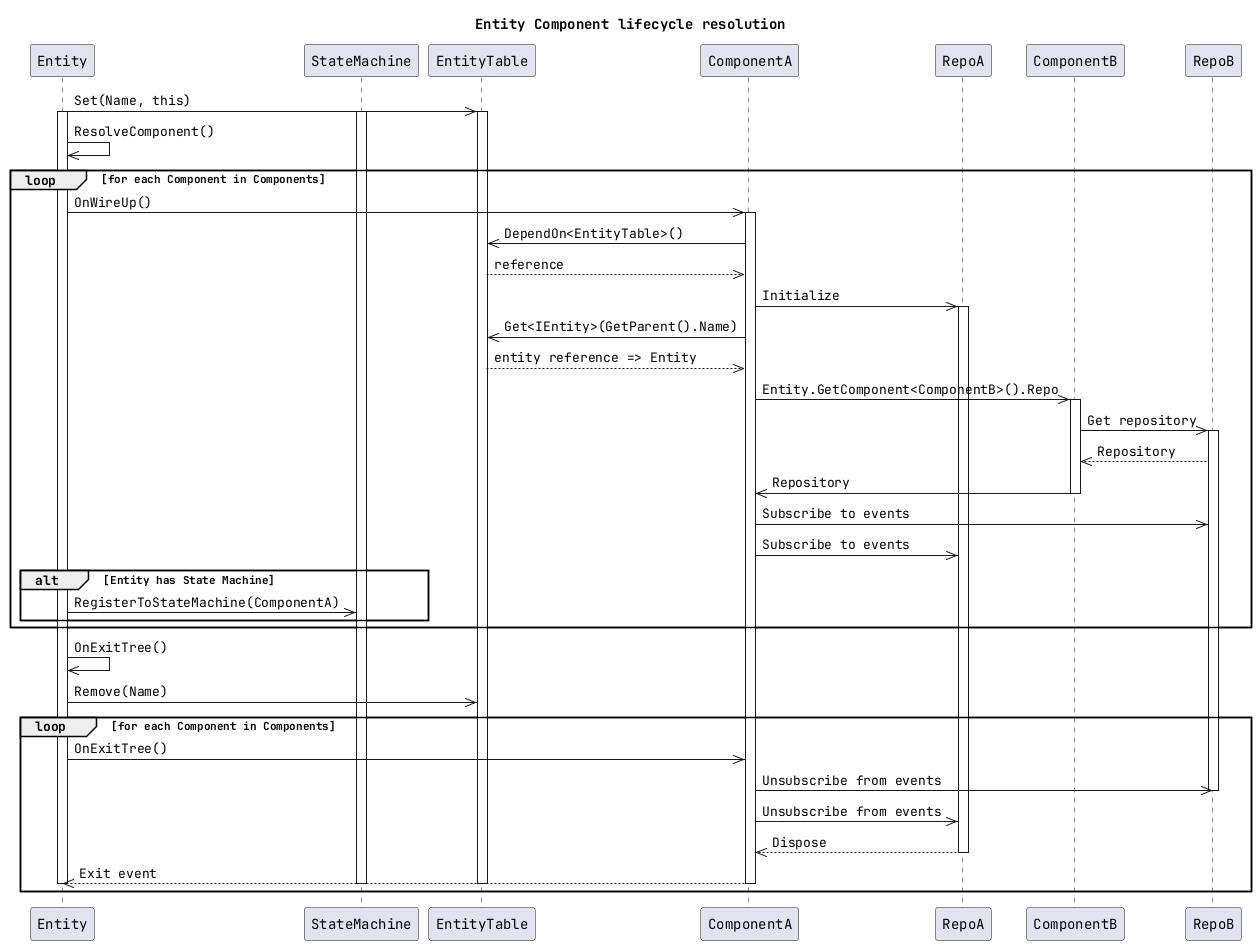
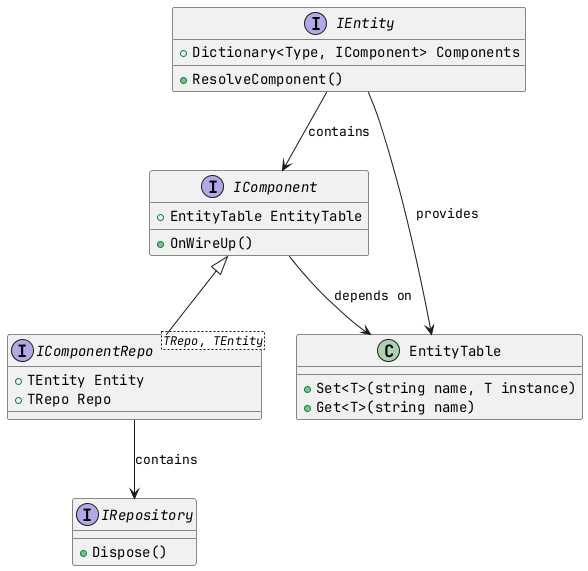
|  |  |
| --- | --- |
| **ID** | 1 |
| **Name** | Chơi trò chơi |
| **Brief description** | Cho phép người dùng điều khiển con chim vượt qua chướng ngại vật |
| **Actors** | Người dùng |
| **Pre-conditions** |  |
| **Basic flow** | 1. Người dùng nhấn vào màn hình để khiến chim bay lên 2. Khi người dùng không nhấn, con chim sẽ rơi xuống 3. Khi người dùng vượt qua 1 chướng ngại vật, hệ thống sẽ cộng thêm 1 điểm |
| **Alternative flow** | Khi người dùng va phải 1 chướng ngại vật, trò chơi sẽ kết thúc |
| **Result** | Người dùng chơi trò chơi |
| **Exceptions** |  |

* Kết thúc trò chơi

|  |  |
| --- | --- |
| **ID** | 2 |
| **Name** | Kết thúc trò chơi |
| **Brief description** | Hiển thị điểm số cho người chơi |
| **Actors** | Người dùng |
| **Pre-conditions** |  |
| **Basic flow** | 1. Hệ thống hiển thị điểm số người chơi vừa đạt được 2. Cho phép người chơi nhấn phím ESC để bắt đầu lại |
| **Alternative flow** |  |
| **Result** | Điểm số được hiển thị |
| **Exceptions** |  |

* 1. **Biểu đồ hoạt động**

# **Thiết kế hệ thống**

* 1. **Lựa chọn kiến trúc**
* Kết hợp giữa MVVM và ECS, tạo nên độ linh hoạt khi phát triển và mở rộng
* Sử dụng các Component như các khối xây dựng, để thêm các chức năng / thuộc tính cho đối tượng
* Liên kết giữa các đối tượng bằng Dependency Injection
  1. **Sơ đồ kiến trúc tổng thể**
  2. **Vòng đời Component**
  3. **Class diagram**

# **Cài đặt và triển khai hệ thống**

* 1. **Môi trường triển khai**
* Dự án có thể được phát triển ở cả Windows, MacOs và Linux
* Sử dụng Game Engine Godot làm nền tảng
  1. **Cài đặt hệ thống**
* Cài đặt [git](https://git-scm.com/) và [git-lfs](https://git-lfs.com/)
* Cài đặt [.Net framework](https://learn.microsoft.com/vi-vn/dotnet)
* Thiết lập git-lfs git-lfs install
* Cài đặt [GodotEnv](https://github.com/chickensoft-games/GodotEnv) và dùng GodotEnv để cài Godot theo phiên bản của dự án
* Clone dự án tại https://github.com/fm39hz/FlappyBirdGame.git
* Build dự án