

# BÁO CÁO ĐỒ ÁN LẬP TRÌNH C#

## XẾP HÌNH CỔ ĐIỂN - Tetris

Giảng viên hướng dẫn:

TS. Bùi Tiến Lên

Thành viên nhóm 2:

Lê Thị Mỹ Hương 3122411077

Đỗ Minh Quân 3122411166

Trần Bùi Ty Ty 3122411241





# NỘI DUNG

01

Giới thiệu đề tài

02

Phân tích yêu cầu

03

Thiết kế lớp và biểu đồ lớp

04

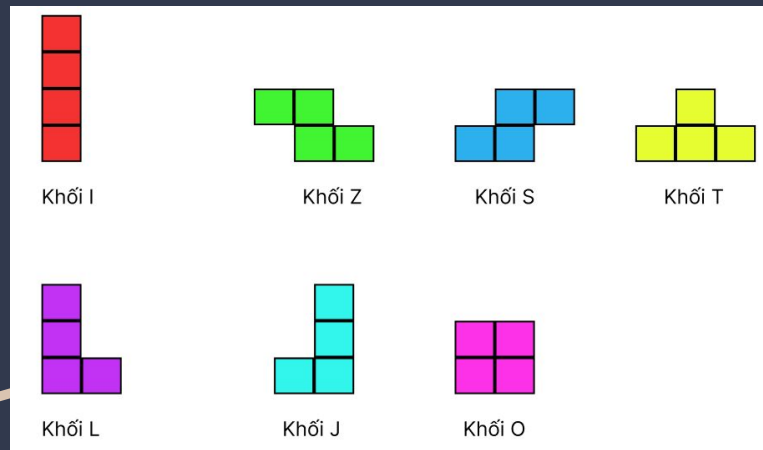
Giải thích giải pháp

05

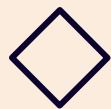
Đánh giá và hướng phát triển

# Giới thiệu đề tài

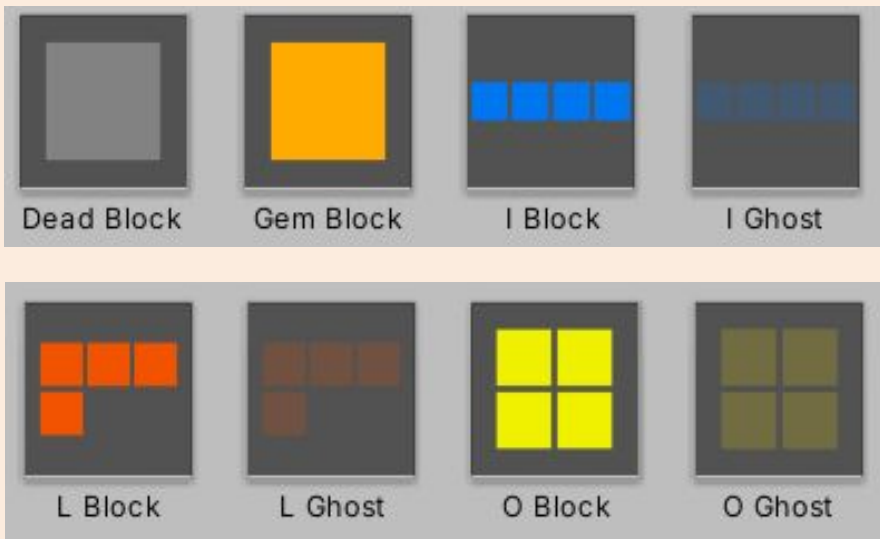
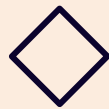
## *~Trò chơi xếp hình cổ điển - Tetris~*



- Tetris là trò chơi xếp hình kinh điển, nơi người chơi điều khiển các khối Tetriminos để lấp đầy hàng ngang và ghi điểm.
- Lối chơi đơn giản nhưng thách thức, giúp rèn luyện tư duy logic, phản xạ nhanh và kỹ năng xử lý tình huống thông qua độ khó của game
- Gồm 7 hình dạng đặc trưng: I, O, T, L, J, S, Z, mỗi khối mang một chiến thuật riêng và góp phần tạo nên sự cuốn hút.



# THUẬT NGỮ



**Khối (Block):** Các mảnh ghép với 7 dạng hình học chính (Tetriminos).

**GemBlock:** Khối đặc biệt trong chế độ “Màn chơi” (Khối màu cam).

**GhostBlock:** Hiển thị vị trí rơi của khối.

**DeadBlock:** Khối không thể di chuyển.

**Chế độ chơi:** Chơi Theo Màn và Chơi Vô Hạn.

**Xuống Nhanh/Xuống Lập Tức:** Điều khiển tốc độ rơi của khối.

**Xóa Dòng:** Xóa hàng khi lấp đầy.

**Ghi Điểm:** Điểm được ghi nhận dựa trên số dòng xóa được.

**Xoay Khối/Di chuyển Khối/ Khối Kế Tiếp:** di chuyển khối/xem trước các khối.

# Phân tích yêu cầu

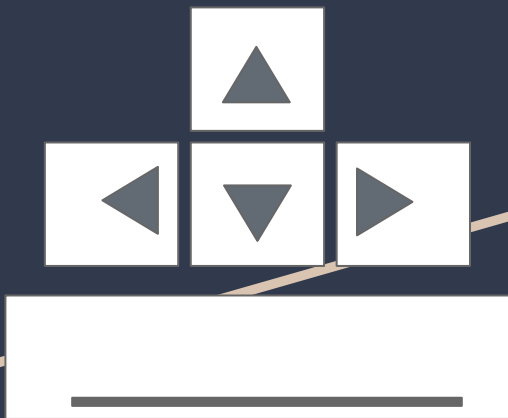
*~Mô tả tổng quan trò chơi~*



- Bàn chơi: Lưới 10x20, nơi các khối Tetrimino rơi từ đỉnh xuống đáy.
- Điều khiển Tetrimino:
  - Xoay khối: Xoay 90° quanh điểm xoay.
  - Di chuyển khối: Trái/phải để tránh va chạm.
  - Hạ nhanh:
    - Soft Drop: Tăng tốc độ rơi.
    - Hard Drop: Đưa khối xuống đáy ngay lập tức.
- Đóng băng khối: Khối cố định khi chạm đáy hoặc va khối khác.

# Phân tích yêu cầu

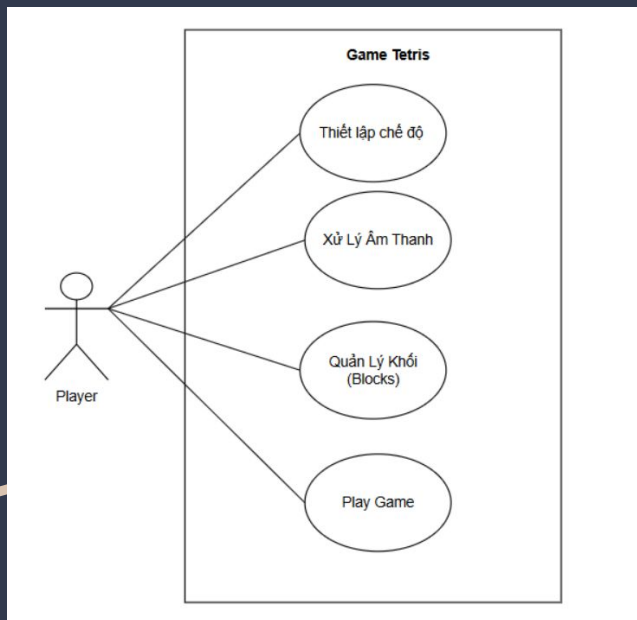
*~Mô tả tổng quan trò chơi~*



- Mục tiêu:
  - Vô hạn: Ghi điểm cao nhất có thể
  - Màn chơi: Xóa GemBlocks, vượt 10 màn để hoàn thành trò chơi.

# Phân tích yêu cầu

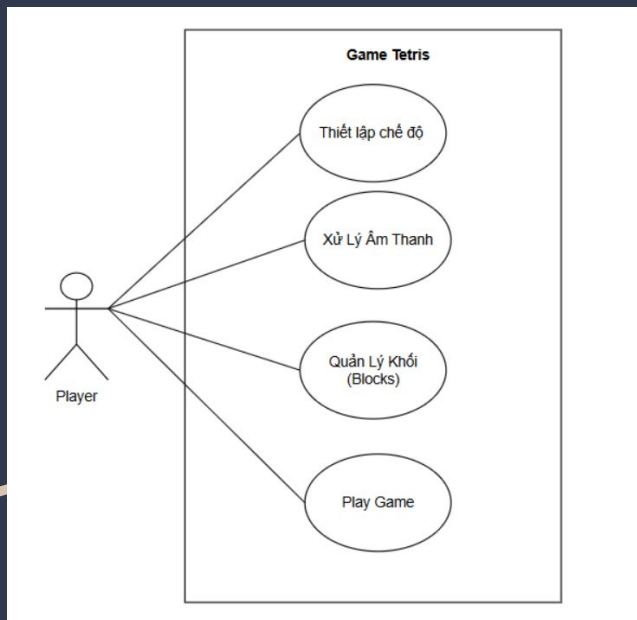
*~Mô tả hệ thống~*



STT	Tên Chức Năng	Mô Tả
UC01	Thiết lập chế độ chơi	Cho phép người chơi thiết lập chế độ chơi bao gồm chế độ "vô hạn" hoặc "theo màn" và xem hướng dẫn.
UC02	Xử lý âm thanh	Hệ thống tự động phát âm thanh nền khi trò chơi bắt đầu, dừng âm thanh khi kết thúc và chuyển âm thanh khi thay đổi màn chơi.

# Phân tích yêu cầu

*~Mô tả hệ thống~*



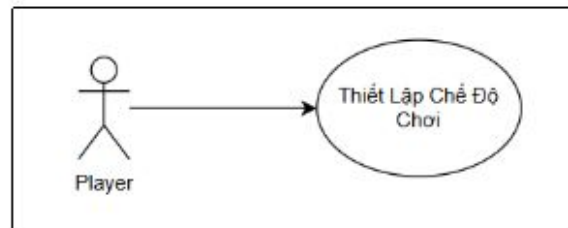
<b>UC03</b>	Quản lý khối (Blocks)	Người chơi thao tác để điều khiển khối rơi bằng các phím: tăng tốc (xuống), rơi thẳng (cách), xoay 90° (lên), và di chuyển ngang (trái/phải).
<b>UC04</b>	Play Game	Quản lý toàn bộ quá trình chơi game: bắt đầu, tạm dừng, xử lý khối rơi, kiểm tra trạng thái thắng/thua, và lựa chọn chơi lại hoặc thoát.



# Phân tích yêu cầu

~Mô tả hệ thống~

-UC01-



Use Case Number:	UC01	
Use Case Name:	Thiết lập chế độ chơi	
Actor (s):	Player	
Maturity:	Focused	
Summary:	Thiết lập chế độ chơi hoặc hiển thị hướng dẫn trò chơi theo nhu cầu, sở thích của Player. Player có thể chọn chế độ chơi “vô hạn”, hoặc “theo màn”	
Basic Course of Events:	Actor Action	System Response
	1. Actor truy cập vào game.	
		2. Hệ thống hiển thị trang chủ game.
	3. Actor <u>chọn chế độ</u> . A1	
		4. Hệ thống thiết lập chế độ chơi.

# Phân tích yêu cầu

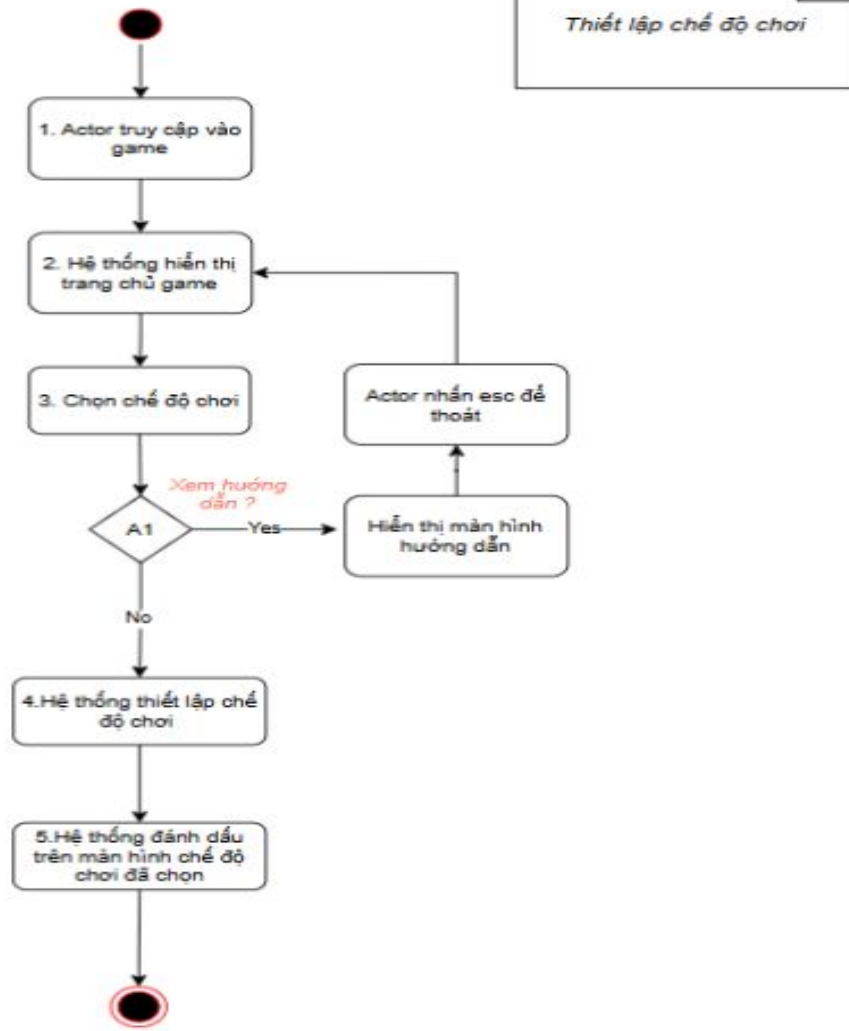
~Mô tả hệ thống~

-UC01-

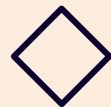
		5. Hệ thống đánh dấu trên màn hình chế độ chơi đã được chọn.  Use Case kết thúc tại đây
Alternative Paths:	A1. Xem Hướng Dẫn	
	Actor Action	System Response
	1. Actor <u>chọn</u> “ <u>Hướng Dẫn</u> ”.	
		2. Hệ thống hiển thị màn hình hướng dẫn.
	3. Actor <u>nhấn Esc để thoát</u> . <u>Quay lại bước 2 UC1 trong Basic Course of Events</u>	
Exception Paths:	None	
Extension Points:	None	
Triggers:	None	
Assumptions:	None	
Preconditions:	None	
Post Conditions:	None	
Reference: Business Rules	None	
Author(s):	Nhóm 2	
Date:	16-12-2024	

# Phân tích yêu cầu

## Activity diagram UC01

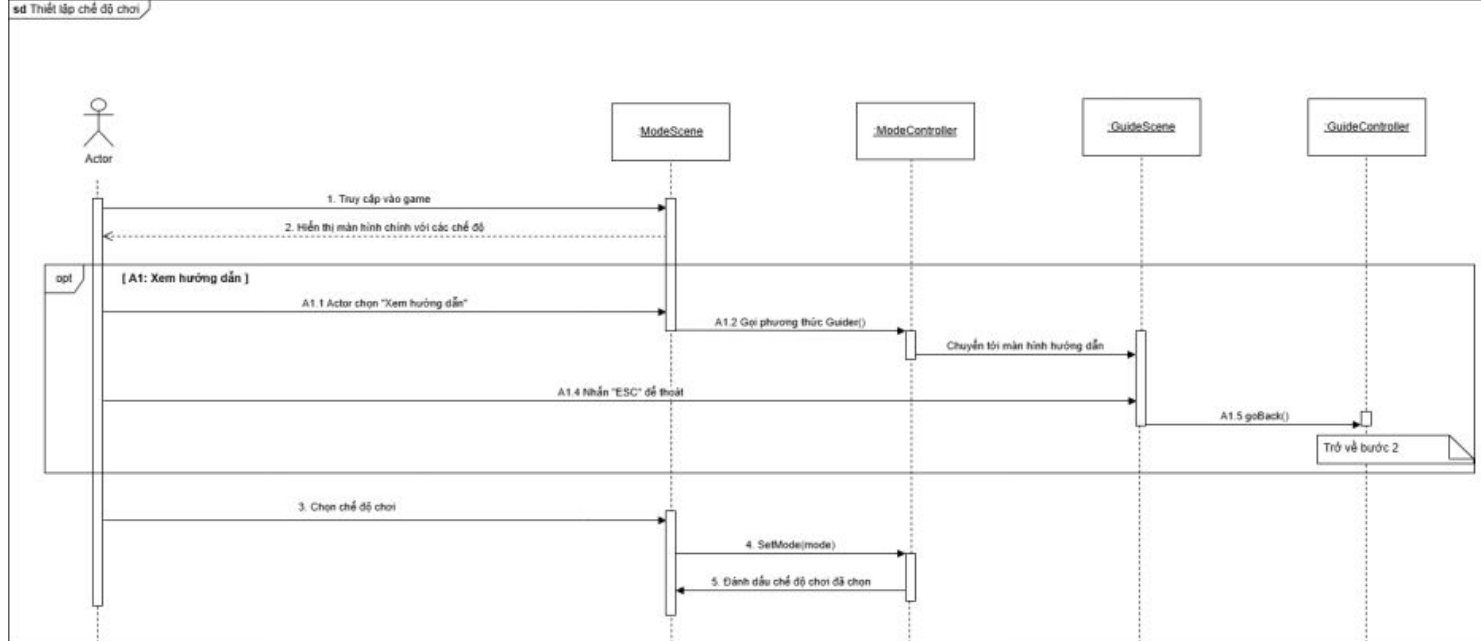


# ◇ Giao diện Thiết Lập Chế Độ



# Phân tích yêu cầu

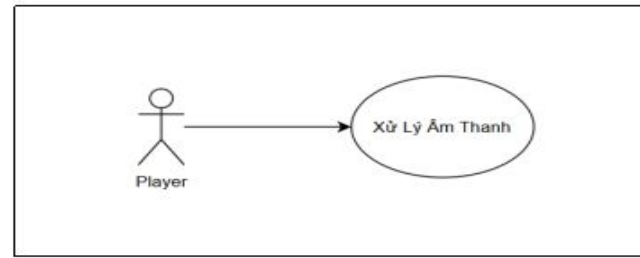
*-UC01-*



# Phân tích yêu cầu

~Mô tả hệ thống~

-UC02-



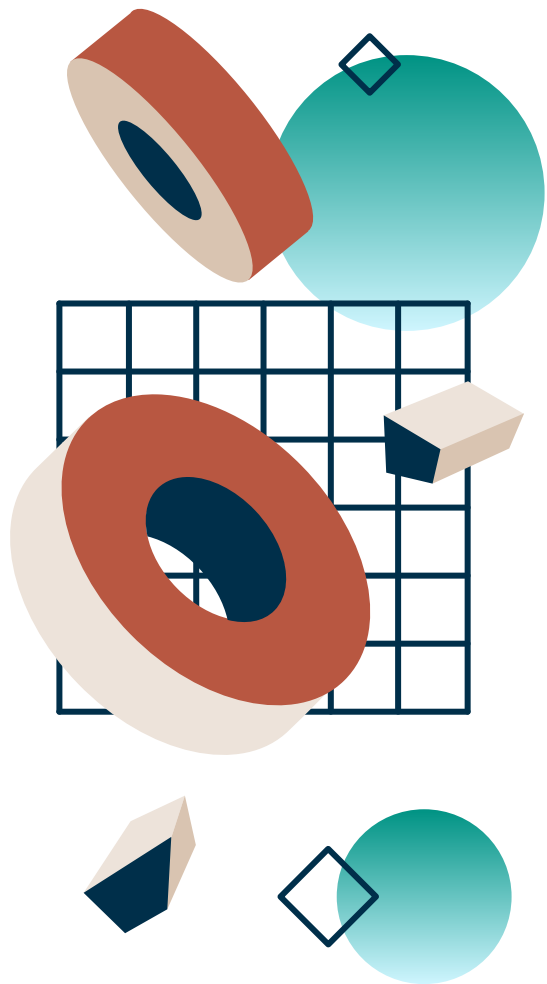
Use Case Number:	UC02	
Use Case Name:	Xử Lý Âm Thanh	
Actor (s):	Player	
Maturity:	Focused	
Summary:	Player thao tác để phát âm thanh nền và các trạng thái tắt/mở âm thanh của hệ thống.	
Basic Course of Events:	Actor Action	System Response
	1. Actor truy cập vào game	
		2. Hệ thống tự động phát âm thanh nền khi trò chơi bắt đầu.
	3. Player thao tác để đặt khối hoặc để khối tự rơi. A1	
		4. Hệ thống kiểm tra điều kiện kết thúc game. A2

# Phân tích yêu cầu

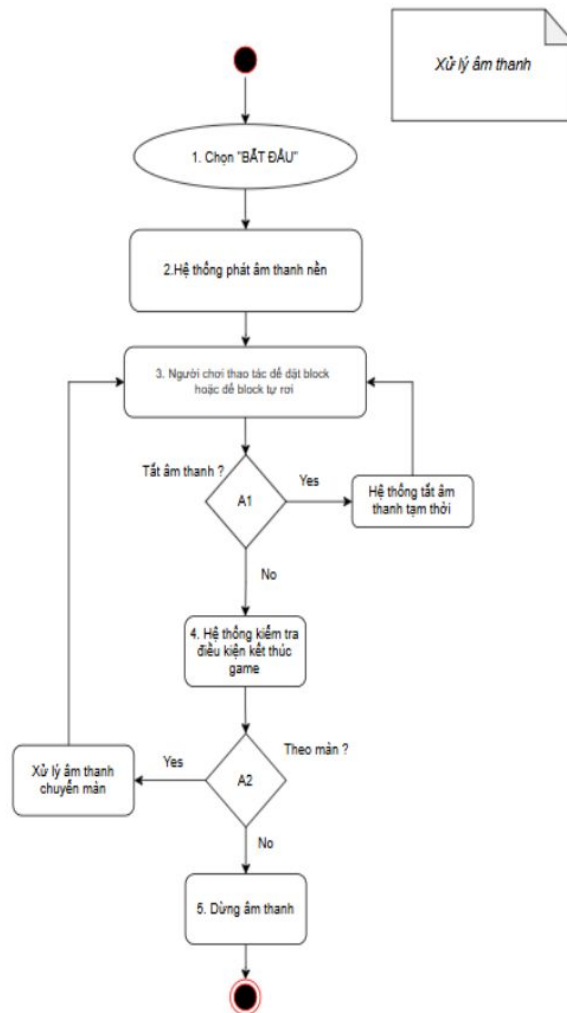
~Mô tả hệ thống~

-UC01-

		5. Hệ thống dừng âm thanh. Use Case kết thúc tại đây
Alternative Paths:	A1. Hệ thống kiểm tra màn chơi hiện tại	
	A2. Xử lý âm thanh chuyển màn.	
Exception Paths:	None	
Extension Points:	None	
Triggers:	Khi Actor bắt đầu trò chơi	
Assumptions:	None	
Preconditions:	None	
Post Conditions:	None	
Reference: Business Rules	None	
Author(s):	Nhóm 2	
Date:	16-12-2024	
Activity Diagram: <u>Xử Lý Âm Thanh</u>		

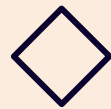


## Activity diagram UC02





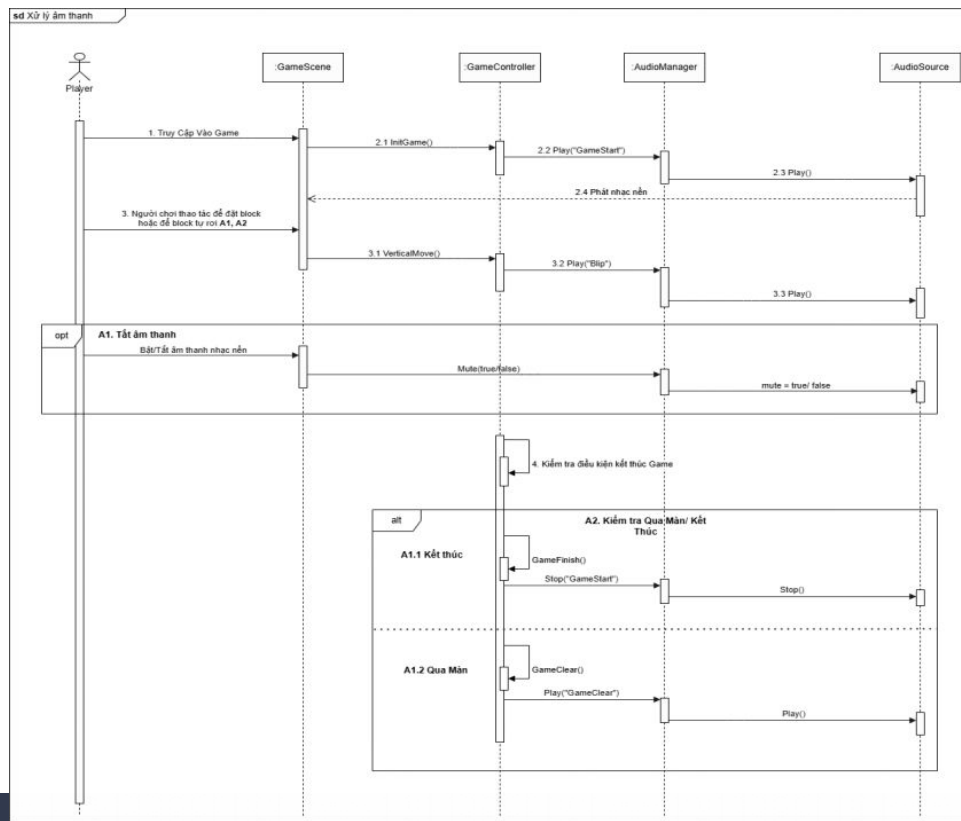
# ◇ Giao diện Xử Lý Âm Thanh



# Phân tích yêu cầu

*~Mô tả hệ thống~*

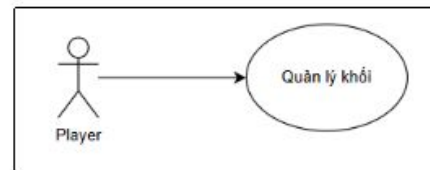
**-UC02-**



# Phân tích yêu cầu

~Mô tả hệ thống~

-UC03-



Use Case Number:	UC3	
Use Case Name:	Quản lý khối	
Actor(s):	Player	
Maturity:	Focused	
Summary:	Player thao tác để điều khiển di chuyển khối đang rơi hiện tại	
Basic Course of Events:	Actor Action	System Response
	1. Actor <u>bắt đầu chơi game</u>	
		2. Hệ thống hiển thị giao diện chơi game
		3. Khối tự rơi xuống theo thời gian
	4. Actor <u>chọn thao tác A1, A2, A3, A4.</u>	
		4. Khối xuất hiện ở vị trí cuối. Use Case kết thúc

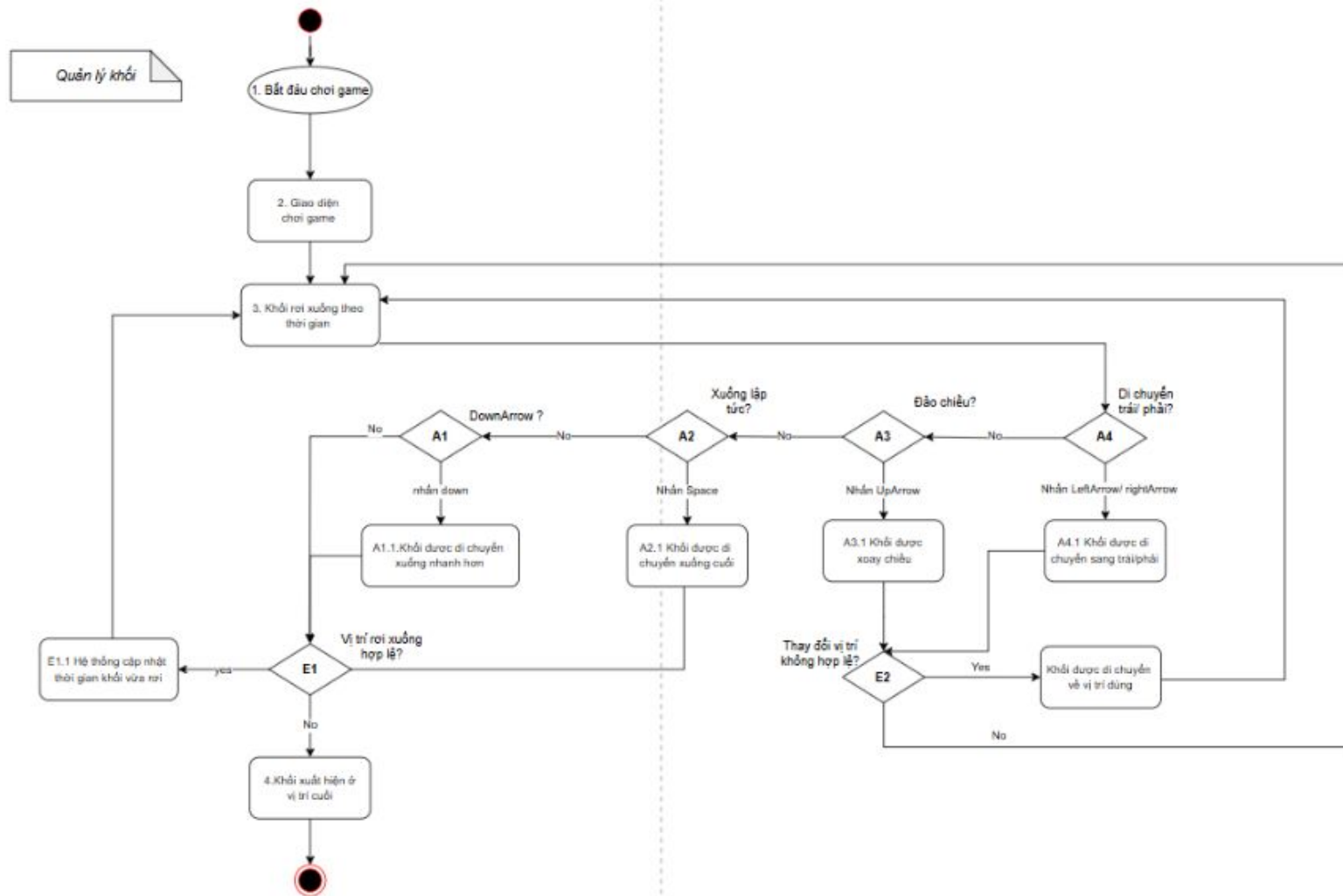
# Phân tích yêu cầu

~*Mô tả hệ thống*~

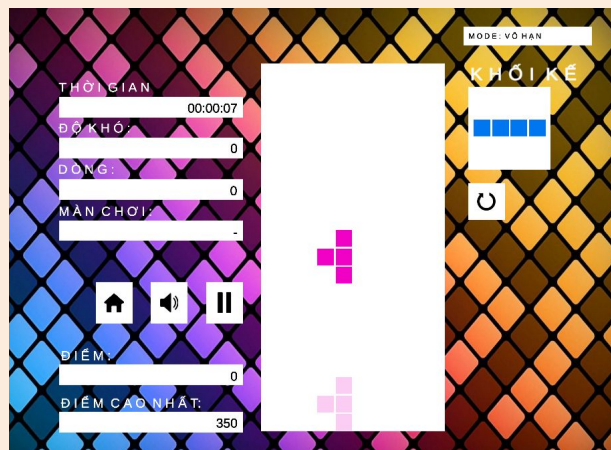
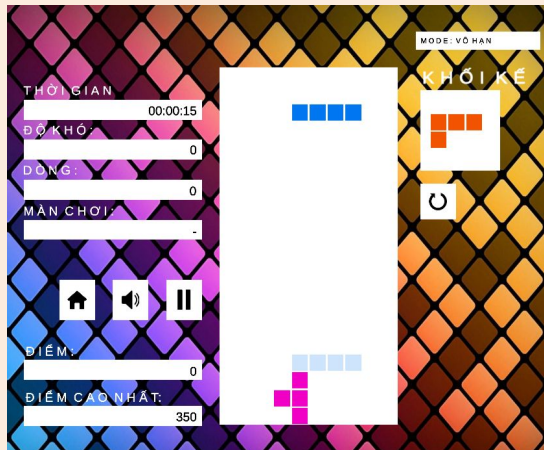
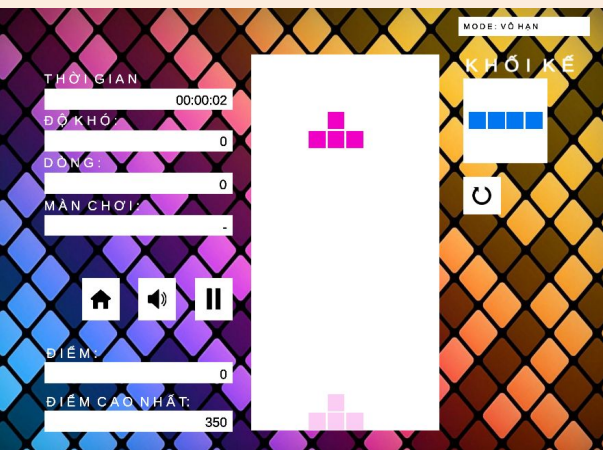
-*UC03*-

Alternative Paths:	A1.Nút downArrow được nhấn thì khối rơi xuống nhanh hơn <b>E1. Quay về bước 6 UC03</b>
	A2. Nút Space được nhấn thì khối rơi xuống vị trí cuối <b>E1 Quay lại bước 3 UC03</b>
	A3 Nút up Arrow được nhấn thì khối đảo chiều 90 độ <b>E2 Quay lại bước 3 UC03</b>
	A4. Nút LeftArrow/ rightArrow được nhấn thì khối được di chuyển sang trái/ phải <b>E2 Quay lại bước 3 UC03</b>
Exception Paths:	<b>E1</b> Nếu vị trí rơi xuống hợp lệ thì hệ thống cập nhật thời gian vừa rơi, <b>Quay lại bước 3 UC03</b>
	<b>E2</b> Nếu vị trí thay đổi không hợp lệ thì hệ thống hiển thị khối ở vị trí đúng <b>Quay lại bước 3 UC03.</b>
Extension Points:	None
Triggers:	Khi Actor bắt đầu trò chơi
Assumptions:	None
Preconditions:	None
Post Conditions:	None
Reference: Business Rules	None
Author(s):	Nhóm 2
Date:	16-12-2024
Activity Diagram: <u>Quản lý khối</u>	

# Activity diagram UC03



# ◊ Giao diện Xử Lý Khối

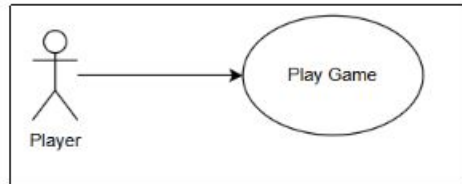


[illegible]

# Phân tích yêu cầu

~*Mô tả hệ thống*~

-UC04-



Use Case Number:	UC04	
Use Case Name:	Play Game	
Actor (s):	Player	
Maturity:	Focused	
Summary:	Actor thực hiện các thao tác bắt đầu trò chơi, chơi game, xử lý các khối và kết thúc trò chơi, với các lựa chọn chơi lại hoặc thoát về màn hình chính.	
	Actor Action	System Response
	1. Actor <u>bắt đầu chơi game</u>	
		2. Hệ thống Xử Lý Âm Thanh A1
		3. Hệ thống hiển thị giao diện chơi "Theo màn"
		4. Hệ thống bắt đầu tạo khối chơi (Khối chính và Khối ảo)
	5. Actor <u>thao tác chơi game A2, A3, A4</u>	
		6. Hệ thống thao tác Xử Lý Khối A5
		7. Hệ thống xóa hàng và cập nhật điểm A6



# Phân tích yêu cầu

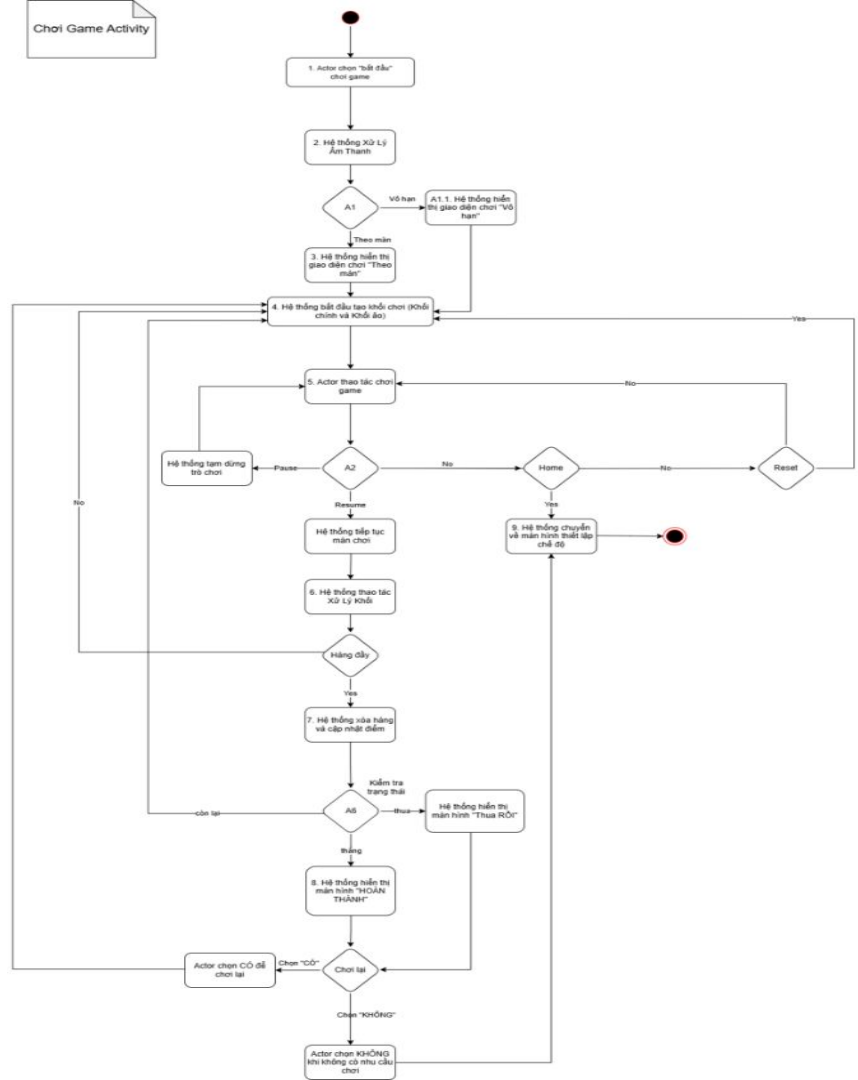
~*Mô tả hệ thống*~

**-UC04-**

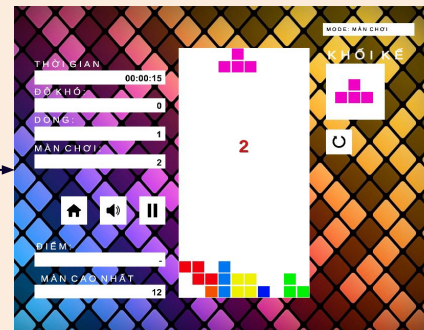
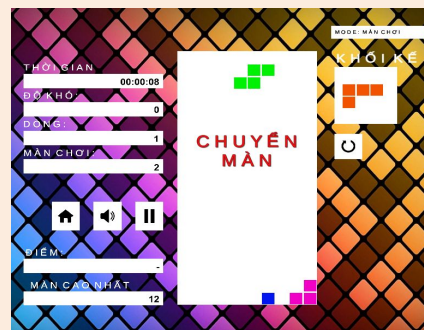
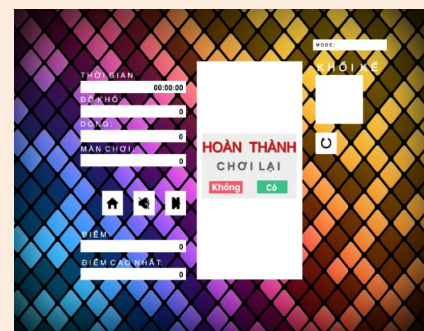
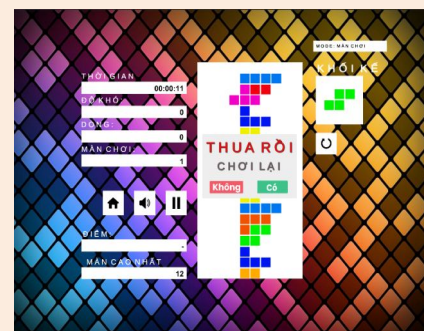
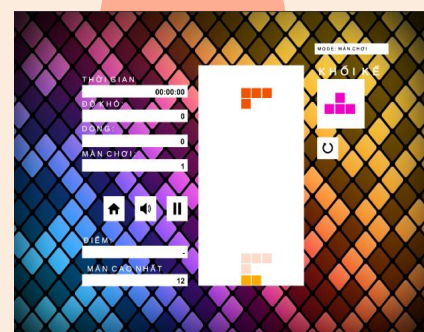
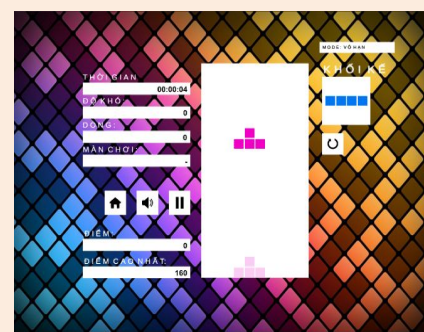
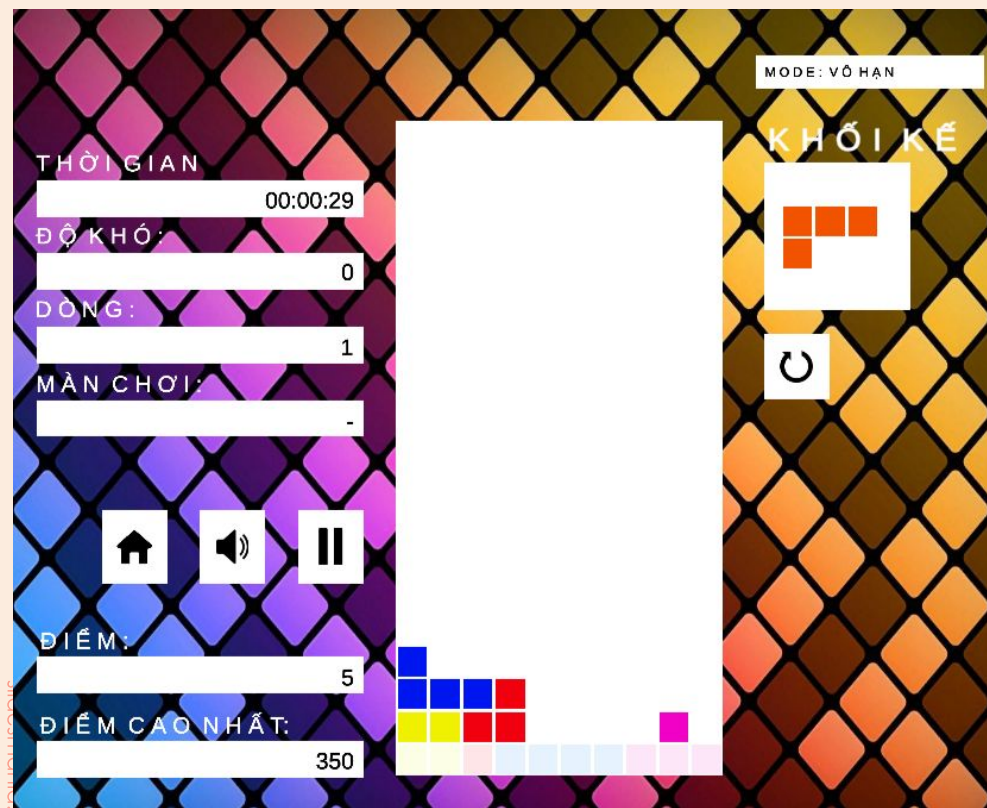
		8. Hệ thống hiển thị màn hình "HOÀN THÀNH" A7
	9. Actor chọn "Không" khi không có nhu cầu chơi lại A7	
		9. Hệ thống chuyển về màn hình thiết lập chế độ Use Case kết thúc tại đây
Alternative Paths:	A1. Hệ thống hiển thị giao diện chơi "Vô hạn". Quay về bước 4 UC04	
	A2 Pause/ Resume	
	A2.1 Hệ thống tiếp tục màn chơi Quay về bước 6 UC04	
	A2.2 Hệ thống tạm dừng trò chơi Quay về bước 5 UC04	
	A3 Actor chọn home Quay lại bước 9 UC04	
	A4. Actor chọn reset Quay lại bước 4 UC04 nếu không Quay lại bước 5 UC04	
	A5. kiểm tra hàng đầy Nếu hàng đầy Quay lại bước 7 UC04 nếu không Quay lại bước 4 UC04	
	A6. Kiểm tra tình trạng A6.1 Nếu thua Hệ thống hiển thị màn hình "thua rồi" A7 A6.2 Nếu thắng hệ thống hiển thị màn hình "Hoàn thành" A7 A6.3 Nếu không Quay lại bước 4 UC04	
Exception Paths:	A7. Người chơi chọn "Có" để chơi lại. Quay lại bước 4 UC04	
	E1 Nếu vị trí rơi xuống hợp lệ thì hệ thống cập nhật thời gian vừa rơi, Quay lại bước 3 UC03	
	E2 Nếu vị trí thay đổi không hợp lệ thì hệ thống hiển thị khối ở vị trí đúng Quay lại bước 3 UC03.	
Extension	None	



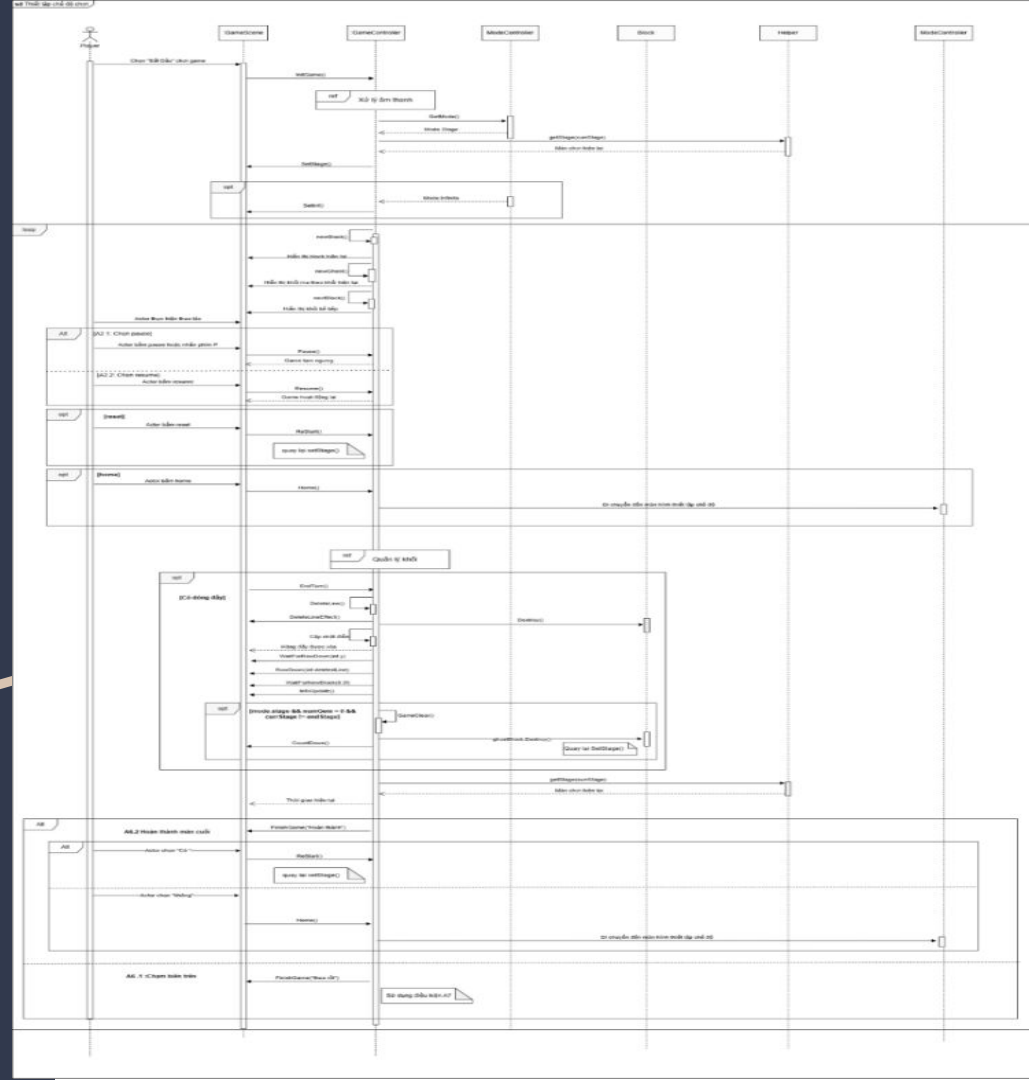
## Activity diagram UC04



# ◆ Giao diện Play Game



## Sequence diagram UC04





# Thiết kế lớp và biểu đồ lớp

## 7 lớp chính

**Sound.cs:** Quản lý thông tin và phát âm thanh (tên, âm lượng, cao độ, chế độ lặp).

**AudioManager.cs:** Điều khiển toàn bộ hệ thống âm thanh (khởi tạo, phát và dừng âm thanh).

**Block.cs:** Class cơ sở cho các khối, quản lý điểm xoay, hiển thị và hủy khối.

**GameController.cs:** Quản lý trạng thái trò chơi, điều khiển khối, xử lý va chạm và UI.

**GuideController.cs:** Hướng dẫn người chơi mới, cung cấp thao tác quay lại màn hình cài đặt.

**ModeController.cs:** Quản lý và chuyển đổi chế độ chơi (vô cực, màn chơi).

**Helper.cs:** Class tiện ích chứa hằng số và cấu hình trò chơi (kích thước lưới, chế độ).





# Thiết kế lớp và biểu đồ lớp

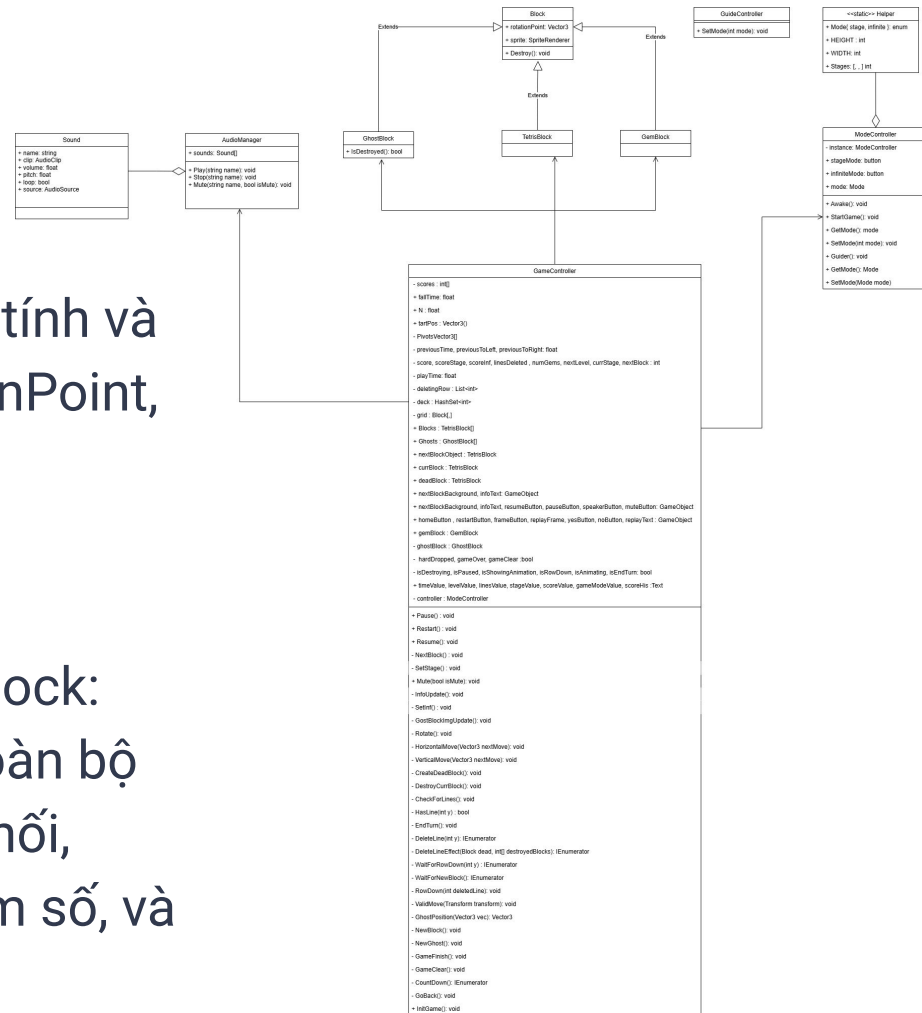
- Do trong game sẽ có các đối tượng là gemBlock, Tetriminos, GhostBlock nên để dễ dàng quản lý và phát triển về sau, ta sẽ tạo ra các lớp con kế thừa từ Block :
  - GemBlock
  - GhostBlock
  - TetrisBlock
- Về bản chất, deadBlock không cần lớp riêng vì không có khác biệt đáng kể về thuộc tính hoặc hành vi so với TetrisBlock. Việc chuyển đổi trạng thái "đóng băng" có thể được xử lý bằng việc đánh dấu lại vị trí đó trên lưới game.





# Thiết kế lớp và biểu đồ lớp

- **Lớp Block (Cơ sở):** Chứa thuộc tính và phương thức chung như rotationPoint, SpriteRenderer, và phương thức Destroy(). (1 method)
- **Các lớp con kế thừa từ Block:** TetrisBlock, GemBlock, GhostBlock:
- **Lớp GameController:** Quản lý toàn bộ quá trình chơi, điều khiển các khối, kiểm tra va chạm, cập nhật điểm số, và kết thúc game. (31 methods)

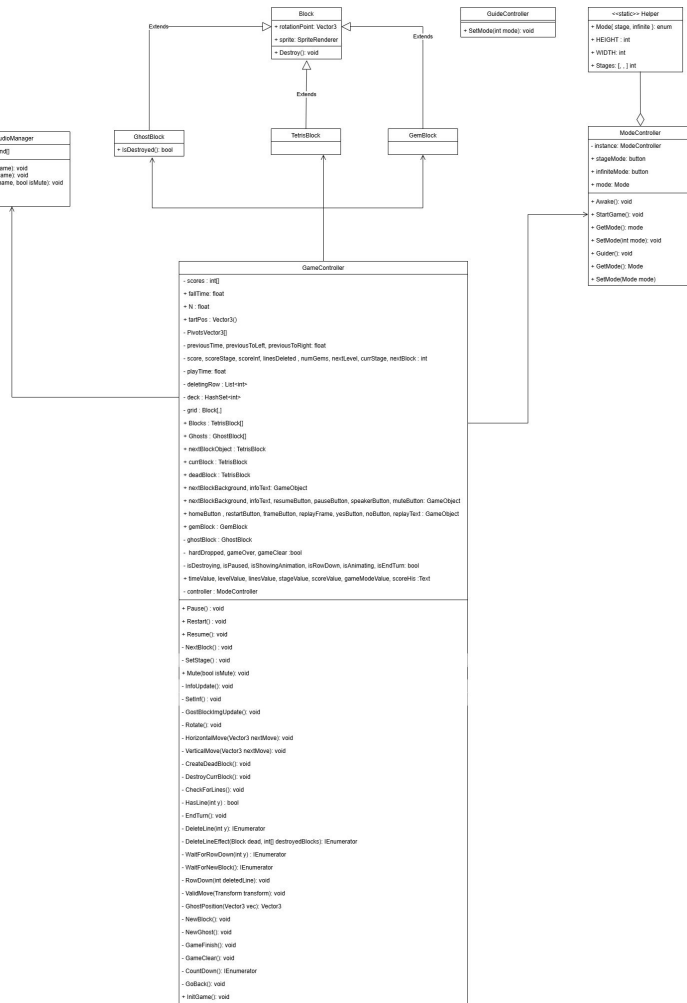




- # Thiết kế độ phức tạp hình ảnh lưới, hướng trò chơi.(1)
- 
- ```

classDiagram
    class Sound {
        + name string
        + clip AudioClip
        + volume float
        + pitch float
        + loop bool
        + clip Audio
    }
    class AudioManager {
        + sounds Sound[]
        + Play(sfxing name) void
        + Stop(sfxing name) void
        + Muting(sfx name, bool sfxMute) void
    }
    class GameBlock {
        + isDestroyed() bool
    }
    class TerrainBlock {
    }
    class GemBlock {
    }
    class GameController {
        - scores int[]
        + gameTime float
        + N: float
        + lastPos Vector3()
        - previousTime, previousTimeLeft, previousTallight float
        - score, scoreStage, scoreUnit, levelDestroyed, numGems, nextLevel, curStage, nextBlock int
        - playTime float
        - destroyRow List<int>
        - deck List<Gem>
        - grid Block[]
        + Blocks TerrainBlock[]
        + Objects GameBlock[]
        + nextBlockObject TerrainBlock
        + curBlock TerrainBlock
        + deadBlock TerrainBlock
        + nextBlockBackground, isNextGem GemObject
        + nextBlockBackground, isNextTerrain, nextBlock, passBlock, speakerButton, multiButton GameObject
        + homeButton, restartButton, transformButton, replayFrame, yesButton, noButton, replayText GameObject
        + pendBlock GameBlock
        - ghostBlock GameBlock
        - hardDropped, gameOver, gameClear bool
        + isDestroying, isPassed, isShowingAnimation, isShowGem, isShowing, isStartTurn bool
        + timeValue, nextValue, levelValue, stageValue, scoreValue, gameEndValue, scoreUnit Text
        - controller ModeController
        + Play() void
        + Restart() void
        + Resume() void
        - nextBlock() void
        - nextStage() void
        + MultiRoll (static) void
        - Info() void
        - Set() void
        - DestroyBlock(update) void
        - Rotate() void
        - MoveOnTable(Vector3 nextBlock) void
        - Vector3Move(Vector3 nextBlock) void
        - CreateDeadBlock() void
        - DestroyCurBlock() void
        - CheckForLines() void
        - HasLine(n) bool
        - isStart() void
        - DeleteLine(n) Enumerator
        - DeleteLineEffectBlock dead, isEffectDestroyBlock() Enumerator
        - WaitForLineDown(n) Enumerator
        - WaitForLineOn(n) Enumerator
        - WaitForLineOn(n) Enumerator
        - RollDown(n) destroyed() void
        - RollDown(Random function) void
        - CheckForLines(n) Vector3 Vector3
        - nextBlock() void
        - NewGem() void
        - GameEnd() void
        - GameClear() void
        - CurBlock() Enumerator
        - GetBlock() void
        + InfoGame() void
    }

    Sound --> AudioManager
    AudioManager --> GameBlock
    GameBlock --> TerrainBlock
    GameBlock --> GemBlock
    GameController --> GameBlock
    GameController --> GemBlock
    GameController --> GameController
  
```
- The diagram illustrates the architecture of a game engine, focusing on the management of sound, game blocks, and the game controller. The **Sound** class represents audio assets with attributes like name, clip, volume, pitch, and loop status. The **AudioManager** class manages a collection of sounds and provides methods for playing, stopping, and muting them. The **GameBlock** class is a base class for game elements, with **TerrainBlock** and **GemBlock** as subclasses. The **GameController** class is the central logic component, managing game state, scores, and the flow of the game. It interacts with the **AudioManager** and **GameBlock** classes to provide a cohesive gaming experience.







## ~Cách vận hành~

**Khởi tạo trò chơi:** Sử dụng GameController để khởi tạo lưới, khối và trạng thái ban đầu.

**Điều khiển:**

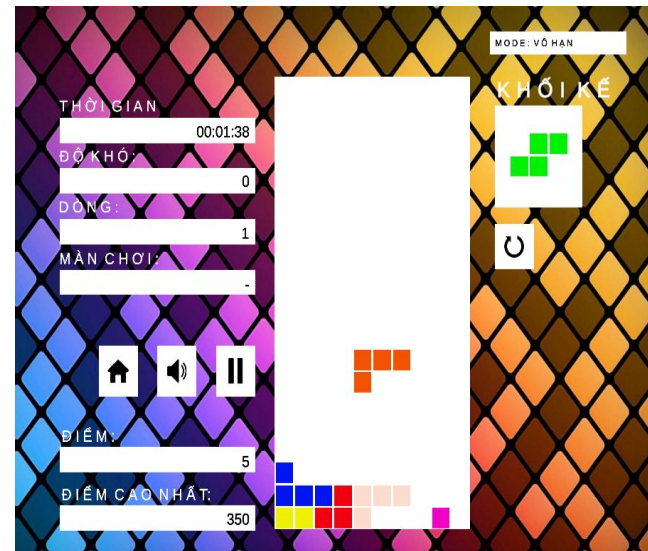
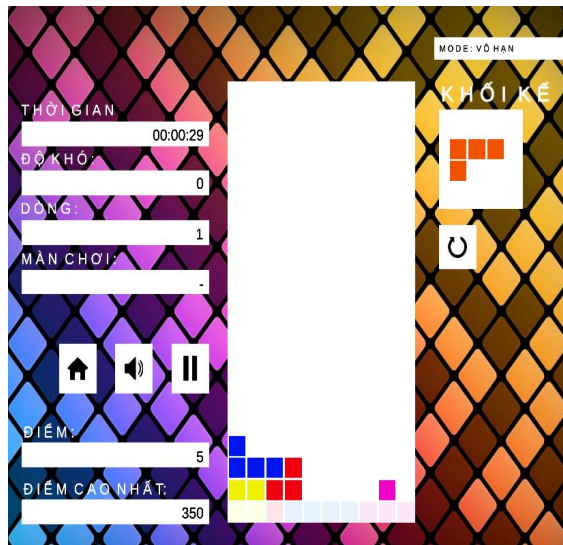
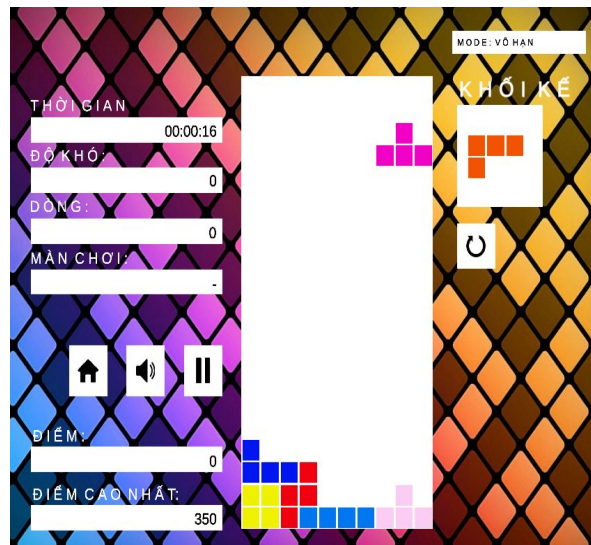
- Xoay khối (phím **Up**).
- Di chuyển trái/phải (phím **Left/Right**).
- Hạ nhanh khối (**Hard Drop**, phím **Space**).

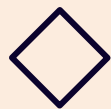
**Kiểm tra va chạm:** Đảm bảo khối không vượt lưới hoặc chồng lên khối khác.

**Xóa hàng:** Hàng lấp đầy được xóa và các khối trên rơi xuống.

**Chuyển chế độ:** Hỗ trợ chơi theo màn và chế độ vô hạn.

# Quy trình xử lý xóa hàng - cập nhật điểm





# Các thư viện và công nghệ sử dụng

- **UnityEngine**: Quản lý các đối tượng trong trò chơi (GameObject, Transform).
- **Unity UI**: Xây dựng giao diện HUD (TextMeshPro, Buttons).
- **System.Collections** và **System**: Xử lý vòng lặp trò chơi, sự kiện, và dữ liệu.
- **PlayerPrefs**: Lưu trữ điểm số và tiến trình.
- **Random**: Sinh khối Tetrimino ngẫu nhiên.
- **SpriteRenderer** (*component trong Unity*) : chỉnh sửa màu sắc, độ mờ của đối tượng
- **Coroutine** trong Unity để trì hoãn thời gian giúp thực hiện hiệu ứng thay đổi chậm hơn

```
while (_progress > 0.0f)
{
    dead.sprite.GetComponent<SpriteRenderer>().color
        = new Color(tmp.r, tmp.g, tmp.b, tmp.a * _progress);
    _progress -= 0.1f;
    yield return new WaitForSeconds(0.03f);
}
```

```
if ((currStage + 1) >= scoreStage)
{
    scoreHis.text = (currStage + 1).ToString();

    // Lưu điểm số
    PlayerPrefs.SetInt("ScoreStage", (currStage + 1));
    PlayerPrefs.Save();
}
else scoreHis.text = scoreStage.ToString();
```

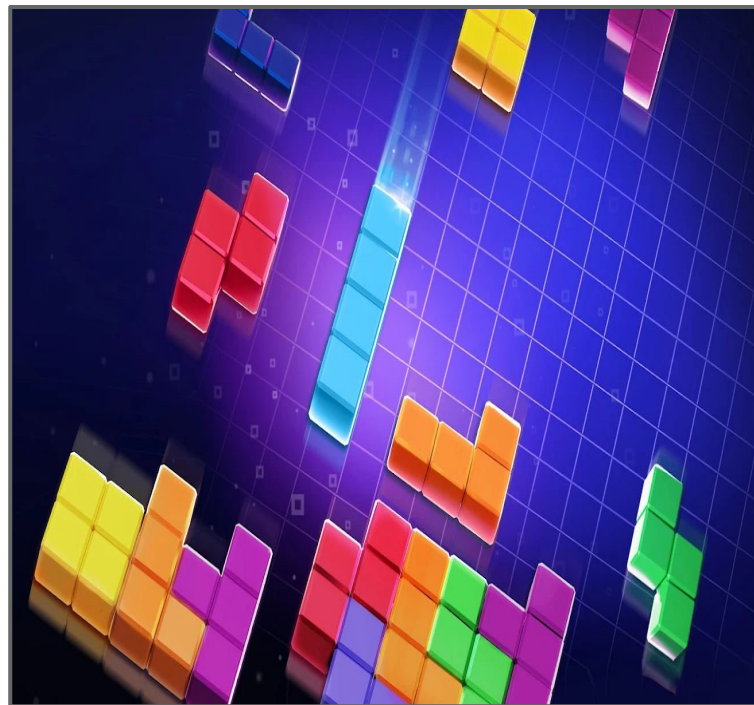
```
3 references
void NextBlock()
{
    print("nextblock start");
    if (deck.Count == Blocks.Length) deck.Clear();
    do nextBlock = Random.Range(0, Blocks.Length);
    while (deck.Contains(nextBlock));
    deck.Add(nextBlock);
}
```



# V. Đánh giá và hướng phát triển

*~Đánh giá~*

- Lối chơi đơn giản nhưng đầy thách thức
- Rèn luyện khả năng tư duy chiến lược và phản xạ nhanh
- Tính dễ tiếp cận và phù hợp với tất cả mọi lứa tuổi





# V. Đánh giá và hướng phát triển

*~Hướng phát triển~*

- **Mở rộng chế độ chơi.**

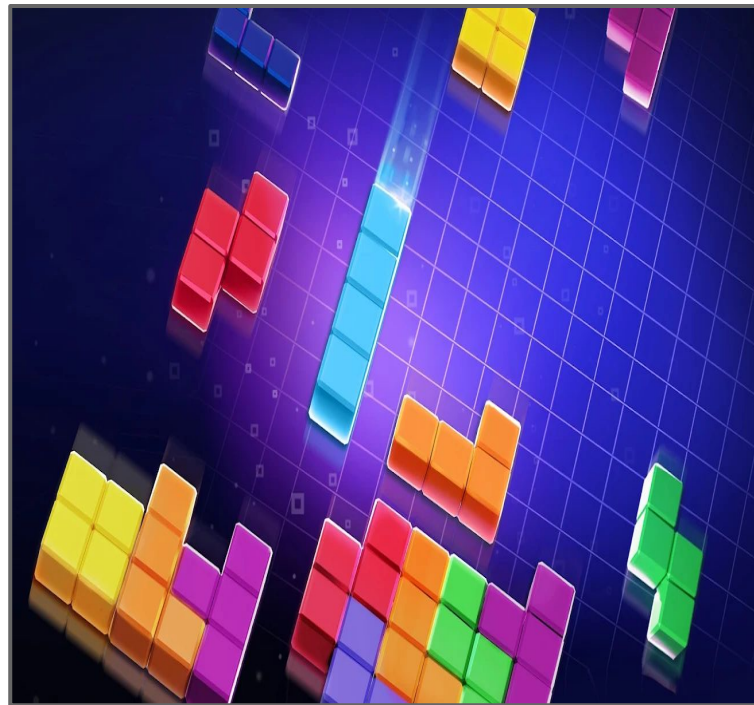
- + **Chạy đua**

- Mục tiêu: Xóa một số lượng dòng nhất định trong thời gian ngắn nhất

- + **Siêu tốc**

- Mục tiêu: Hoàn thành một số lượng dòng trong thời gian ngắn nhất.

- **Thêm tính năng Multiplayer**



An abstract geometric composition on a dark purple background. It features several 3D ring-like shapes in shades of purple and orange. One large orange ring is positioned in the center-left, resting on a white grid. Other purple and orange rings are scattered around it. Small white-outlined diamonds and 3D cubes are also present, adding to the geometric theme. A large orange circle is in the top right corner.

# Thank you!

Do you have any questions?

