ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA CÔNG NGHỆ PHẦN MỀM

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN 1

ĐỀ TÀI:
DUNGEON CRAWLER GAME

Giảng viên:
Trần Anh Dũng

Sinh viên thực hiện:
Nguyễn Minh Thiện – 19522262

Thành phố Hồ Chí Minh, năm 2022

VIETNAM NATIONAL UNIVERSITY - HO CHI MINH CITY
UNIVERSITY OF INFORMATION TECHNOLOGY
THE FACULTY OF SOFTWARE ENGINEERING

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

PROJECT 1 REPORT

PROJECT:
DUNGEON CRAWLER GAME

Instructor:
Trần Anh Dũng

Student:
Nguyễn Minh Thiện – 19522262

Ho Chi Minh City, 2022

Document change record sheet

Date	Version	Description	Authors
25/9/2021	0.1	Added basic game function like player movement shooting, background, basic collision	Nguyễn Minh Thiện. Nguyễn Nhật Huy
02/1/2022	1.0	Adding layout	Nguyễn Minh Thiện. Nguyễn Nhật Huy
05/1/2022	1.1	Finalize report of v1.0 of the game	Nguyễn Minh Thiện. Nguyễn Nhật Huy
15/6/2022	1.2	Update report for v1.2.5 of the game	Nguyễn Minh Thiện

Disclaimer:

This game is used as a study material we do not intend to publish or making a profit from this. If anyone who read this document and saw their assets in our game without your permission, please don’t sue us

Table of Contents

Table of Contents 2

Images Table of Content 6

ACKNOWLEDGEMENT 7

DETAILED OUTLINE 8

CHAPTER I: INTRODUCTION 13

I. Basic Information **13**

1. Group Information 13
2. Project Information 13
3. Technology used 13

3.1. Unity 13
3.2. C# 14
3.3. Github 15
3.4. VSCode 15

II. Statement of the Project **16**

1. Game concept 16

1.1. Purpose of the game 16
1.2. Story 16
1.3. Features 17

2. Brief description of entities and objects in the game 17

2.1. Entities 17

2.1.1. Player 17
2.1.2. Slime 17
2.1.3. Archer 18
2.1.4. Shield Enemy 18
2.1.5. Bomber Enemy (Also known as SuicideEnemy) 18
2.1.6. Boss Golem 18
2.1.7. Shop NPC 18
2.1.8. Bringer of Death Boss 18

2.2. Objects 19

2.2.1. Bullet 19
2.2.2. Portal 19
2.2.3. Buffs 19
2.2.4. Non-Moving Objects 19
2.2.5. Drops 19
2.2.6. Item 19

3. Functional requirements 19
4. Non-functional requirements 20

4.1. Interface requirements 20
4.2. Game balancing requirements 20
4.3. Compatibility requirements 20
4.4. Quality and efficiency requirements 20

CHAPTER II: USE CASE _____ *21*

I. User story _____ **21**

II. Use case models _____ **22**

1. Use case diagrams _____ 22
1.1. Player _____ 22
1.2. Slime _____ 23
1.3. Archer _____ 24
1.4. Golem _____ 25
1.5. ShopNPC _____ 26
1.6. Shield Enemy _____ 26
1.7. Bomber _____ 27
1.8. Audio Manager _____ 27
1.9. Bringer Of Death Boss _____ 28
2. Actor list _____ 28
3. Use case list _____ 29
4. Use case specification _____ 32
4.1. Play game _____ 32
4.2. Exit menu _____ 32
4.3. Pause game _____ 33
4.4. Save game _____ 33
4.5. New game _____ 34
4.6. Load game _____ 34
4.7. Restart game _____ 35
4.8. Interact with interactable objects/NPC _____ 35
4.9. Melee Slash _____ 36
4.10. Firing projectiles _____ 36
4.11. Consume buffs _____ 37
4.12. Check restrict movement _____ 37
4.13. Check HP _____ 38
4.14. Die _____ 38
4.15. Take damage _____ 39
4.16. Moving around _____ 39
4.17. Melee Attack _____ 40
4.18. Seek Player _____ 40
4.19. Go idle _____ 41
4.20. Slam _____ 41
4.21. Fire laser _____ 42
4.22. Roll _____ 42
4.23. Fire arm _____ 43
4.24. Check Cooldown _____ 43
4.25. Summon Pillars _____ 44
4.26. Pulling player _____ 44
4.27. Aim _____ 45
4.28. Interact with NPC _____ 45
4.29. Display purchase state _____ 46
4.30. Trade coins _____ 46

4.31. Verify Eligibility 47

4.32. Explode 47

4.33. Shield Take Damage 48

4.34. Activate Shield 48

4.35. Deactivate Shield 49

4.36. Teleport 49

4.37. Clone 49

4.38. Cast Spell 50

4.39. Play Sound 50

4.40. Play Soundtrack 51

4.41. Remove Sound 51

4.42. Fade Soundtrack 52

4.43. Get Audio Clip 52

4.44. Check Sound Availability 53

CHAPTER III: ANALYSIS DESIGN 54

I. Classes Analysis 54

1. Class diagrams 54

1.1. Player and Enemy class Diagram 54

1.2. Interactable Class Diagram 55

1.3. Menu and Managers Class Diagram 55

2. List of classes and relationships 56

2.1. Classes 56

2.2. Relationships 57

3. Classes in details 59

II. State Diagrams 79

1. Player state diagram 79

2. Slime state diagram 80

3. Archer state diagram 81

4. Golem state diagram 82

5. Bullet state diagram 83

6. Non-Moving Object state diagram 84

7. ShopNPC state diagram 85

8. Buff state diagram 86

9. ShieldEnemy state diagram 87

10. Bomber state diagram 88

11. BringerOfDeathBoss state diagram 89

I. Player data 90

II. Modifiers Data 90

CHAPTER V: STRUCTURAL DESIGN 92

I. Game design 92

1. Core Game Loop 92

2. Progression 92

II. Software design 92

III. Design Patterns Use 93

CHAPTER VI: INTERFACE DESIGN ***94***

I. Regulation 94

II. List of UI 94

III. Details description 94

1. Main menu screen 94
2. Player's Health bar 96
   2.1. Circular health bar 96
   2.2. Status effect bar 96
3. Player stats sheet 96
4. Number Pop up 97
5. Player exit to menu and retreat to hideout UI 98
6. Tutorial UI 99
7. Coins counter 99
8. NPCShop 99
9. Interact with object buttons 101
10. Item tooltip 101
11. Inventory 102

CHAPTER VII: INSTALLATION ***103***

I. Unity Hub and Editor 103

II. The game 104

CHAPTER VIII: CONCLUSION ***105***

I. Result 105

II. Limitations 105

III. Future Scope 105

REFERENCES ***107***

1. Unity 107
2. Unity Basic Tutorial (YouTube) 107
3. Unity UI tutorial (YouTube/Forums) 107
4. Assets "borrowed" 108
5. Creating assets/VFX tutorial and tool use to create assets 108
6. Path finding 108
7. Buff weight and pool references 108
8. Fire rate reference 108
9. Player's Dash tutorial 108
10. Audio Manager tutorial 109
11. Item tooltip tutorial 109

WORKLOAD DISTRIBUTION ***110***

Images Table of Content

Image 1: Unity Logo 13
Image 2: C# logo 14
Image 3: Github logo 15
Image 4: VSCode logo 15
Image 5: Player Use Case Diagram 22
Image 6: Slime Use Case Diagram 23
Image 7: Archer Use Case Diagram 24
Image 8: Golem Use Case Diagram 25
Image 9: Shope NPC Use Case Diagram 26
Image 10: Shield Enemy Use Case Diagram 26
Image 11: Bomber Use Case Diagram 27
Image 12: Audio Manager Use Case Diagram 27
Image 13: Bringer of Death Boss Use Case Diagram 28
Image 14: Player and Enemies Class Diagram 54
Image 15: Interactable Class Diagram 55
Image 16: Menu and Managers Class Diagram 55
Image 17: Player State Diagram 79
Image 18: Slime State Diagram 80
Image 19: Archer State Diagram 81
Image 20: Golem State Diagram 82
Image 21: Bullet State Diagram 83
Image 22: Non-Moving Object State Diagram 84
Image 23: Shop NPC State Diagram 85
Image 24: Buff State Diagram 86
Image 25: Shield Enemy State Diagram 87
Image 26: Bomber State Diagram 88
Image 27: Bringer of Death Boss State Diagram 89
Image 28: The game structural design 92
Image 29: Main menu UI 95
Image 30: Save slots options UI 96
Image 31: Player Health bar and Status Effect bar 96
Image 32: Player Stat Sheet UI 97
Image 33: Damage Pop Up 97
Image 34: Exit to main menu button UI 98
Image 35: Retreat button UI 98
Image 36: Retreat Confirmation Button UI 98
Image 37: Tutorial UI 99
Image 38: Coin Counter UI 99
Image 39: NPC Dialogue 1 100
Image 40: NPC Dialogue 2 101
Image 41: Interact Button UI 101
Image 42: Item Tooltip UI 102
Image 43: Inventory UI 102

ACKNOWLEDGEMENT

After studying and training at the Faculty of Software Engineering, University of Information Technology - Vietnam National University, Ho Chi Minh City, we have been equipped with basic knowledge and practical skills to be able to do Project 1 course.

To be able to fnish this course we would like to express our deepest gratitude toward those who made this all possible:

The Board of Directors of the University of Information Technology - Vietnam National University, Ho Chi Minh City for creating favorable conditions in terms of facilities with a modern library system and with a variety of books and documents that are convenient for searching and researching information.

We would like to express our sincere thanks to Mr. Tran Anh Dung for his enthusiastic helps and guidances in scientific thinking and working methods. Those are very valuable suggestions not only in the process of doing this project, but also as a steppingstone for us in the process of studying and setting up a career in the future.

Next, we would like to thank Brackeys, Code Monkey, Blackthornprod and various other youtubers for making all those wonderful and insightful tutorial videos for Unity to help us develop this game.

And finally, we would like to thank our family, all teachers in the faculty, friends, and the whole class of KTPM2019 who are always ready to share and help in study and life. Hopefully, we will be able to work together again in the near future.

In the process of doing this project, mistakes and missteps are unavoidable, if possible, we would like to receive guidances, suggestions and advice from our dear teachers to improve and develop the project more in the near future for Project 2 course as well as in the future Graduation Thesis if we ever take it.

Sincerely thank you. Wish the best of luck will always accompany you all

Ho Chi Minh City 14/06/2022

Students

Nguyễn Minh Thiện

DETAILED OUTLINE

PROJECT NAME: Dungeon Crawler Game

Instructor: ThS. Trần Anh Dũng

Implementation time: From 21/02/22 To 18/6/2022

Students:

Nguyễn Minh Thiện - 19522262

Project information:

1. **Purpose of the project:**

Before this course we learnt about Object-oriented analysis and design (OOAD) which is a technical approach for analyzing and designing an application, system, or business by applying object-oriented programming. A game is made up of objects and their interactions with each other and the environment around them. And if we want to make a more interesting game that’s not just some bricks falling out of the sky that fill in a line or a ball bouncing between 2 brackets on each side of the screen, we need to utilize OOP. We also learnt about design patterns during some of the seminars of the courses we have taken which is also being used a lot in game development like singletone and component design pattern. Moreover, we want to learn about game development and Unity seem to be the most beginner friendly and accessible. So, to put our knowledge to use after we have learnt more about software development, we decide to make a game

2. **Game genre**

- This game is a 2D, dungeon crawler, Roguelike, shooter games

- Currently this is a single player game

- Like any other dungeon crawler game, the game will have a stage where player spawn in whenever they died during their run or when they load their saves. During their gameplay they will encounter variety of enemy types like slime, archer, bomber, shield and also some big boss at the end of the loop when they defeat the boss they go back to their hideout (which is the place the player spawn in) and start another loop. Player can shoot at the moster with projectiles that can be changed to different element like fire, cold, lightning and physical. Player also has an inventory to store the item they have picked up from killing the monster, the inventory will lose all the items in it when player died or when player defeat the boss and go back to hideout. Item and Buffs will give player permanent stats boost during their gameplay until they died or return to hideout. Gold coins can be dropped from monsters when they died, and player can use that coin to buy themselves "slots" from the Goblin NPC which is used to give player permanent buff during their gameplay at the start when player enter a portal to an instance.

3. Scope

Environmental scope: The game can be play on Window 10 or Linux and you can aslo play on some of the io site that host the game like itch.io and simmer.io (link down below in the Installation chapter)

Features scope:

- When playing the game, the player can shoot out elemental projectile that canbe changed by consuming a buff
- A buff can also change the player total health point, fire type, movespeed, defense etc
- Player can also knock back enemies with their melee slash
- Player can use a short cool down dash to avoid incoming hostile projectiles and to escape from getting rush down by monsters around them
- Player can buy permanent slots from the green goblin to randomly gain quirks depending on the number of slots they have to start a run with some advantages.
- Player can also access a tutorial area that has a dummy and 2 commonly see monster types and test out all the buffs there currently are in the game
- A counter to count the amount of coins player have
- Boss enemy type at the end of an instance

- Player can apply status effect on monsters by shooting and also can be affected by status effect coming from monsters' attack
- Player can save and load up to 3 save slots and it will save player slots and current coins
- Player gets invincibility time whenever they take damage this is so that player won't get bombarded by projectiles and swamp by melee enemies and die almost instantly
- Weighting system for buffs and using cumulative probability to spawn random buff for player when they cleared a room and also to apply random quirks when player enter an instance
- Items with modifiers and weights can be dropped from monsters for player to pick up and equip and get their stats increase

4. Target audiences

Anyone who like to play dungeon crawler, roguelike game

5. Implementation

- Working online and offline with the instructor during the course

- Learning Unity and doing research on game design to develop features for the game

- Using C# to write script for the game

- Version control through Github

- Google doc and Mircrosoft word to do report and documents. Using Draw.io or similar software to draw diagram

- Spending 1 week or less to clean up the code and organize it

- 2 weeks of development for each major features prioritizing adding sound effect to the game

6. Platforms and Environments used for development

- Platforms: Windows 10

- Environments: Unity

- VCS: Github

- Language: C#

- IDE: VS Code, Visual Studio Community

- Diagram drawing software: Draw.io

7. Project development direction

- Items with modifiers and weight for player to pick up and equip

- New boss and 2 new enemy types

- Sound effect and background music for the game

- Improving UI and fixing bugs

8. Expected result

- The game run smoothly and no game breaking bugs

- Boss and enemy behave as intended and not out of place breaking the game

- Collision detection working well and player not clipping out of the walls

- Can play the game on some sites that host it with all the features like in the PC version

9. Work plan

Implmenting time: from the beginning of Project 1 course to the end of it

We split the development process into 3 phases each phase will typically take around 3-4 weeks to finish

Phase 1:

- Reorganize and refactoring the code base to the MVC model

- Adding sound effect and background music

- Fixing bugs with the old Golem Boss and all bugs that we can find during our play test

Phase 2:

- Designing movesets and adding new "Bringer of Death" boss

- Designing movesets and adding 2 new enemy types bomber and shield enemy

Phase 3:

- Adding Inventory UI and Inventory

- Adding items with modifiers system and weight system for items

- Adding game “loop” that increase difficulty each loop by using a modifers system develop to increase the monsters’ and bosses’ stats

10. Work Distribution

Nguyễn Minh Thiện:

+ doing documents, report

+ researching game design for the game

+ developing new features for the game and fixing bugs

CHAPTER I: INTRODUCTION

I. Basic Information

1. Group Information

Student ID	Name	Email
19522262	Nguyễn Minh Thiện	19522262@gm.uit.edu.vn negativetwoyt@gmail.com

2. Project Information

- Project name: Dungeon Crawler Game
- Game name: Pizza delivery quest
- Technology: Unity Engine
- Game development environment: Desktop game on windows 10

3. Technology used

3.1. Unity

Image 1: Unity Logo

Unity is a cross-platform game engine developed by Unity Technologies, first announced and released in June 2005 at Apple Worldwide Developers Conference as a Mac OS X game engine. The engine has since been gradually extended to support a variety of desktop, mobile, console and virtual reality platforms. It is particularly popular for iOS and Android mobile game development and is considered easy to use for beginner developers and is popular for indie game development.

The engine can be used to create three-dimensional (3D) and two-dimensional (2D) games, as well as interactive simulations and other experiences. The engine has been adopted by industries outside video gaming, such as film, automotive, architecture, engineering, construction, and the United States Armed Forces.

Unity gives users the ability to create games and experiences in both 2D and 3D, and the engine offers a primary scripting API in C# using Mono, for both the Unity editor in the form of plugins, and games themselves, as well as drag and drop functionality. Prior to C# being the primary programming language used for the engine, it previously supported Boo, which was removed with the release of Unity 5, and a Boo-based implementation of JavaScript called UnityScript, which was deprecated in August 2017, after the release of Unity 2017.1, in favor of C#.

Within 2D games, Unity allows importation of sprites and an advanced 2D world renderer. For 3D games, Unity allows specification of texture compression, mipmaps, and resolution settings for each platform that the game engine supports, and provides support for bump mapping, reflection mapping, parallax mapping, screen space ambient occlusion (SSAO), dynamic shadows using shadow maps, render-to-texture and full-screen post-processing effects.

3.2. C#

Image 2: C# logo

C# is a general-purpose, multi-paradigm programming language. C# encompasses static typing, strong typing, lexically scoped, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines.

The C# programming language was designed by Anders Hejlsberg from Microsoft in 2000 and was later approved as an international standard by Ecma (ECMA-334) in 2002 and ISO/IEC (ISO/IEC 23270) in 2003. Microsoft introduced C# along with .NET Framework and Visual Studio, both of which were closed-source. At the time, Microsoft had no open-source products. Four years later, in 2004, a free and open-source project called Mono began, providing a cross-platform compiler and runtime environment for the C# programming language. A decade later, Microsoft released Visual Studio Code (code editor), Roslyn (compiler), and the unified .NET platform (software framework), all of which support C# and are free, open-source, and cross-platform. Mono also joined Microsoft but was not merged into .NET.

3.3. Github

Image 3: Github logo

GitHub, Inc. is a provider of Internet hosting for software development and version control using Git. It offers the distributed version control and source code management (SCM) functionality of Git, plus its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, continuous integration, and wikis for every project. Headquartered in California, it has been a subsidiary of Microsoft since 2018.

It is commonly used to host open-source projects. As of November 2021, GitHub reports having over 73 million developers and more than 200 million repositories (including at least 28 million public repositories). It is the largest source code host as of November 2021

3.4. VSCode

Image 4: VSCode logo

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

In the Stack Overflow 2021 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 70% of 82,000 respondents reporting that they use it.

II. Statement of the Project

1. Game concept

1.1. Purpose of the game

- Before this course we learnt about Object-oriented analysis and design (OOAD) which is a technical approach for analyzing and designing an application, system, or business by applying object-oriented programming. A game is made up of objects and their interactions with each other and the environment around them. And if we want to make a more interesting game that's not just some bricks falling out of the sky that fill in a line or a ball bouncing between 2 brackets on each side of the screen, we need to utilize OOP. We also learnt about design patterns during some of the seminars of the courses we have taken which is also being used a lot in game development like singletone and component design pattern. Moreover, we want to learn about game development and Unity seem to be the most beginner friendly and accessible. So, to put our knowledge to use after we have learnt more about software development, we decide to make a game
- This game is a 2D, dungeon crawler, Roguelike, shooter games
- Currently this is a single player game

1.2. Story

In the grim dark of the far future of the 41st Millennium. For more than a hundred centuries The Emperor of Man has sat immobile on the Golden Throne of Earth. He is the Master of Mankind by the will of the gods, and master of a million worlds by the might of his inexhaustible armies. He is a rotting carcass writhing invisibly with power from the Dark Age of Technology. He is the Carrion Lord of the Imperium for whom a thousand souls are sacrificed every day, so that he may never truly die.

You, a merely pizza delivery boy trying to make ends meet in this dark twisted world crash landed on an unknown planet yet to be discovered by the Imperium during one of your delivery trips to Earth. You wake up and miraculously completely unharmed, but your ship is beyond repair, the only living creature that seems to be friendly near you is a big fat green goblin with a pouch of what seem like coins standing next to a table. You suddenly realize there is a portal ahead of you but there is no certainty what lie beyond, but with no other choice but to move on and try to survive you grab your trusty elemental conversion rifle and force field stick and head in not fully prepared for what lie beyond the portal

1.3. Features

- When playing the game, the player can shoot out elemental projectile that can be changed by consuming a buff
- A buff can also change the player total health point, fire type, movespeed, defense etc
- Player can also knock back enemies with their melee slash
- Player can use a short cool down dash to avoid incoming hostile projectiles and to escape from getting rush down by monsters around them
- Player can buy permanent slots from the green goblin to randomly gain quirks depending on the number of slots they have to start a run with some advantages.
- Player can also access a tutorial area that has a dummy and 2 commonly see monster types and test out all the buffs there currently are in the game
- A counter to count the amount of coins player have
- Boss enemy type at the end of an instance
- Player can apply status effect on monsters by shooting and also can be affected by status effect coming from monsters' attack
- Player can save and load up to 3 save slots and it will save player slots and current coins
- Player gets invincibility time whenever they take damage this is so that player won't get bombarded by projectiles and swamp by melee enemies and die almost instantly
- Weighting system for buffs and using cumulative probability to spawn random buff for player when they cleared a room and also to apply random quirks when player enter an instance
- Items with modifiers and weights can be dropped from monsters for player to pick up and equip and get their stats increase

2. Brief description of entities and objects in the game

2.1. Entities

2.1.1. Player

- Player is a game object that the user can control and move around doing all kinds of stuff depending on the rules of the game. Player can do all of the things mentioned in the feature section above.

2.1.2. Slime

- Slime is a melee enemy time that chases down player and hit them with a physical attack. Slime attack might be weak, but they make up for it with their tankiness and their large group attack

2.1.3. Archer

- Archer is a range type enemy with relatively low health. Their attack can apply cold status effect which can freeze player, they're extremely dangerous in group and can fire projectiles constantly at player almost making player unable to move at all

2.1.4. Shield Enemy

- Shield enemy is a melee type with low movement speed but high on defense and can put up their shield to block player projectiles and take completely zero damage from the side of the shield that's facing forward. When they have their shield up their damage increase depend on their damage multiplier. They do lower their shield after a certain amount of time so player can use that down time window to take them out

2.1.5. Bomber Enemy (Also known as SuicideEnemy)

- Bomber enemy is also a melee type with medium amount of hp but they have high movespeed. They track player down and explode dealing massive damage to player in the blast radius

2.1.6. Boss Golem

- Boss golem, an extremely dangerous monster that can one shot player if they're not paying attention with relatively high-speed Boss Golem can catch up to the player and deliver a slam that deals half of the player max hp. If Golem hp reduce to a certain threshold it will initiate it second phase attack firing a laser that instantly kill player if hit, player can hide behind pillar that get summon before golem fire its laser to avoid getting killed

2.1.7. Shop NPC

- ShopNPC which is the green goblin supplies player with slots to get quirks at the beginning of each run, but it is going to cost a lot of coins to buy 1 slot, slots will also increase in cost each time the player buys.

2.1.8. Bringer of Death Boss

- Bringer of Death Boss is one of the Boss that spawn at the end of each instance. It doesn't have high damage like Boss Golem, but it can attack much faster with lower cooldowns, and it can summon its death grasp spell that attack player from above to disrupt player movement and poking away player hp. It can also teleport to player when its health is lower to a certain threshhold, and it can also clone itself to create mirage image that can attack player at lower attack speed and lower damage to confuse and disrupt player. These clones can't cast their spell or teleport to player it can only use the default attack

2.2. Objects

2.2.1. Bullet

- Bullet is the projectile that player and range type enemy can shoot out. Bullet currently has 4 types of elements: physical, fire, cold and lightning. Bullet has a 25% chance to apply status effect and only fire, cold and lightning has status effect to apply

2.2.2. Portal

- Portal is what player use to enter an instance and start playing the game. It's also the thing that loop player back to the hideout to start another instance of the game.

2.2.3. Buffs

- Buffs are what player use to enhance and power themselves up. There are currently 12 buffs available.

2.2.4. Non-Moving Objects

- Non-moving Objects are objects in the game that can be damage but doesn't move around. Some Non-moving objects can be hit but can't take damage and be destroyed.

2.2.5. Drops

- Drops are items that drop from monsters after player kill them. Monsters can drop:

  + A random amount of coins
  + A potion to heal 10% of player max health points
  + An item with random modifiers (can only have 1-3 modifiers on a single item)

2.2.6. Item

- Item is object that can be drop from monsters when they died. These items will have modifiers that increase the player stats when equip, it can also change player fire type to a shotgun like attack where player fire out multiple projectiles at random direction within a cone shape area. The modifiers of the item will have different weighting and be chosen from a pool of modifiers. The chance of a modifier appears depend on its weighting compared to the total weight of all the modifiers in the modifier pool using cummilative probability method.

3. Functional requirements

- Menu screen with play and tutorial and quit game button/
- Player coins earn during an instance must be save when they retreat to hideout (keep 30% coins) or defeat the golem and return to hideout (keep 100% of the coins in that instance)
- When player die all the coins, they earnt in that instance is gone
- Player can't dash over object that can be collided
- Player must get invincibility time when they got hit
- When player or any monsters reach 0 or below hp all their movement and action must stop immediately and run their respective dying animation

- Always spawn a buff when player complete a room
- Player won't get obstruct by monster when in invincible frame but can still collide with objects
- Player can only get a max of 5 slots and can't get more than that
- All status effects on player and monsters must be applied and remove after a certain duration without fail
- When player press "Tab" a stats sheet should pop up and show player their current stats information
- When press "I" key the Inventory UI appear and show all the items the player currently has. Player can interact with the item and put it ino their inventory when staying close to the item and press "E". Player can "right click" item in inventory to equip and "ctrl + right" click to drop the item on the ground if they no longer want to have it in their inventory.
- Item modifiers need to increase player stats according to the modifier description show on the item
- Sound effect get play whenever they nede to and background music being played according to the current scene.

4. Non-functional requirements

4.1. Interface requirements

- Uniform and easy to see the text and button
- Color should fit the theme of the game

4.2. Game balancing requirements

- Player should feel fair and not feeling frustrated when playing
- Hit box should work according to the sprite so that nothing will hit player unfairly
- Golem must summon pillar for player to hide behind during laser phase
- Bringer Of Death should feel fair and not too hard to beat

4.3. Compatibility requirements

- Should be able to run on Windows 10.
- Should be able to run on some the sites that host the game well and perform as well as the PC version

4.4. Quality and efficiency requirements

- No drop in fps (frame per second) during play time
- Any game objects that are no longer of use in the scene should be destroyed to clear up memory
- Sprite and animation shouldn't be blurry and drop in quality during play time.

CHAPTER II: USE CASE

I. User story

Users	User stories	Important nouns	Important verbs	Connections
Player	As a player when an enemy get close to me, I want to melee attack and knock them back far from me	player enemy melee attack	attack get close	player - melee attack enemy - knockback
	As a player I want to move around the instance that I was spawned into	player stage	move	player - stage
	As a Player I want to dash around the stage to escape from enemy	player enemy stage	dash escape	player - enemy enemy - stage
	As a player I want to pick up buffs to empower me	player buff	empower	player - buff
	As a player I want the ability to pause the game so o I can go and do other stuffs	Player game	player pause	player - game
	As a player, I want to shoot out projectiles to kill the enemy in front of me.	enemy player projectiles	shoot kill	player - projectiles enemy - death
	As a player I want the option to quit to main menu	player main menu	quit	player -main menu
	As a player I want to retreat to hideout if I feel like I can't clear the room I'm currently playing in	player hideout	retreat	player - hideout
	As a player I want to interact with interactable object and NPC around me	player interactable object NPC	interact	player - interactable object player - NPC
	As a player when I got hit by an enemy attack, I want to take damage as a punishment	player enemy	take damage got hit	player - enemy
	As a player when my character reach 0 hp or lower I want my character to die	player character	die reach	player - character
	As a player I want to make new save file	player save file	make	player - save file
	As a player I want to save my game progress	player game progress	save	player - game progress
	As a player I want to load my current save	player save	load	player - save

	As a player I want my current character to be restrict of movement if it’s affected by the freeze status effect	player character Status effect	affected restrict	player - character character - status effect

II. Use case models

1. Use case diagrams

1.1. Player

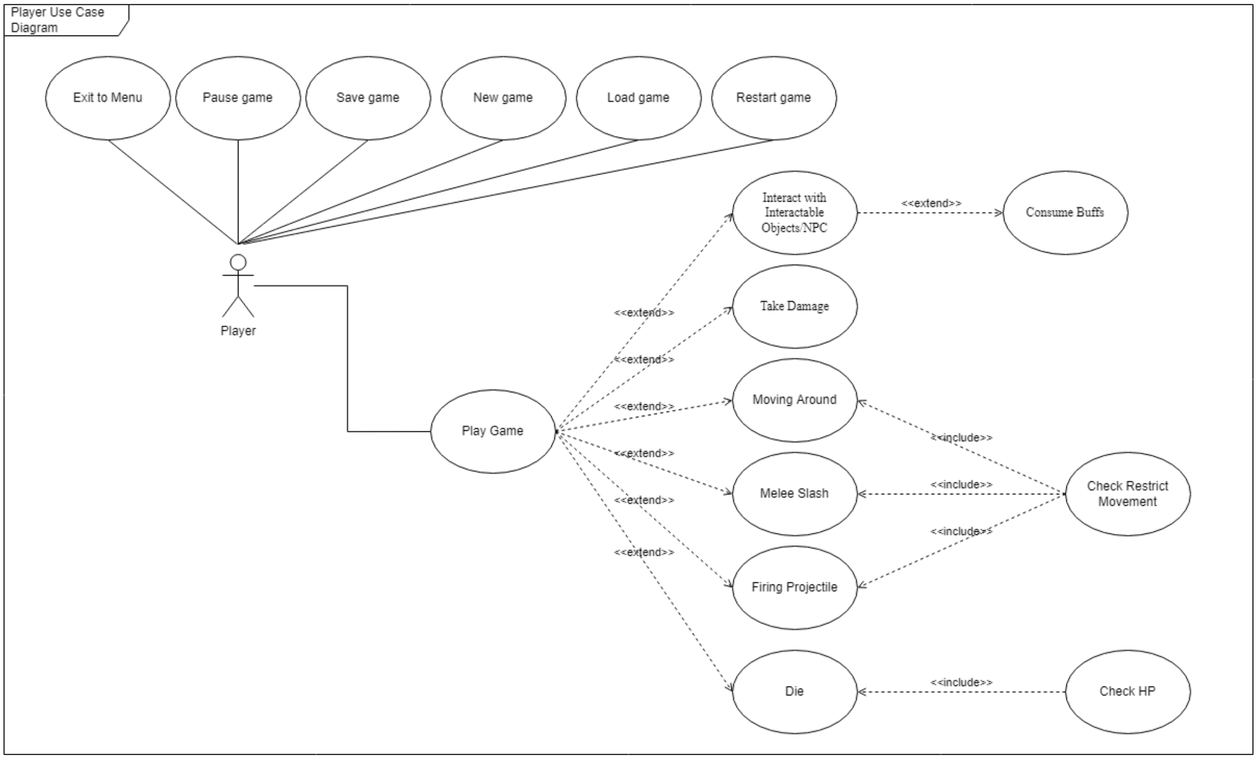


Image 5: Player Use Case Diagram

1.2. Slime

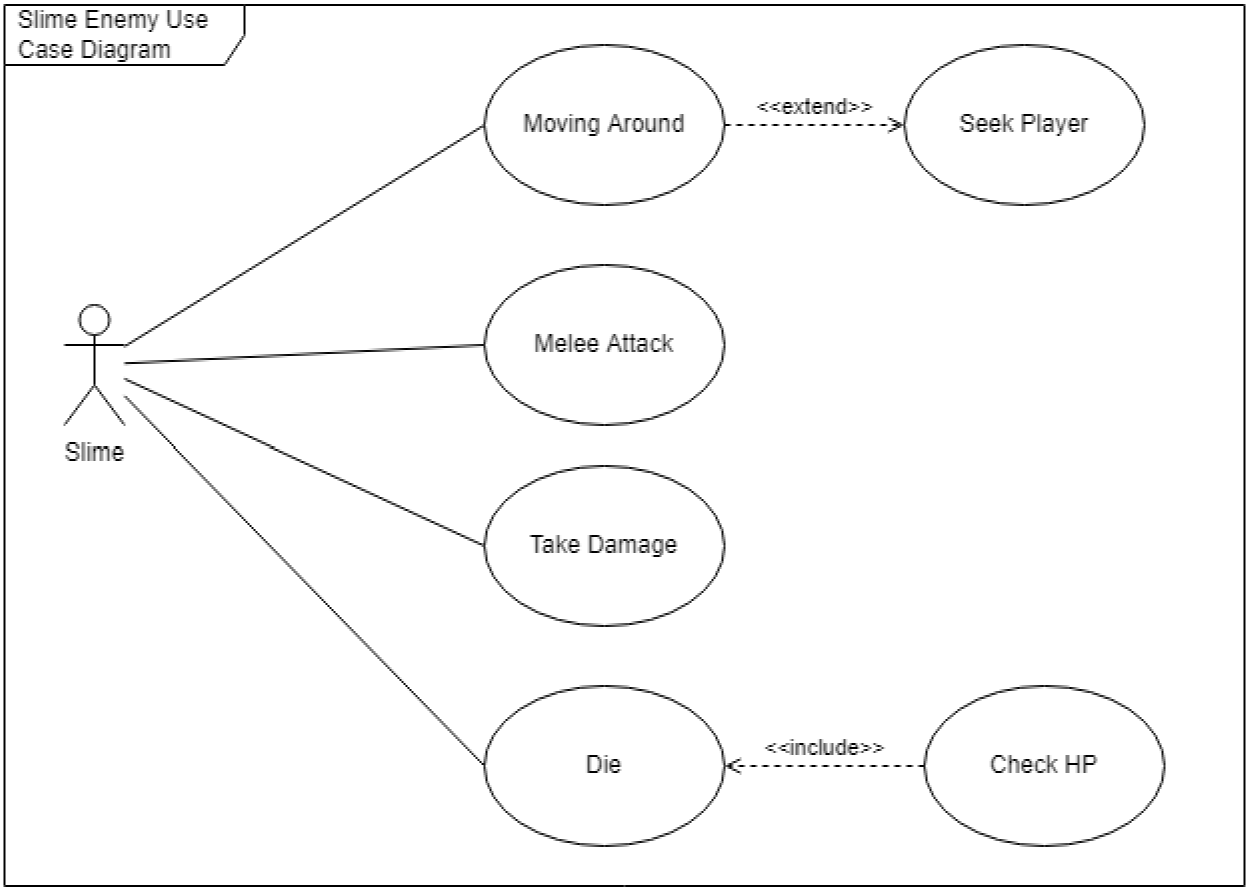


Image 6: Slime Use Case Diagram

1.3. Archer

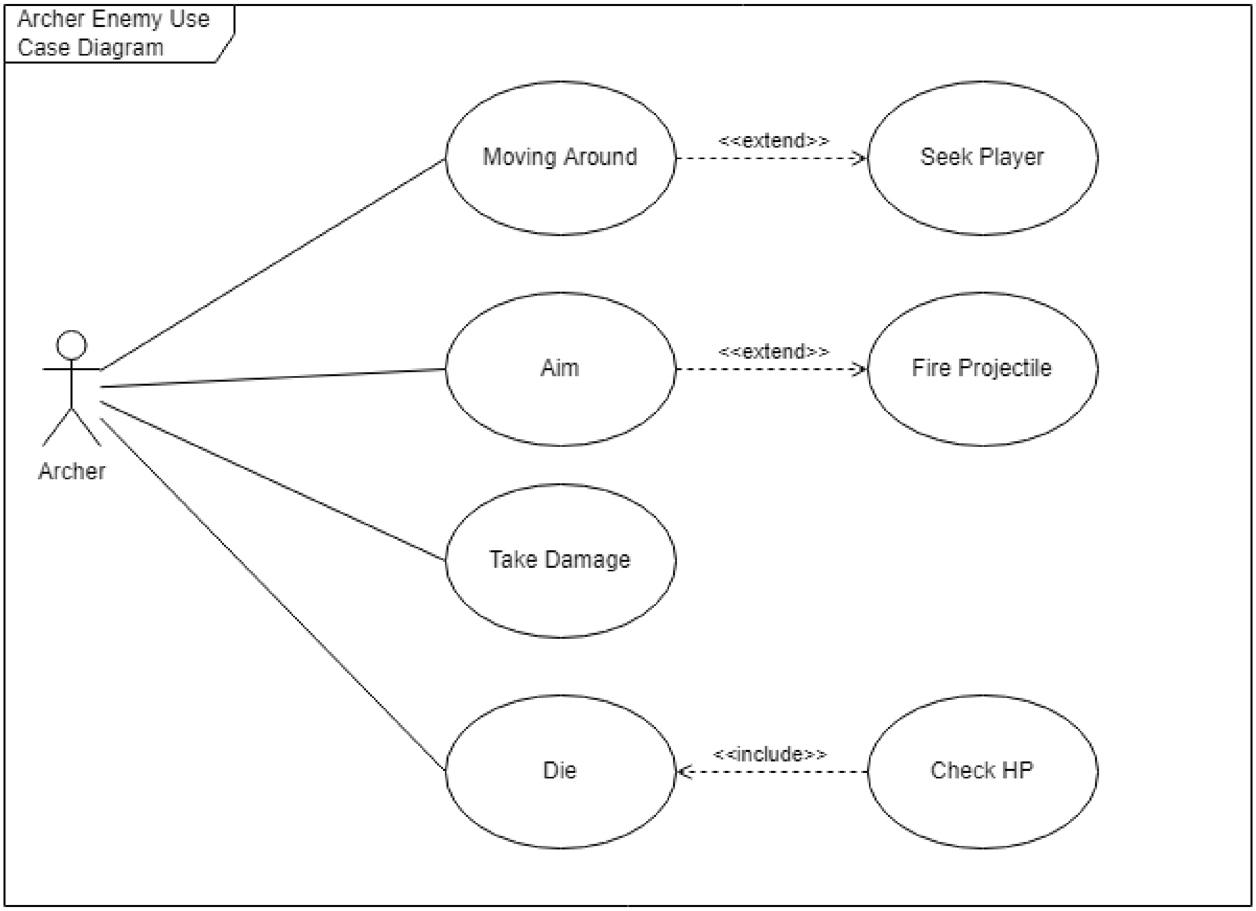


Image 7: Archer Use Case Diagram

1.4. Golem

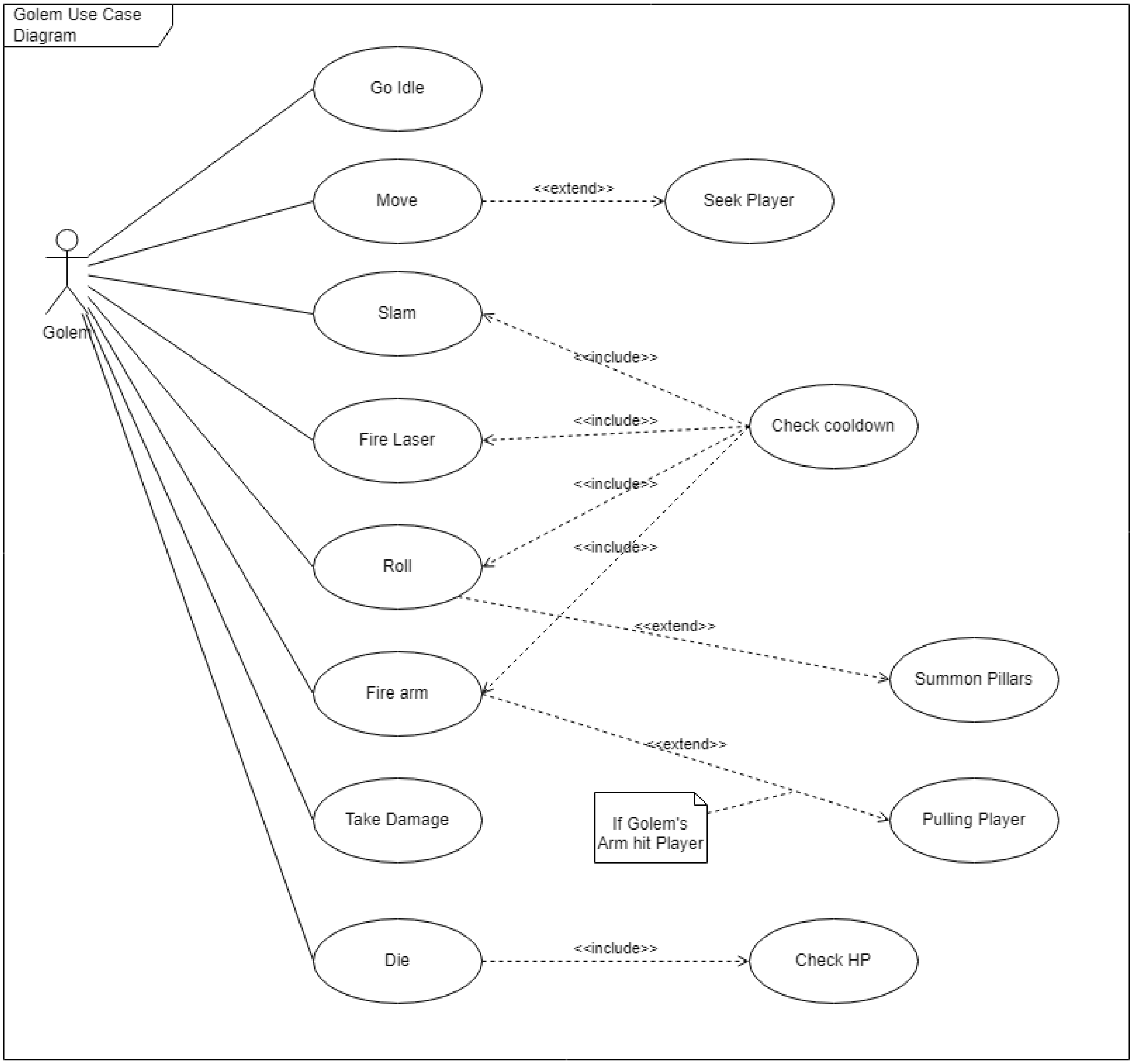


Image 8: Golem Use Case Diagram

1.5. ShopNPC

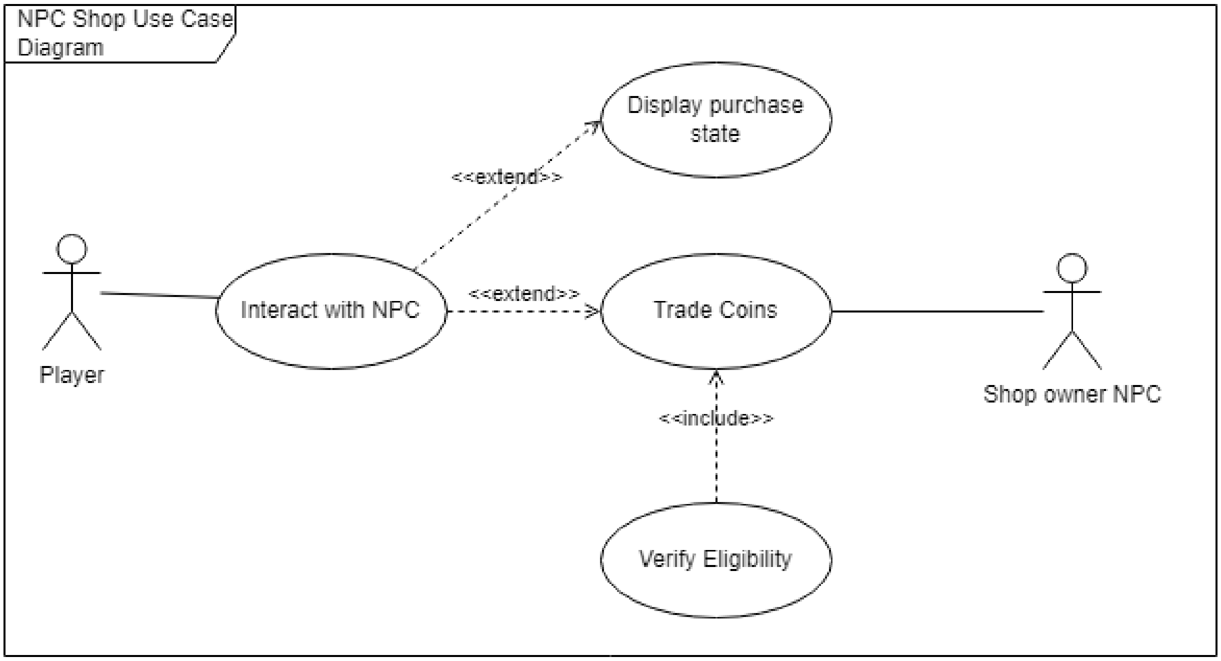


Image 9: Shope NPC Use Case Diagram

1.6. Shield Enemy

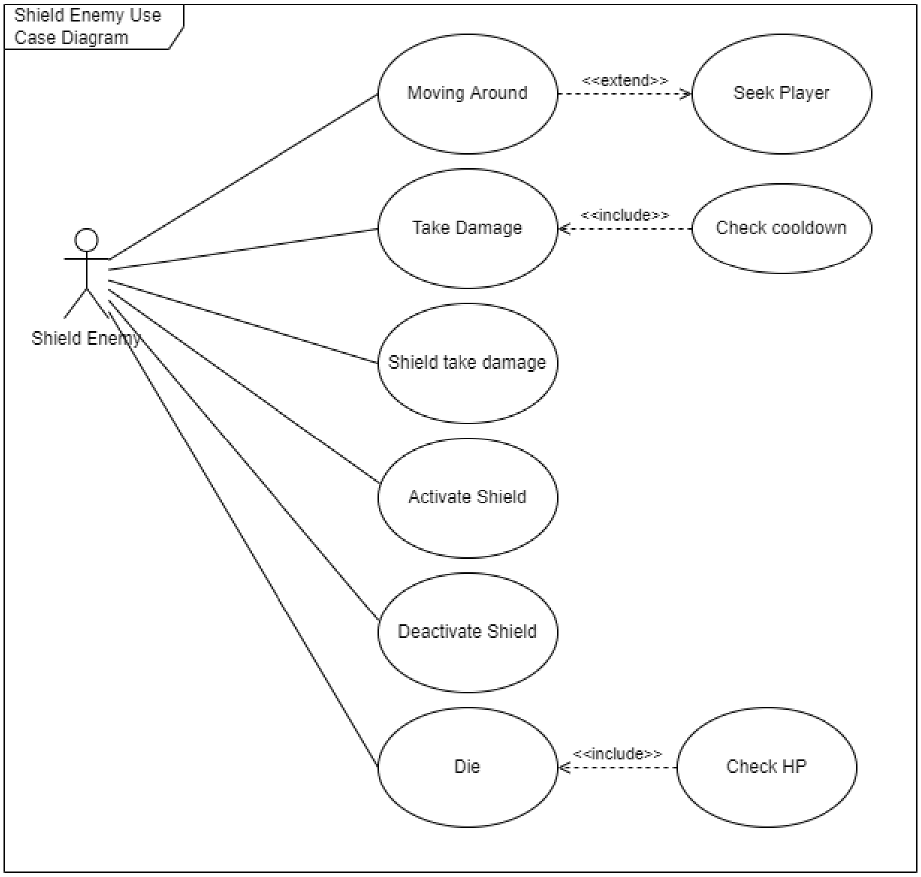


Image 10: Shield Enemy Use Case Diagram

1.7. Bomber

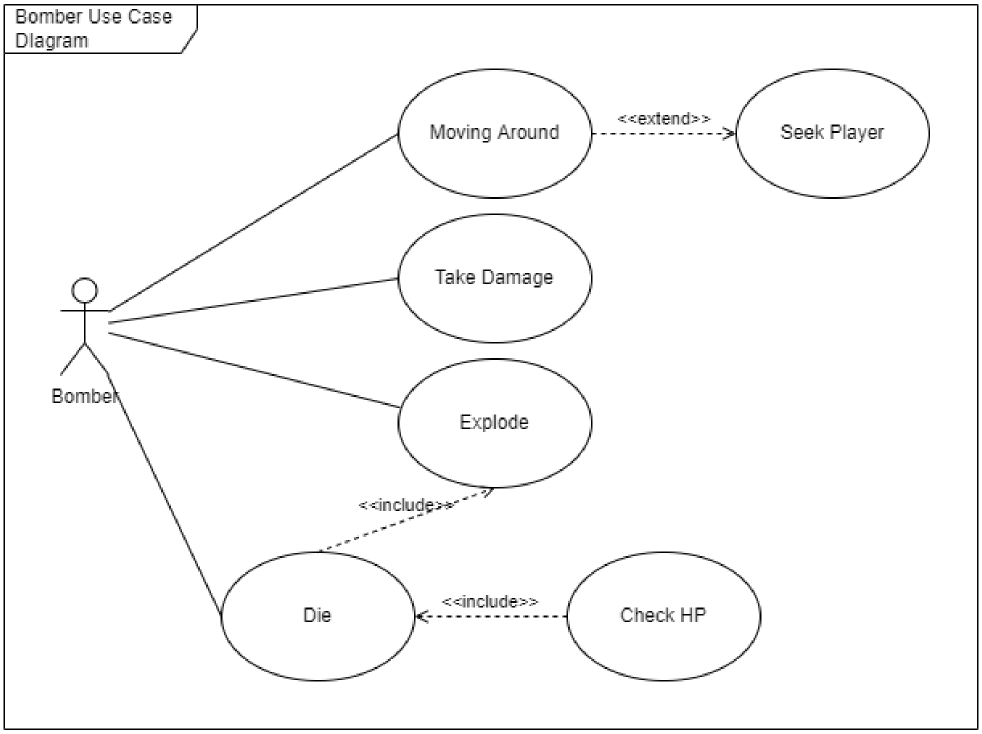


Image 11: Bomber Use Case Diagram

1.8. Audio Manager

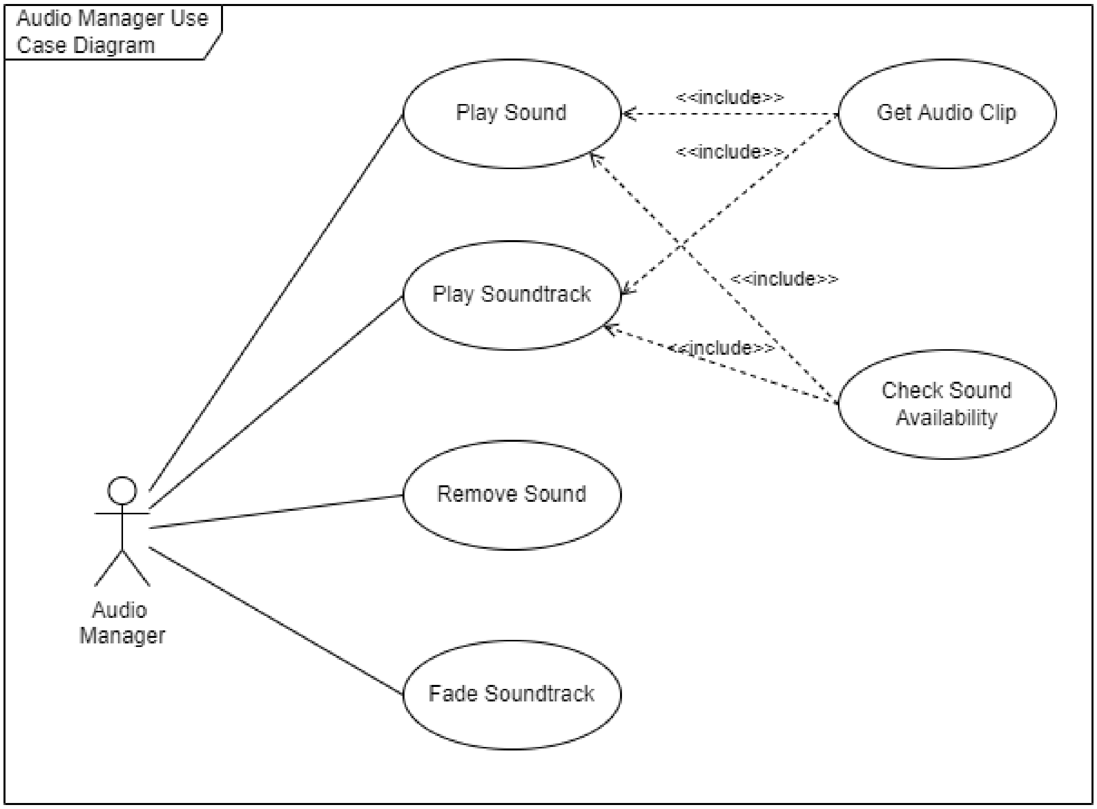


Image 12: Audio Manager Use Case Diagram

1.9. Bringer Of Death Boss

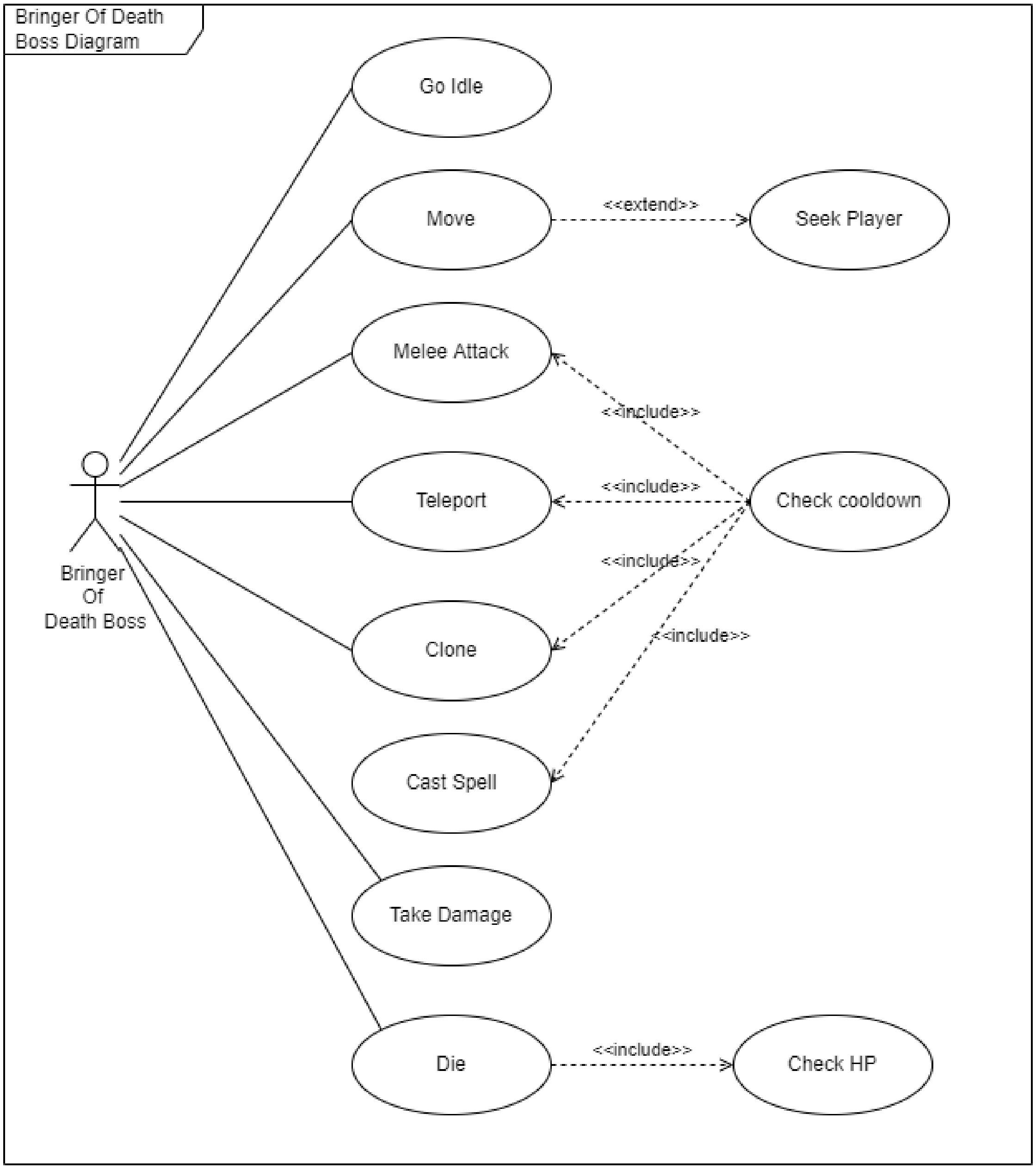


Image 13: Bringer of Death Boss Use Case Diagram

2. Actor list

No.	Actor's name	Brief description / notes
1	Player	Is the person who's currently playing the game. PLayer can shoot, melee slash, dash, moving around the instance, collect coins, drink health potion, consume buffs and so on

2	Slime	Is a melee type enemy that swarms and attacks player. Has high hp and medium speed
3	Archer	Range type enemy that shoots projectile at player. Projectiles can freeze and restrict player movement and action. Has moderate hp and low movement speed
4	Golem	Boss type enemy that deals % max hp damage to player. Can slam, pull player toward it, fire laser, summon pillars, rolling around. Has hight hp, moderate speed and very high damage, it also drop a lot of coins when killed
5	ShopNPC	exchange coins for slots upgrade to player. Slots give player random quirks depending on the number of slots
6	Shield Enemy	Is a melee type enemy that can activate its shield to absorb damage and deal extra damage when its shield is up. They're dangerous up close but they have low speed and mobility so they can be neutralized at range
7	Bomber	Is a memlee type enemy but rather than melee attacking they deal damage to the player by exploding if get close to player or when their healthpoint reach below 0. They're very fast and can deal a lot of damage to player if the player is in their blast radius
8	Bringer of Death Boss	Boss type enemy with fast attack and high speed. The Boss mechanic is simple but can be discruptive and confuse player with its clone. Bringer of Death Boss can cast skill that rain down to player deal a small amount of damage but can apply shock status effect make player take increase damage. Its melee attack is telegraph so player can easilty notice it and dodge in time. Its clone skill and teleport can disrupt player momentum and use that opportunity to strike player down
9	Audio Manager	Play, control and process the sound and soundtrack in the game

3. Use case list

No.	Use case name	Short description / note
1	Play game	The use case begins when player click the "Play" button and load a save file
2	Exit Menu	The use case begins when user presses quit button in the menu

3	Pause game	The use case begins when user pressed pause button
4	Save game	The use case begins when player successfully makes a trade with the NPC , enters an instance, returns to the hideout or return to the main menu screen
5	New game	The use case begins when user presses create button or load button but no saved file is found
6	Load game	The use case begins when user presses load button
7	Restart game	The use case begins when player returns to hideout
8	Interact with interactable objects/NPC	The use case begins when player is in range to interact and presses the interact button
9	Melee Slash	The use case begins when the user is not in the main menu screen and presses the melee slash button (right click)
10	Firing projectiles	The use case begins when the user is not in the main menu screen and presses the fire buttons. This use case also triggers when archer enemy type finished aiming
11	Consume buffs	The use case begins when player enter a new instance or interacts with a buff
12	Check restrict movement	The use case begins when the player are trying to move around or perform an action while affected by freezing status effect or being pull by golem arm
13	Check HP	The use case begins when the entity is instantiated and still active. This use case also stay active throughout the player active play time
14	Die	The use case begins when the player's character hp or monster's hp reaches 0 or below
15	Take damage	The use case begins when the player or monster takes damage
16	Moving around	The use case begins when the AI selects a destination for the entity to move to. It also begin when the user is not in the main menu screen and presses either the arrows button or wasd buttons. It's being used by player to move around the instance and used by monsters to get close to player to perform their attacks
17	Melee Attack	The use case begins when the player is in the enemy entity's attack range and the entity perform a melee attack toward the player direction
18	Seek Player	The use case begins in the process of selecting destination for the enemy entity to lead to player current position. This use case is active throughout the life duration of an enemy
19	Go idle	The use case begins when golem need to switching state, phase or changing up an attack. When active all other golem actions stop.

20	Slam	The use case begins when golem is in its first stage, after it has done the fire arm action
21	Fire laser	The use case begins when the fire laser isn't on cooldown and the laser has finished summoning pillars
22	Roll	The use case begins when roll isn't on cooldown
23	Fire arm	The use case begins when firing arm isn't on cool down
24	Check Cooldown	The use case begins when golem perform any actions pertaining cooldown
25	Summon Pillars	The use case begins when golem has finished rolling
26	Pulling player	The use case begins when golem has fired its arm and it hits the player
27	Aim	The use case begins when the player is in range of any monster that can fire a projectile
28	Interact with NPC	The use case begins when the player is in the range to interact and presses the interact button
29	Display purchase state	The use case begins when the player chooses a buy option or in the transition from the first to the second dialogue
30	Trade coins	The use case begins when the player has interacted with the NPC
31	Verify Eligibility	The use case begins when the player has chosen a buy option
32	Explode	The use case begins when the bomber enemy get close enough to the player to explode
33	Shield Take Damage	The use case begins when the Shield Enemy type activate their shield and they got hit with a projectile or melee attack. The shield will take the damage instead of the Shield Enemy
34	Activate Shield	The use case begins when the Shield Enemy type has its shield come off cooldown
36	Deactivate Shield	The use case begins when the Shield Enemy’s Shield got its hp below 0
36	Teleport	The use case begins when the Bringer of Death Boss teleport skill is off cooldown
37	Clone	The use case begins when the Bringer of Death Boss clone skill is off cooldown
38	Cast Spell	The use case begins when the Bringer of Death Boss spell skill is off cooldown
39	Play Sound	The use case begins whenever an object wants to play a sound effect
40	Play Soundtrack	The use case begins whenever a soundtrack need to be played in a scene
41	Remove Sound	The use case begins when an object that container game object of sound or soundtrack need to be destroy to remove the sound from

		playing
42	Fade Soundtrack	The use case begins when there is a need to fadeout current soundtrack and play another soundtrack if specified
43	Get Audio Clip	The use case begins whenever a sound effect or soundtrack need to be played so it can get the sound effect or sountrack needed from a list
44	Check Sound Availability	The use case begins whenever a sound effect or soundtrack need to be played. It's used to check if the sound effect or soundtrack is available or not

4. Use case specification

4.1. Play game

Short description		The use case begin when player click the "Play" button and load a save file
Flow of events	**Basic flow**	1. Player click "Play" button in the main menu 2. Player load or create a new save file 3. Player enter the hideout and enter to an instance by interacting with the portal
	Alternative flow	none.
Special requirements		none.
Pre-condition		- The game is on
Post-condition		- Player playing the game
Extend Points		- Use case "Interact with Interactable Objects/NPC": player interact with intractable things around them - Use case "Take Damage": player take damage when hit - Use case "Moving Around": player move around the instance or hideout - Use case "Melee Slash": player slash at monsters and knock them back - Use case "Firing Projectiles": player shoot out projectiles to kill monsters - Use case "Die": player's character die after reaching 0 hp

4.2. Exit menu

Short description	Player exit to main menu

Flow of events	**Basic flow**	1. Player press esc key 2. A pop up appear 3. Player click the "exit to menu" button
	Alternative flow	3. Press the esc key again to close the pop up window
Special requirements		none.
Pre-condition		- Player is in hideout
Post-condition		- Player get back to the main menu
Extend Points		none.

4.3. Pause game

Short description		Player pause the game
Flow of events	**Basic flow**	1. Player enter the game after clicking the play button in main menu 2. Player click "load" button on a save slot 3. Player press esc to pause the game 4. Press esc again to unpause the game
	Alternative flow	none.
Special requirements		none.
Pre-condition		- Player is in an instance or in the hideout
Post-condition		- The game is paused
Extend Points		none.

4.4. Save game

Short description		The use case begins when player successfully makes a trade with the NPC , enters an instance, returns to the hideout or return to the main menu screen
Flow of events	**Basic flow**	1. Player enter the game after clicking the play button in main menu 2. Player click "load" button on a save slot 3. Player enter an instance by interacting with the portal

	Alternative flow	Alternative 1: 3. Player interact with the npc and buy a slot Alternative 2: 3. Player enter an instance by interacting with the portal 4. Player retreat to hideout
Special requirements		none.
Pre-condition		- Player is in an instance or in the hideout - Player character isn’t dead
Post-condition		- Player coins and slots are saved
Extend Points		none.

4.5. New game

Short description		The use case begins when user presses create button or load button but no saved file is found
Flow of events	**Basic flow**	1. Player enter the game after clicking the play button in main menu 2. Player click “load” button on a save slot
	Alternative flow	2. Player click “new‟ button on a save slot.
Special requirements		None.
Pre-condition		- If click “load” button the save slot need to be empty
Post-condition		- New game save file is created or overwritten (if click “new” button on an already exist save file)
Extend Points		none.

4.6. Load game

Short description		The use case begins when user presses create button or load button but no saved file is found
Flow of events	**Basic flow**	1. Player enter the game after clicking the play button in main menu 2. Player click “load” button on a save slot
	Alternative flow	none.

Special requirements	none.
Pre-condition	none.
Post-condition	- Load the save file player chosen
Extend Points	none.

4.7. Restart game

Short description		The use case begins when player returns to hideout
Flow of events	**Basic flow**	1. Player enter an instance by interacting with the portal 2. Player fight the monster but feel like they can’t continue 3. Player press esc key to pause the game and the pause menu appear 4. Player press the “Retreat to hideout” button 5. A pop up appear to warn player that they only keep 30% of the coins they found 6. Player click “yes” button 7. Player return to hideout
	Alternative flow	6. Player click “no” button 7. Return to the pause menu
Special requirements		none.
Pre-condition		- Player is in an instance
Post-condition		- Player return to hideout and only keep 30% of the coins they earnt during the previous instance
Extend Points		none.

4.8. Interact with interactable objects/NPC

Short description		The use case begins when player is in range to interact and presses the interact button
Flow of events	**Basic flow**	1. Player enter the hideout 2. Player get close to the NPC or Interactable Object until an “E” key pressing animation appeared on top of player 3. Player press E and interact with the NPC or Interactable Object
	Alternative flow	none.

Special requirements	none.
Pre-condition	- Player is close to the NPC or Interactable Object.
Post-condition	- Player interact with the NPC or Interactable Object
Extend Points	none.

4.9. Melee Slash

Short description		The use case begins when the user is not in the main menu screen and presses the melee slash button
Flow of events	**Basic flow**	1. Player enter an instance and start playing the game 2. An enemy get close to the player 3. Player right click to perform a melee slash attack 4. The monster is knockback
	Alternative flow	none.
Special requirements		none.
Pre-condition		- Player movement and action are not restricted
Post-condition		- Monsters in player melee slash range are knock back
Extend Points		none.

4.10. Firing projectiles

Short description		The use case begins when the user is not in the main menu screen and presses the fire buttons. This use case also trigger when archer enemy type finished aiming
Flow of events	**Basic flow**	1. Player enter an instance and start playing the game 2. Player see monsters and left click to fire projectile at their current cursor position 3. Projectile coming out from player's character gun barrel
	Alternative flow	none.
Special requirements		none.
Pre-condition		- Player movement and action are not restricted

Post-condition	- A Projectile shoot out from player barrel
Extend Points	none.

4.11. Consume buffs

Short description		The use case begins when player enter a new instance or interacts with a buff
Flow of events	**Basic flow**	1. Player enter an instance and start playing the game 2. After clearing a room a random buff spawn depending on their weight 3. Player get close to the buff until the interact icon show up 4. Player press the interact key and consume the buff 5. The buff disappear and apply its buff to player stats
	Alternative flow	none.
Special requirements		none.
Pre-condition		- Player movement and action are not restricted - Player are near the buff enough so that the interact icon show up
Post-condition		- The buff disappear and it apply the buff to the player
Extend Points		none.

4.12. Check restrict movement

Short description		The use case begins when the player has pressed a combination of move buttons
Flow of events	**Basic flow**	1. Player enter an instance and start playing the game 2. Player get hit by either an Archer's projectiles and getting freeze or getting pull by golem arm 3. Player try different inputs but nothing go through 4. After restrict movement is lifted player can move and perform action again
	Alternative flow	none.
Special requirements		none.
Pre-condition		- Player movement and action are not already restricted

Post-condition	- Player can’t move or perform action
Extend Points	none.

4.13. Check HP

Short description		The use case begins when the entity is instantiated and still active. This use case also stay active throughout the player active play time
Flow of events	**Basic flow**	1. An entity is instantiated 2. During the entity active time check entity health points every interval using either Update of FixedUpdate function from Unity
	Alternative flow	Entity is a golem: 3. Golem hp are below a certain threshold 4. Switch state or phase depending on the threshold condition Entity’s hp are lower or equal to 0 3. Switch entity animation to die animation 4. Destroy the gameobject that represent the entity
Special requirements		None.
Pre-condition		- Entity is instantiated and active
Post-condition		None.
Extend Points		None.

4.14. Die

Short description		The use case begins when the player's character hp or monster's hp reaches 0 or below. It run the player’s character die animation or monster’s die animation if there are any then destroy the game object that represent either player’s character or monster
Flow of events	**Basic flow**	1. Player’s character or monster’s hp reach 0 2. Play the respective dying animation for the player’s character or monster 3. Destroy the game object that represent player’s character or monster at the end of the animation
	Alternative	None.

	flow	
Special requirements		None.
Pre-condition		- Player’s character hp or monster’s hp reach 0.
Post-condition		- Destroy game object that represent the entity that reach 0 hp
Extend Points		None.

4.15. Take damage

Short description		The use case begins when the player or monster takes damage
Flow of events	Basic flow	1. Entity's rigid body detect a collision from the player’s attack 2. Entity's health point is reduced according to damage inflicted
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Entity is alive and active
Post-condition		- Entity hp are reduce according to the damage taken and reacts to it accordingly
Extend Points		None.

4.16. Moving around

Short description		Entity moves around the instance
Flow of events	Basic flow	1. Entity checks the distance from its standing point to player 2. Entity moves toward the player 3. Entity run animation activates
	Alternative flow	Player isn’t in Entity's range 1. Entity checks the distance from its standing point to player 2. Entity moves to a random destination

		3. Entity's run animation activates If Entity is player 1. Detect directional input from player 2. Move the player’s character according to player movement speed 3. Run the running animation for player’s character
Special requirements		None.
Pre-condition		- Entity is alive and active - If Entity is player then player’s character doesn’t affected by any kind restrict movement
Post-condition		- Entity's velocity is larger than 0
Extend Points		None.

4.17. Melee Attack

Short description		When the player is in a melee type enemy attack range that enemy perform a melee attack
Flow of events	**Basic flow**	1. Entity's attack animation activates 2. Increase the size of its collider 3. Player takes damages 4. Return the size of its collider to normal
	Alternative flow	The attack doesn’t hit player 1. Entity's attack animation activates 2. Increase the size of its collider 3. return the size of its collider to normal
Special requirements		None.
Pre-condition		- Entity is alive and active
Post-condition		- Entity perform a melee attack.
Extend Points		None.

4.18. Seek Player

Short description	The process of finding a path from one to the player

Flow of events	**Basic flow**	1. Register the position of the entity and the player 2. find the shortest path from the entity to the player without clashing with other objects
	Alternative flow	None.
Special requirements		- A* pathfinding algorithm
Pre-condition		- The entity and the player must be in the same scene, and is alive and active
Post-condition		None.
Extend Points		None.

4.19. Go idle

Short description		Entity changes its state to idle
Flow of events	**Basic flow**	1. Entity can't do any other action 2. Entity changes its state to idle 3. Activate Entity's idle animation
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Entity is still alive and active; - Golem is in its first stage or in its second stage and is in post firing arm actions and slam action(for golem only)
Post-condition		None.
Extend Points		None.

4.20. Slam

Short description		Slam attack perform by Golem Boss
Flow of events	**Basic flow**	1. Golem's slam animation activates 2. Instantiate slam effect
	Alternative	Golem is too far from the player (only in fist stage)

	flow	1. golem checks the distance from its standing point to player 2. Golem moves toward player
Special requirements		None.
Pre-condition		- Golem is alive ,active and the slam action isn’t on cooldown
Post-condition		None.
Extend Points		None.

4.21. Fire laser

Short description		Golem stays still and shoot a laser beam sweeping from right to left
Flow of events	Basic flow	1. Golem’s laser animation activates 2. Instantiate laser object 3. laser start from the right side of the golem and move toward the left side 4. halt a bit upon reaching the left side and come back to the right side 5. deactivate the laser object when the beam has reached the right side
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Golem is in 2nd phase ,still active ,the laser action isn’t on cooldown and - The has already summoned pillars
Post-condition		None.
Extend Points		None.

4.22. Roll

Short description	Golem rolls

Flow of events	**Basic flow**	1. golem's fold animation activates 2. Golem speed is increased 3. Golem moves to all waypoints 4. Golem's unfold animation activate 5. Golem speed returns to its base
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Golem is alive ,active , in its 2nd phase and the roll action isn't on cooldown
Post-condition		None.
Extend Points		None.

4.23. Fire arm

Short description		Golem Fire its arm at Player pulling Player toward its position then proceed to slam the Player
Flow of events	**Basic flow**	1. golem fires its arm to pull the player back to its position 2. pull the player back to the golem 3. Golem's slam animation activates 4. Instantiate slam effect
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Golem is alive and active - Golem is in its second phase - Fire arm isn't on cooldown
Post-condition		- Golem's arm shoot out toward the player at high speed.
Extend Points		Use case "Pulling player".

4.24. Check Cooldown

Short description	Check the cooldown of an action and return a flag indicating if if it is still on cooldown or not

Flow of events	**Basic flow**	1. Start cooldown timer 2. return the flag 3. Reset the timer
	Alternative flow	None.
Special requirements		None.
Pre-condition		- There is a skill on cooldown.
Post-condition		- Return the cooldown state of that skill.
Extend Points		None.

4.25. Summon Pillars

Short description		Golem stays still and summons a set a pillar
Flow of events	**Basic flow**	1. Activate summon animation 2. instantiate pillars for each waypoint there are in the arena
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Golem is in 2nd phase ,still active. - Has already done the rolled action
Post-condition		- Pillars are summoned from the ground for player to hide behind
Extend Points		None.

4.26. Pulling player

Short description		The arm that the golem shoots out and grab the player back to near the golem
Flow of events	**Basic flow**	1. Golem fire out its arm 2. The arm hit player 3. move the player position next to the golem
	Alternative	None.

	flow	
Special requirements		None.
Pre-condition		- The Golem has shot its arm and it hits the player
Post-condition		- Player are pulled toward golem's position
Extend Points		None.

4.27. Aim

Short description		Archer aims
Flow of events	**Basic flow**	1. Archer aim animation activates 2. Archer stay still and calculate the fire direction 3. Archer fires projectiles
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Archer is alive and active - Player is in archer's attack range
Post-condition		- A projectile shoot out from Archer.
Extend Points		None.

4.28. Interact with NPC

Short description		Player interacts with the NPC
Flow of events	**Basic flow**	1. Player presses interact button
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Player is active - The player is in the range to interact - The object the player interacts with must be interactable

Post-condition	None.
Extend Points	Use case "Display purchase state". Use case "Trade coins"

4.29. Display purchase state

Short description		This use-case is used for the shop owner NPC to show the buy options to the player
Flow of events	**Basic flow**	1. Player interacts with the shop owner to buy unlockable slot 2. The shop owner NPC checks if player can still buy more unlockable slots 3. The shop owner shows the player buy options 4. The shop owner shows if the trade is success or not
	Alternative flow	Player has reached the maximum number of slot that can be unlocked 3. The shop owner alerts the player to not buy more unlockable slots.
Special requirements		None.
Pre-condition		- Player is active - The number of player's current unlockable slots isn't negative
Post-condition		- Display NPC Shop UI and dialogue.
Extend Points		None.

4.30. Trade coins

Short description		Player trade in the coin for unlockable slots
Flow of events	**Basic flow**	1. Player chooses a buy option 2. Player's coin is reduced 3. Player unlocks new slot
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Player meet the eligibility criteria

Post-condition	- Player's coin is reduced accordingly to the purchase - Player unlock a slot
Extend Points	None.

4.31. Verify Eligibility

Short description		This use-case is used for the shop owner NPC to check if the player can buy or not
Flow of events	**Basic flow**	1. Player interacts with the shop owner to buy unlockable slot 2. The shop owner NPC show options to the player 3. The player chooses the available option 4. The shop owner NPC checks if the player has enough coins or already maxed out slots 5. The shop owner notify user on their eligibility
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Player interact with the ShopNPC
Post-condition		- Return player eligibility.
Extend Points		None.

4.32. Explode

Short description		This use-case is used for the Bomber enemy so they can explode and deal massive daamge to player if the player is in the blast radius
Flow of events	**Basic flow**	1. Player gets close to the range of attack of Bomber enemy or Bomber enemy is dying 2. Bomber enemy plays the explode animation 3. Bomber enemy explode and create a fire explosion 4. Player is caught in the explosion and take big damage
	Alternative flow	4. Player is not in the explosion radius and take no damage
Special requirements		- None
Pre-condition		- Bomber enemy is dying or player in range of Bomber enemy

		attack
Post-condition		- Player takes damage if in the blast radius - Player doesn't take damage if not caught in the blast radius
Extend Points		

4.33. Shield Take Damage

Short description		This use-case is used for the Shield Enemy so their shield can take the damage instead
Flow of events	**Basic flow**	1. Shield Enemy activate its shield 2. The Shield Enemy got hit by an attack 3. The Shield Enemy's shield health point is reduced by the amount of damage taken
	Alternative flow	None
Special requirements		- None
Pre-condition		- Shield Enemy Activate their shield - Shield Enemy is hit by an attack
Post-condition		- Shield Enemy's shield health point is reduce by the amount of damage taken
Extend Points		None.

4.34. Activate Shield

Short description		This use-scase is used by Shield Enemy to activate their shield
Flow of events	**Basic flow**	1. Shield Enemy's shield is off cooldown 2. Shield Enemy Activate their shield 3. Shield Enemy's shield is up and Shield Enemy get increased damage and protection from damage whe their shield is up
	Alternative flow	None.
Special requirements		- None.
Pre-condition		- Shield Enemy's shield is off cooldown
Post-condition		- Shield Enemy's shield is activated

Extend Points	None.

4.35. Deactivate Shield

Short description		This use-case is ued by the Shield Enemy to deactivate their shield when their shield health point is equal or below 0
Flow of events	**Basic flow**	1. Shield Enemy's shield got damage and its health point is below 0 2. Shield Enemy's shield get deactivated 3. Shield Enemy lost protection from damage and damage increase
	Alternative flow	None.
Special requirements		- None.
Pre-condition		- Shield Enemy's shield health point is equal or below 0
Post-condition		- Shield Enemy's shield is deactivated and go on cooldown
Extend Points		None.

4.36. Teleport

Short description		This use-case is used by the Bringer of Death Boss to teleport to player position with an offset depend on the Boss width and height
Flow of events	**Basic flow**	1. Bringer of Death Boss's teleport skill is off cooldown 2. Bringer of Death Boss cast the teleport skill 3. Bringer of Death Boss is invulnerable and go in its portal 4. Bringer of Death Boss re-appear at the player position
	Alternative flow	None.
Special requirements		- None.
Pre-condition		- Bringer of Death Boss's teleport skill is off cooldown
Post-condition		- Brginer of Death Boss is teleported to the player position
Extend Points		None.

4.37. Clone

Short description	This use-case is used by the Bringer of Death Boss to clone itself and to discrupt player

Flow of events	**Basic flow**	1. Bringer of Death Boss's clone skill is off cooldown 2. Bringer of Death Boss cast the clone skill 3. Four Bringer of Death Boss clones appear at player position and melee attack player 4. Bringer of Death Clone die after a certain amount of time
	Alternative flow	None.
Special requirements		- Bringer of Death clones need to die after amount of time
Pre-condition		- Bringer of Death's clone skill is off cooldown
Post-condition		- Bringer of Death clones itself
Extend Points		None.

4.38. Cast Spell

Short description		This use-case is used by Bringer of Death Boss to rain down spell attack on player
Flow of events	**Basic flow**	1. Bringer of Death Boss's spell skill is off cooldown 2. Bringer of Death Boss cast the spell skill 3. Portals appear on top the head of player and rain down attack
	Alternative flow	None.
Special requirements		None.
Pre-condition		- Bringer of Death's spell is off cooldown
Post-condition		- Bringer of Death's spells rain down and attack player
Extend Points		None.

4.39. Play Sound

Short description		This use-case is used by Audio Manager to play sound effect
Flow of events	**Basic flow**	1. A game object calls the play sound function from the Audio Manager instance 2. Audio Manager find the sound needed to play 3. Audio Manager play the sound
	Alternative flow	3. Audio Manager can't find the sound 4. Audio Manager display an error message saying specific

		sound is not available or missing
Special requirements		None.
Pre-condition		- Audio manager can find the sound needed to play
Post-condition		- A sound is played by the Audio Manager instance
Extend Points		None

4.40. Play Soundtrack

Short description		This use-case is used by Audio Manager to play soundtrack
Flow of events	**Basic flow**	1. A game object calls the play sountrack function from the Audio Manager instance 2. Audio Manager find the soundtrack needed to play 3. Audio Manager play the soundtrack
	Alternative flow	3. Audio Manager can't find the sound 4. Audio Manager display an error message saying specific soundtrack is not available or missing
Special requirements		None.
Pre-condition		- Audio manager can find the soundtrack needed to play
Post-condition		- A soundtrack is played by the Audio Manager instance
Extend Points		None.

4.41. Remove Sound

Short description		This use-case is used by the Audio Manager to remove a specific game object container of a sound and stop it from playing
Flow of events	**Basic flow**	1. Audio Manager find the object with the sound needed to be removed 2. Audio Manager remove the game object that contain the specific sound
	Alternative flow	2. Audio Manager can't find the game object that contain that need to be removed 3. Audio manager stop the function
Special requirements		None.
Pre-condition		- The sound needed to be removed is found

Post-condition		- The game object that contains specific sound that needed to be removed is destroyed
Extend Points		None.

4.42. Fade Soundtrack

Short description		This use-case is used by Audio Manager to fade out the current soundtrack and play another soundtrack if specified
Flow of events	**Basic flow**	1. Audio Manager find current playing soundtrack 2. Audio Manager fade out the soundtrack over time until the soundtrack volume reach 0
	Alternative flow	3. Audio Manager play another soundtrack
Special requirements		None.
Pre-condition		- None.
Post-condition		- Current soundtrack is fade out and new soundtrack is played if specified
Extend Points		None.

4.43. Get Audio Clip

Short description		This use-case is used by Audio Manager to get specific audio clip to play it
Flow of events	**Basic flow**	1. Audio Manager find the needed audio clip to play 2. Audio Manager the audio clip and assign it to a game object to play the sound
	Alternative flow	2. The Audio Manager can't find the audio clip 3. The Audio Manager display an error saying the specified audio clip cannot be found
Special requirements		None.
Pre-condition		- Audio Clip is found by Audio Manager
Post-condition		- Audio Manager get the audio clip and process it
Extend Points		None.

4.44. Check Sound Availability

Short description		This use-case is used by Audio Manager to check if a sound is available to play
Flow of events	**Basic flow**	1. Audio Manager find the audio clip 2. Audio Manager check the audio clip 3. Return the availability of the specified audio clip
	Alternative flow	None.
Special requirements		None.
Pre-condition		- None.
Post-condition		- Availability of specified audio clip
Extend Points		None.

CHAPTER III: ANALYSIS DESIGN

I. Classes Analysis

1. Class diagrams

1.1. Player and Enemy class Diagram

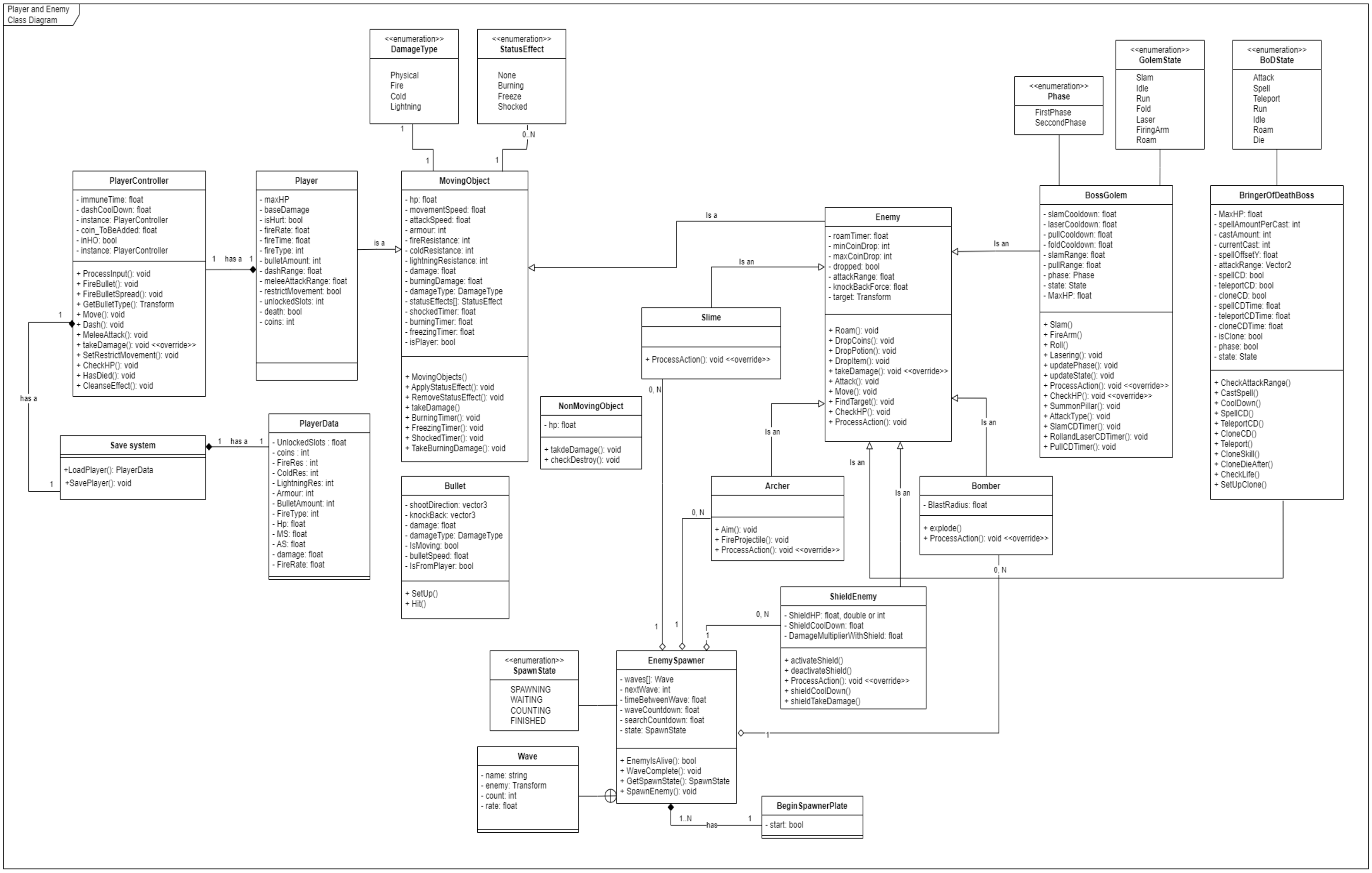


Image 14: Player and Enemies Class Diagram

1.2. Interactable Class Diagram

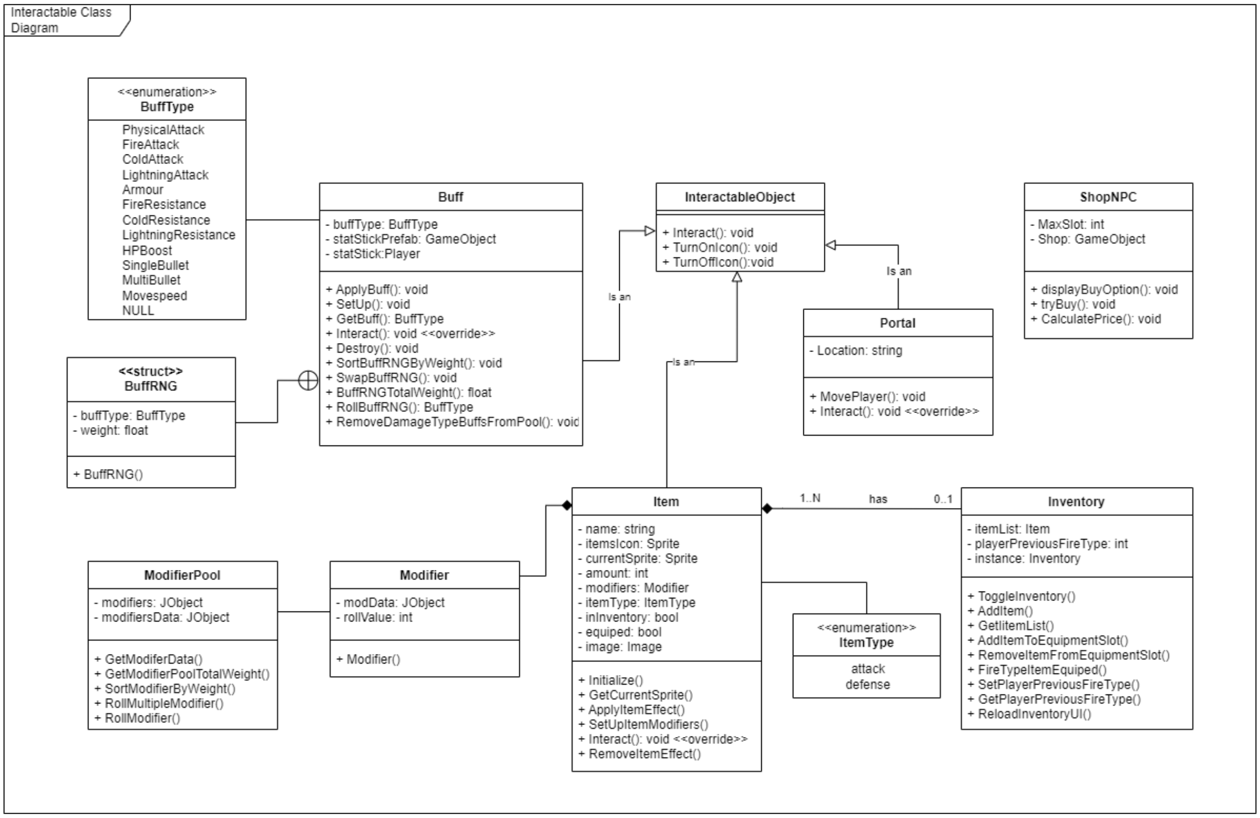


Image 15: Interactable Class Diagram

1.3. Menu and Managers Class Diagram

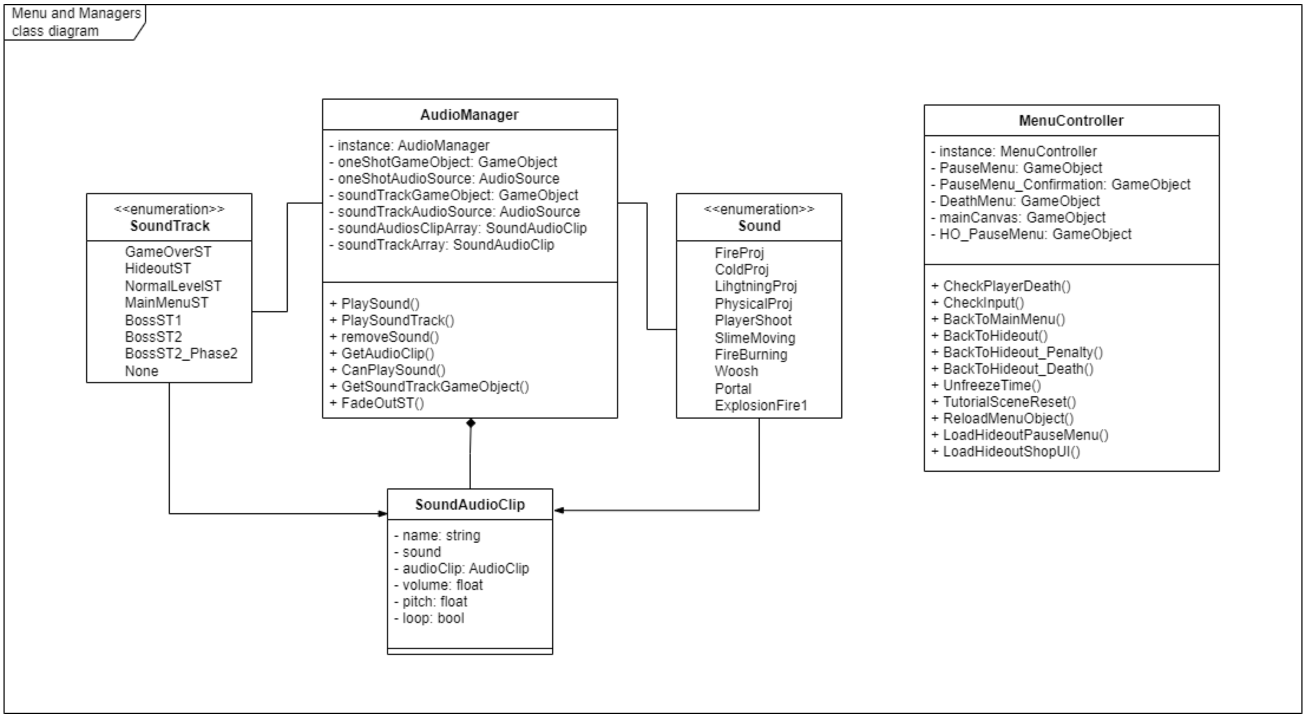


Image 16: Menu and Managers Class Diagram

2. List of classes and relationships

2.1. Classes

No.	Class Name	Type	Note
1	MovingObject	Normal Class	Every moving Player and Enemy parent class
2	DamageType	Enum	Damage type every Moving Object has
3	StatusEffect	Enum	Status effect that affect every Moving Object
4	Player	Normal Class	Player class
5	PlayerController	Normal Class	Controller for player class, processing player input and action
6	Player Data	Normal Class	Use to save player data to a save file
7	Save system	Normal Class	Save system
8	NonMovingObject	Normal Class	Any Object that don't perform an action in the game
9	Enemy	Normal Class	Parent class for enemy type
10	Slime	Normal Class	Melee enemy type
11	Archer	Normal Class	Range enemy type
12	BossGolem	Normal Class	Boss enemy type
13	Phase	Enum	Phase golem use to perform attacks
14	State	Enum	State golem use to change animation and process next action
15	EnemySpawner	Normal Class	Spawn enemy in an instance
16	BeginSpawnerPlate	Normal Class	Use to trigger spawner to spawn enemies
17	SpawnState	Enum	Spawner state to know which state the spawner currently in i.e., SPAWNING, FINISHED...
18	Wave	Normal Class	Hold the information about an enemy type, the amount, and rate to spawn them
19	Bullet	Normal Class	Use by all projectile types in the game
20	InteractableObject	Normal Class	Object that player can interact with
21	Portal	Normal Class	Use to transition player from one scene to another
22	Buff	Normal Class	Buff use to enhance player stats and ability
23	BuffType	Enum	Use to differentiate between the buff

24	BuffRNG	Struct	Use to spawn/apply random buff(s) after player cleared a room or enter an instance
25	ShopNPC	Normal Class	Control ShopNPC behaviour in game
26	Item	Normal Class	Control the item behavior in the game
27	ItemType	Enum	Item type to know which modifier pool to get the modifiers for the item
28	Inventory	Normal Class	Control the behavior of the player inventory in the game
29	Modifier	Normal Class	Hold the modifer data for the item
30	ModifierPool	Normal Class	Use to get random modifier from a pool of modifiers depend on the item type of the pool
31	ShieldEnemy	Normal Class	Shield Enemy type
32	Bomber	Normal Class	Suicide Enemy type
33	BringerOfDeathBoss	Normal Class	Boss enemy type
34	BoDState	Enum	State Bringer of Death use to control its behaviors
35	AudioManager	Normal Class	Control all the sound effect and soundtrack in the game
36	Sound	Enum	Sound enumerable to know which sound to play
37	SoundTrack	Enum	Soundtrack enumerable to know which soundtrack to play
38	SoundAudioClip	Normal Class	Control the audio source properties like volumn, pitch, loop, etc that being played by the AudioManager
39	MenuController	Normal Class	Control the menu of the game

2.2. Relationships

No.	Class Relationship	Relationship type	Note
1	MovingObject - DamageType	Association relationship	Relationship: (1) - (1)
2	MovingObject - StatusEffect	Association relationship	Relationship: (1) - (0..N)
3	Player - MovingObject	Inheritance relationship	Verb: "is" Relationship: (1) - (1)
4	PlayerController - Player	Composition relationship	Verb: "has" Relationship: (1) - (1)

5	Save System	Composition relationship	Verb: "has" Relationship: (1) - (1)
6	PlayerData	Composition relationship	Verb: "has" Relationship: (1) - (1)
7	Enemy - Moving Object	Inheritance relationship	Verb: "is"
8	Slime - Enemy	Inheritance relationship	Verb: "is"
9	Archer - Enemy	Inheritance relationship	Verb: "is"
10	Golem - Enemy	Inheritance relationship	Verb: "is"
11	Phase - Golem	Association relationship	None
12	GolemState - Golem	Association relationship	None
13	EnemySpawner - Slime	Aggregation relationship	Relationship: (1) - (0..N)
14	EnemySpawner - Archer	Aggregation relationship	Relationship: (1) - (0..N)
15	EnemySpawner - Wave	Association relationship	Wave is a subclass inside EnemySpanwer
16	EnemySpawner - SpawnState	Association relationship	None
17	BeginSpawnerPlate - EnemySpawner	Composition relationship	Verb: "has" Relationship: (1) - (1..N)
18	Portal - InteractableObject	Inheritance relationship	Verb: "is"
19	Buff - InteractableObject	Inheritance relationship	Verb: "is"
20	BuffType - Buff	Association relationship	None
21	BuffRNG - Buff	Association relationship	BuffRNG is a subclass inside Buff
22	ShieldEnemy - Enemy	Inheritance relationship	Verb: "is"
23	Bomber - Enemy	Inheritance relationship	Verb: "is"
24	BringerOfDeathBoss - Enemy	Inheritance relationship	Verb: "is"
25	Item - Modifier	Composition relationship	None
26	ModifierPool – Modifier	Association relationship	None
27	Inventory – Item	Composition relationship	Verb: "has" Relationship: (1) - (1..N)

28	AudioManager - SoundAudioClip	Composition relationship	None
29	Sound - AudioManager	Association relationship	None
30	SoundTrack - AudioManager	Association relationship	None
31	Sound - SoundAudioClip	Association relationship	None
32	SoundTrack - SoundAudioClip	Association relationship	None

3. Classes in details

MovingObject				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	hp	private	float	Health point of a moving object
2	movementSpeed	private	float	The movement speed in the game of the moving object
3	attackSpeed	private	float	Attack speed of the moving object
4	armour	private	int	Armour of a moving object, use to reduce the amount of physical damage taken
5	fireResistance	private	int	Fire resistance of a moving object, use to reduce the amount of fire damage taken
6	coldResistance	private	int	Cold resistance of a moving object, use to reduce the amount of cold damage taken
7	lightningResistance	private	int	Lightning resistance of a moving object, use to reduce the amount of lightning damage taken
8	damage	private	float	Damage a moving object can deal

9	burningDamage	private	float	Damage the moving object will taken overtime when affected by burning status effect. This is calculated base on the initial hit damage taken
10	damageType	private	DamageType	The damage type the moving object can deal
11	statusEffects	protected	StatusEffect	Status effects that get apply on moving object (Max of 3 at the moment). Status effects currently cannot stack
12	shockedTimer	protected	float	Use to remove the shock status affect after a period of time
13	burningTimer	protected	float	Use to remove the burning status affect after a period of time
14	freezingTimer	protected	float	Use to remove the freeze status affect after a period of time
15	isPlayer	protected	bool	Use to show the status effect Icon on player's HP bar
16	AppyStatusEffect()	public	void	Apply status effect on moving object
17	RemoveStatusEffect()	public	void	Remove status effect from moving object
18	takeDamage()	public	virtual void	Reduce moving object hp by the damage taken
19	BurningTimer()	private	void	Count down to remove burning status effect
20	FreezingTimer()	private	void	Count down to remove freezing status effect
21	ShockedTimer()	private	void	Count down to remove shocked status effect
22	TakeBurningDamage()	private	void	Take burning damage when affected by burning status effect

<table>
<tr><th colspan="5">Player</th></tr>
<tr><th>No.</th><th>Attribute/Method name</th><th>Access</th><th>Type</th><th>Description/Note</th></tr>
</table>

		Type		
1	MaxHP	private	float	Player's Max health points
2	baseDamage	private	float	Players base damage
3	isHurt	private	bool	Use to trigger immune time and immune state
4	fireRate	private	float	Use to calculate fire rate of player
5	fireTime	private	float	Use to calculate fire rate of player
6	fireType	private	int	Player fire type, can be single or multiple
7	bulletAmount	private	int	The amount of bullet player can shoot out in fire type: multiple
8	dashRange	private	float	Player's Dash range
9	meleeAttackRange	private	float	Player's melee attack range
10	restrictMovement	private	bool	Use to check if player's movements and actions are restricted
11	unlockedSlots	private	int	Player's unlocked slots
12	death	private	bool	Trigger player's death
13	coins	private	int	Amount of coins player currently has

PlayerController				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	ImmuneTime	private	double	Time until player are no longer immune after taking damage
2	dashCoolDown	private	float	Player's dash cooldown
3	instance	public	static PlayerController	Instance of player
4	coin_ToBeAdded	public	bool	Coin going to be added when the game save

5	inHO	public	bool	Player's in hideout or not
6	ProcessInput()	private	void	Process player input
7	FireBullet()	private	void	Shoot out projectiles single mode
8	FireBulletSpread()	private	void	Shoot out projectiles spread mode
9	GetBulletType()	private	Transform	Get bullet type prefab to fire out projectile of that type
10	Move()	private	void	Player moving around the instance
11	Dash()	private	void	Player dahs
12	MeleeAttack()	private	void	Player perform melee attack
13	takeDamage()	public	override void	Override take damage from moving object to add immunity to player when hurt
14	SetRestrictMovement()	public	void	Use by outside source to set restrict movement on player
15	CheckHP()	private	void	Check player current HP and process next action base on that
16	HasDied()	public	void	Use by outside sources to confirm player is dead
17	CleanseEffect()	public	void	Use when player die and return to hideout

Save System				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	LoadPlayer()	public	static PlayerData	Load player data
2	SavePlayer()	public	static void	Save player data

PlayerData				
No.	Attribute/Method name	Access Type	Type	Description/Note

1	UnlockedSlots	public	float	Player's unlocked slots
2	coins	public	int	Player's coins
3	FireRes	public	int	Only save base value of FireRes
4	ColdRes	public	int	Only save base value of ColdRes
5	LightningRes	public	int	Only save base value of LightningRes
6	Armour	public	int	Only save base value of Armour
7	BulletAmount	public	int	Only save base value of BulletAmount
8	FireType	public	int	Only save base value of FireType
9	Hp	public	float	Only save base value of Hp
10	MS	public	float	Only save base value of MS
11	AS	public	float	Only save base value of AS
12	damage	public	float	Only save base value of damage
13	FireRate	public	float	Only save base value of FireRate

Buff				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	buffType	private	BuffType	Buff type of the buff
2	statStickPrefab	private	GameObject	Stat stick prefab to get the stat stick player class
3	statStick	private	Player	Use to get the stats to buff player
4	ApplyBuff()	public	static void	Apply buff to player
5	SetUp()	public	void	Set up the buff with buff type
6	GetBuff()	public	BuffType	Get buff type from buff
7	Interact()	public	override void	Use by player to interact with the buff
8	Destroy()	public	void	Destroy the buff after it has applited its buff to player

9	SortBuffRNGByWeight()	public	static void	Sort the struct BuffRNG array by its weight value
10	SwapBuffRNG()	public	static void	Swap 2 BuffRNG variables with each other
11	BuffRNGTotalWeight()	public	static float	Return the BuffRNG total amount of weight
12	RollBuffRNG()	public	static BuffType	Roll the BuffRNG to return a random BuffType
13	RemoveDamageTypeBuffsFromPool()	public	static void	Remove DamageType buffs from the pool of BuffRNG. This only use when apply quirks to player at the beginning of an instance

BuffRNG				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	buffType	public	float	Buff type of the BuffRNG
2	weight	public	float	Weight of BuffRNG
	BuffRNG()	public		Constructor for BuffRNG

InteractableObject				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	Interact()	public	abstract void	Abstract function for inherited class to implement
2	TurnOnIcon()	public	void	Use by all inherited class to turn on interact icon
3	TurnOffIcon()	public	void	Use by all inherited class to turn off interact icon

Portal				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	Location	private	string	Scene name to move player to

2	MovePlayer()	private	void	Move player to the scene match the location string
3	Interact()	public	override void	Implement interact function from parent class (InteractableObject)

Bullet

No.	Attribute/Method name	Access Type	Type	Description/Note
1	shootDirection	private	vector3	Direction the bullet will be traveling
2	knockBack	private	vector3	Knock back that the bullet will have when hitting an object
3	damage	private	float	Damage the bullet carry
4	damageType	private	DamageType	Damage type the bullet carry
5	IsMoving	private	bool	To check if the bullet is still movin and proceed further action
6	bulletSpeed	private	float	Bullet traveling speed
7	IsFromPlayer	private	bool	Check if bullet is shot out from player or not
8	SetUp()	public	void	Set up the bullet damage, damageType, shootDirection etc
9	Hit()	private	void	Apply damage to the object the bullet hit, reduce bullet velocity to 0 and play the hit animation for bullet

NonMovingObject

No.	Attribute/Method name	Access Type	Type	Description/Note
1	hp	public	float	The hp of the non moving object
2	takeDamage()	public	void	Non moving object taking damage after getting hit

3	checkDestroy()	public	bool	Check if the hp is lower or equal to 0 and return a boo value

ShopNPC				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	MaxSlot	private	int	Max slots the player can purchase
2	Shop	private	GameObject	The Shop game object to display buy option
3	displayBuyOption()	public	void	Display buy option to the Shop game object
4	tryBuy()	public	void	Perform buying action
5	CalculatePrice()	public	void	Calculate and increase the price depending on the amount of slots unlocked

Enemy				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	roamTimer	public	float	Count down to start roaming
2	minCoinDrop	public	int	Minimum numberof coins the enemy can drop
3	maxCoinDrop	public	int	Maximum number of coins the enemy can drop
4	dropped	private	bool	Check if the monster has already dropped item
5	attackRange	private	float	Enemy attack range
6	knockBackForce	protected	float	Knock back force apply to enemy when hit by player
7	target	public	Transform	Target to chase and focus attack on
8	Roam()	protected	void	Roam around the instance in a pattern if player is too far away and roam timer is 0

9	DropCoins()	public	void	Drop coins
10	DropPotion()	public	voiid	Drop potion
11	DropItem()	public	void	Drop either coins or potion randomly when Enemy die
12	takeDamage	public	override void	Override take damage from moving object to apply other effects to enemy when damage is taken
13	Attack()	protected	void	Perform an attack
14	Move()	protected	void	Moving around an instance
15	FindTarget()	private	void	Find target to focus attack on and chase
16	CheckHP()	private	void	Check current HP if lower or equal to 0 play die animation and drop item
17	ProcessAction()	public	void	Process next action for Enemy

<table>
<tr><th colspan="5">Slime</th></tr>
<tr><td>No.</td><td>Attribute/Method name</td><td>Access Type</td><td>Type</td><td>Description/Note</td></tr>
<tr><td>1</td><td>ProcessAction()</td><td>public</td><td>override void</td><td>Override process action in parent class to perform custom action for slime</td></tr>
</table>

<table>
<tr><th colspan="5">Archer</th></tr>
<tr><td>No.</td><td>Attribute/Method name</td><td>Access Type</td><td>Type</td><td>Description/Note</td></tr>
<tr><td>1</td><td>ProcessAction()</td><td>public</td><td>override void</td><td>Override process action in parent class to perform custom action for Archer</td></tr>
<tr><td>2</td><td>Aim()</td><td>private</td><td>void</td><td>Calculate the direction to shoot at player</td></tr>
<tr><td>3</td><td>FireProjectile()</td><td>private</td><td>void</td><td>Fire a projectile toward player</td></tr>
</table>

ShieldEnemy				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	ProcessAction()	public	override void	Override process action in parent class to perform custom action for ShieldEnemy
2	ShieldHP	private	double	Shield health point foro ShieldEnemy this get reduce before the ShieldEnemy healthpoint when it get hit
3	ShieldCoolDown	private	float	Shield cooldown time
4	DamageMultiplerWithShild	private	float	The multipler of damage the ShieldEnemy get when they have their shield up
5	activateShield()	private	void	Activate the ShieldEnemy’s Shield
6	deactivateShield()	private	void	Deactivate the ShieldEnemy’s Shield
7	shieldCoolDown()	private	void	Process the cooldown for the ShieldEnemy’ shield
8	shieldTakeDamage()	public	void	Call when the ShieldEnemy’ shield is up and got hit

Bomber				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	BlastRadius	private	float	Radius of the blast when bomber explodes
2	explode()	private	void	Bomber explodes deal massive damage to player if player is in the blast radius
3	ProcessAction()	private	void	Override process action in parent class to perform custom action for Bomber

BossGolem				
No.	Attribute/Method name	Access Type	Type	Description/Note

1	slamCoolDown	private	float	Slam skill cooldown
2	laserCoolDown	private	float	Laser skill cooldown
3	pullCoolDown	private	float	Pull skill cooldown
4	foldCoolDown	private	float	Fold skill cooldown
5	slamRange	private	float	Slam range
6	pullRange	private	float	Range to perform the pulling action
7	phase	private	Phase	Phase of the golem depending on hp
8	state	private	State	Golem's state, use to switch between action and play animation
9	Slam()	public	void	Perform slam attack, make public so that it can be use after the arm of golem reach the ground at the end of slamming animation
10	FireArm()	private	void	Perform fire arm attack
11	Roll()	private	void	Perform rolling attack
12	Lasering()	private	void	Perform lasering attack
13	updatePhase()	private	void	Update golem phase
14	updateState()	private	void	Update golem state
15	ProcessAction()	private	override void	Override parent function to add custom actions and behaviour
16	CheckHP()	private	override void	Check golem hp to update phase and check if golem is dying or not
17	SummonPillar()	private	void	Summon pillar after finish rolling
18	AttackType	private	void	Choose the attack type to perform i.e. Slam, FireArm, Laseringi etc
19	SlamCDTimer()	private	void	Slam cooldown timer
20	RollandLaserCDTimer()	private	void	Roll and Laser cooldown timer
21	PullCDTimer()	private	void	Fire arm cooldown timer

22	MaxHP	private	float	Golem Boss’s max health point

BringerOfDeathBoss				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	MaxHP	private	float	Bringer of Death Boss’s max health point
2	spellAmountPerCast	private	int	The number of spells the boss will rain down on the player (default is 15)
3	castAmount	private	int	The amount of time the boss will cast the spell when the cooldown is up
4	currentCast	private	int	The current cast the boss is performing to know when to stop the spell casting animation
5	spellOffsetY	private	float	The spell offset in y position to spawn the spell above player
6	attackRange	private	Vector2	This is different from attack range because the Bringer of Death Boss’s attack require it to know the x and y of its position compared to the player x and y position
7	spellCD	private	bool	The cooldown variable to trigger spell cast of Bringer of Death Boss
8	teleportCD	private	bool	The cooldown variable to trigger teleport skill of Bringer of Death Boss
9	cloneCD	private	bool	The cooldown variable to trigger clone skill of Bringer of Death Boss
10	spellCDTime	private	float	The time for the Bringer of Death spell cooldown
11	teleportCDTime	private	float	The time for the Bringer of Death teleport skill cooldown

12	cloneCDTime	private	float	The time for the Bringer of Death clone skill cooldown
13	isClone	private	bool	The variable to check if the Bringer of Death Boss prefab is a clone to change its stats and behaviro
14	Phase	private	Bool	Phase of the Bringer of Death Boss
15	state	private	BoDState	State to determine the Bringer of Death Boss behavior
16	CheckAttackRange()	private	void	Check the position of the Bringer of Death Boss and the Player position and see if the Bringer of Death Boss should perform an attack or another action
17	CastSpell()	private	IEnumerator	Bringer of Death cast the spell that rain down on Player
18	CoolDown()	public	IEnumerator	Perform the cooldown action for all Bringer of Death Boss's skills by calling a function to flip the bool variable to false so the Boss know to perform the next skill action
19	SpellCD()	private	void	Use to flip the bool variable for spell cooldown to false
20	TeleportCD()	private	void	Use to flip the bool variable for teleport skill cooldown to false
21	CloneCD()	private	void	Use to flip the bool variable for clone skill cooldown to false
22	Teleport()	private	IEnumerator	Teleport the Bringer of Death Boss to the position of the Player with slight offset depend on the Boss's sprite width and height
23	CloneSkill()	private	void	Bringer of Death Boss perform a clone skill and create mirage of itself to discrupt player
24	CloneDieAfter()	private	IEnumerator	The function is used to kill the clone of Bringer of Death Boss

				after a certain amount of time
25	CheckLife()	private	void	Check the life of Bringer of Death Boss to switch its phase
26	SetUpClone()	public	void	Function use to set up the Bringer of Death clone stats and behavior

EnemySpawner				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	waves	public	Wave	Array of Wave to spawn, Wave hold enemy prefab and extra information
2	nextWave	private	int	Next wave to spawn
3	timerBetweenWave	public	float	Time between wave
4	waveCountDown	public	float	Count down before the wave start
5	searchCountDown	public	float	Countdown before start searching for alive enemies that come out of spawner
6	state	private	SpawnState	Current state of the spawner
7	EnemyIsAlive()	private	bool	Search for alive enemies that come out from the spawner
8	WaveComplete()	private	void	Proceed to next wave
9	GetSpawnState()	public	SpawnState	Get the current SpawnState of the spawner
10	SpawnEnemy()	private	void	Spawn enemy base on the Wave information

Wave				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	name	public	string	Wave name
2	enemy	public	Transform	Prefab of enemy to spawn

3	count	public	int	Amount of enemy to spawn
4	rate	public	float	The rate of spawning

BeginSpawnerPlate				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	start	public	bool	To notify the spawner that player has stepped on the plate and begin the spawner

Item				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	name	public	string	Name of the item to display on the item tooltip
2	itemsIcon	public	Sprite	Item Sprite to set up the image display in the Inventory UI
3	currentSprite	public	Sprite	Current Sprite the Item have
4	amount	public	int	The amount of the same item to display in the Inventory
5	modifiers	public	Modifier	Modifier data to apply to the Player when the item is equiped
6	itemType	public	ItemType	The type of the item it can be a weapon or a defense item
7	inInventory	public	bool	Bool variable to check if the item is in the Inventory or not
8	equiped	public	bool	Bool variable to check if the item is equiped or not
9	image	public	Image	The image to display in the Inventory UI
10	Initialize()	public	void	Initialize the item when it spawned
11	GetCurentSprite()	public	Sprite	Get the current sprite the item is using
12	ApplyItemEffect()	public	void	Applying the item effect to

				Player
13	SetUpItemModifiers()	public	void	Setting up the item modifiers by getting the modifers data from a Modifer Pool
14	Interact	public	override void	Use by player to Interact with the item by picking it up and put it in the Inventory
15	RemoveItemEffect	public	void	Removing the item effect from Player

Modifier				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	modData	public	JObject	Contain the modifier data get from the item modifier database which is a json file that hold all the modifier data
2	rollValue	public	int	Roll value is the random value that the modifers can have from a range of value after the item is initialize and the modifers in the item are setup

ModifierPool				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	modifiers	public	JObject	List of JObject that represent the modifiers in ModifierPool
2	modifiersData	public	JObject	Contain the data information of the modifiers in the ModifierPool
3	GetModifierData()	public	JObject	Get the data of modifier and return a JObject this is use to set up the ModifierPool
4	GetModifierPoolTotalWeight()	public	int	Get the total weighting of all the modifiers in the ModifierPool to use in cumulative probability calculation to roll the modifiers

5	SortModifierByWeight()	public	void	Sort the modifiers list in the ModifierPool to use in rolling modifiers for item process
6	RollMultipleModifier()	public	Modifier	Get the JObject from the RollModifier() function to create Modifier object and return a list of rolled Modifier objects
7	RollModifier()	public	JObject	Get a modifier from the modifiers list inside the ModifierPool by using cumulative probability

Inventory				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	itemList	pulbic	Item	The list of Item object in the Inventory
2	playerPreviousFireType	public	int	Player previous fire type. This is assigned when the player equips an item that change the player fire type to something else. It's being used as a save point to revert the player fire type when the item that affect the player fire type is unequiped
3	instance	public	Inventory	The instance of the Inventory so that it can be access globally
4	ToggleInventory()	public	void	Toggle the Inventory UI on or off
5	AddItem()	public	bool	Add the item into the inventory if the slot in the inventory is full the function will return and true if otherwise
6	GetIitemList()	public	Item	Return the list of Item object that's in the Inventory
7	AddItemToEquipmentSlot()	public	bool	Add the item in the Inventory into the equipment slot to apply the item modifers to the player. The function returns false if equipment slots is fulled and return

				true if otherwise. If false, the Item will stay inside the Inventory until the equipment slots is free up so it can be equipped
8	RemoveItemFromEquipmentSlot()	public	bool	Remove the item from equipment slot back to the Inventory slot if the Inventory is full the funcntion return false and true if otherwise. If false, the Item object will stay in the equipment slot until the Inventory has free up some space
9	FireTypeItemEquiped()	public	bool	Function to check if there are any item that can change Player's fire type is being equiped
10	SetPlayerPreviousFireType()	public	void	Set the playerPreviousFireType
11	GetPlayerPreviousFireType()	public	int	Get the playerPreviousFireType
12	ReloadInventoryUI()()	public	void	Reload the Inventory UI

AudioManager				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	instance	public	AudioManager	Instance of the AudioManager to access it globally
2	oneShotGameObject	private	GameObject	The sound effect that gets play once
3	oneShotAudioSource	private	AudioSource	The audio source of the oneshot sound effect
4	soundTrackGameObject	private	GameObject	The soundtrack of the game (background music)
5	soundTrackAudioSource	private	AudioSource	The audio source of the soundtrack
6	soundAudiosClipArray	public	SoundAudioClip	A list of all the audio clip to play for the sound effect
7	soundTrackArray	public	SoundAudioClip	A list of all the soundtrack to play depend on the scene
8	PlaySound()	public	void	Play the sound effect
9	PlaySoundTrack()	public	void	Play the soundtrack

10	removeSound()	public	void	Remove a specific sound by deleting the gameojbect that contain the sound
11	GetAudioClip()	public	AudioClip	Get the AudioClip from the soundAudiosClipArray or soundAudiosClipArray to set it up or playing it
12	CanPlaySound()	private	static bool	Check if the specify sound can be play
13	GetSoundTrackGameObject()	public	GameObject	Get the gameobject that cotain the soundtrack to reuse the object to play another soundtrack without having to constantly delete it whenever a new soundtrack is being played
14	FadeOutST()	public	IEnumberator	Fade the current soundtrack out and player another one if specified

SoundAudioClip				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	name	public	string	Name of the audio clip
2	sound	public	<generic>	Sound type of the audio clip. It’s a generic variable and can be either Sound or Soundtrack
3	audioClip	public	AudioClip	The audio clip to play
4	pitch	public	float	The pitch of the audio clip
5	loop	public	bool	Variable to check if the audio should be looping
6	volume	public	float	The volume of the audio clip

MenuController				
No.	Attribute/Method name	Access Type	Type	Description/Note
1	instance	public	MenuController	Instance of the menu controller so it can be access globally

2	PauseMenu	public	GameObject	The pause menu UI gameobject
3	PauseMenu_Confirmation	public	GameObject	The confirm button in the pause menu UI gameobject
4	DeathMenu	public	GameObject	The death menu UI gameobject
5	mainCanvas	public	GameObject	The main canvas of the game
6	HO_PauseMenu	public	GameObject	The hideout pause menu UI gameobject to return to main menu
7	CheckPlayerDeath()	public	void	Checking if the player health point is <=0 to show the death menu
8	CheckInput()	public	void	Check the player input to know which menu to show
9	BackToMainMenu()	public	void	Return player to main menu
10	BackToHideout()	public	void	Return player to hideout
11	BackToHideout_Penalty()	public	void	Return player to hideout but with penalty (when player retreat)
12	BackToHideout_Death()	public	void	Return player to hideout but with no reward because player died
13	UnfreezeTime()	public	void	Unpause the game
14	TutorialSceneReset()	public	void	Reset the objects in tutorial area
15	ReloadMenuObject()	public	void	Reload the menu UI gameobject
16	LoadHideoutPauseMenu()	public	void	Load the hideout pause menu UI gameobject
16	LoadHideoutShopUI()	public	void	Load hideout npc shop UI gameobject

II. State Diagrams

1. Player state diagram

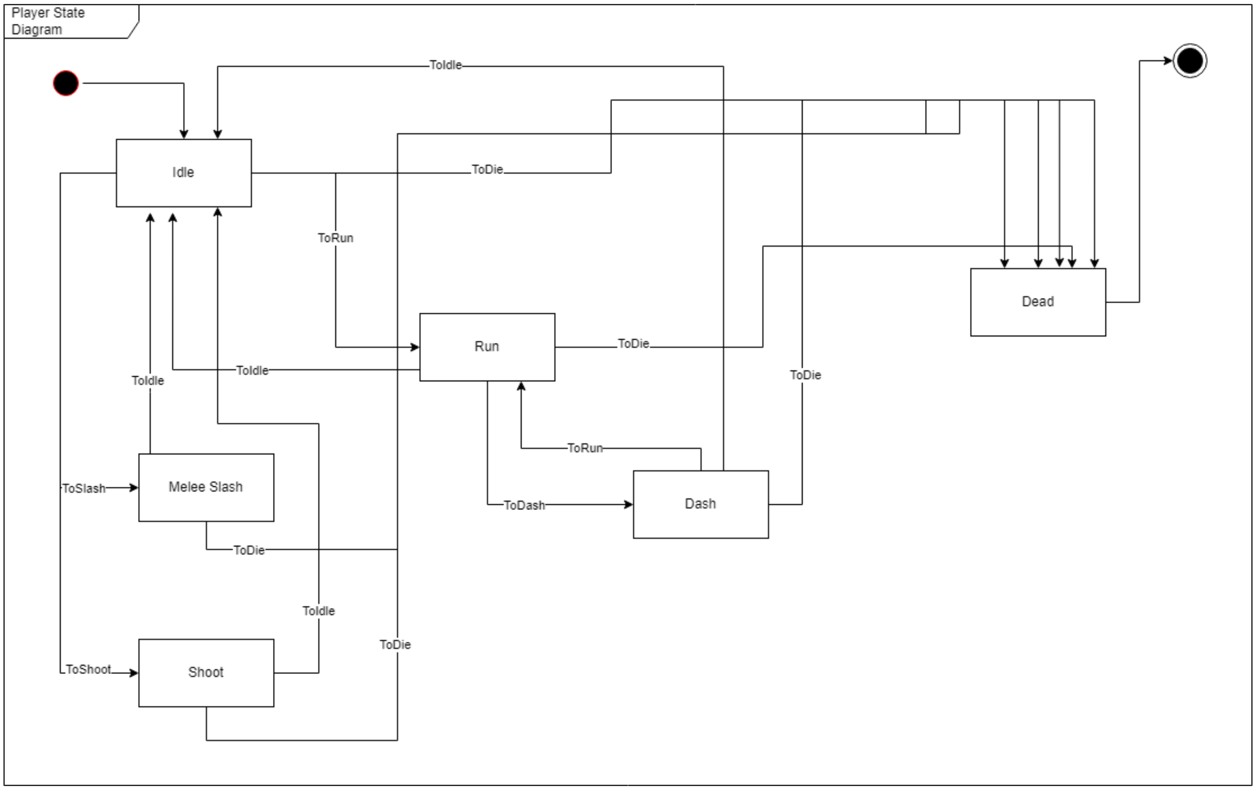


Image 17: Player State Diagram

Alias	State
s0	Initial state
s1	Idle state
s2	Run state
s3	Dash state
s4	Melee splash state
s5	Shoot
s6	Dead state
s7	Final state

Alias	Action
e1	ToIdle
e2	ToRun
e3	ToDash
e4	ToSlash
e5	ToShoot
e6	ToDead

No.	State	Action/Event	Result state
0	s0		s1
1	s1	e2	s2
2	s1	e4	s4
3	s1	e5	s5
4	s1	e6	s6
5	s2	e1	s1
6	s2	e3	s3
7	s2	e6	s6
8	s3	e1	s1
9	s3	e2	s2
10	s3	e6	s6
11	s4	e1	s1
12	s4	e6	s6
13	s5	e1	s1
14	s5	e6	s6
15	s6		s7

2. Slime state diagram

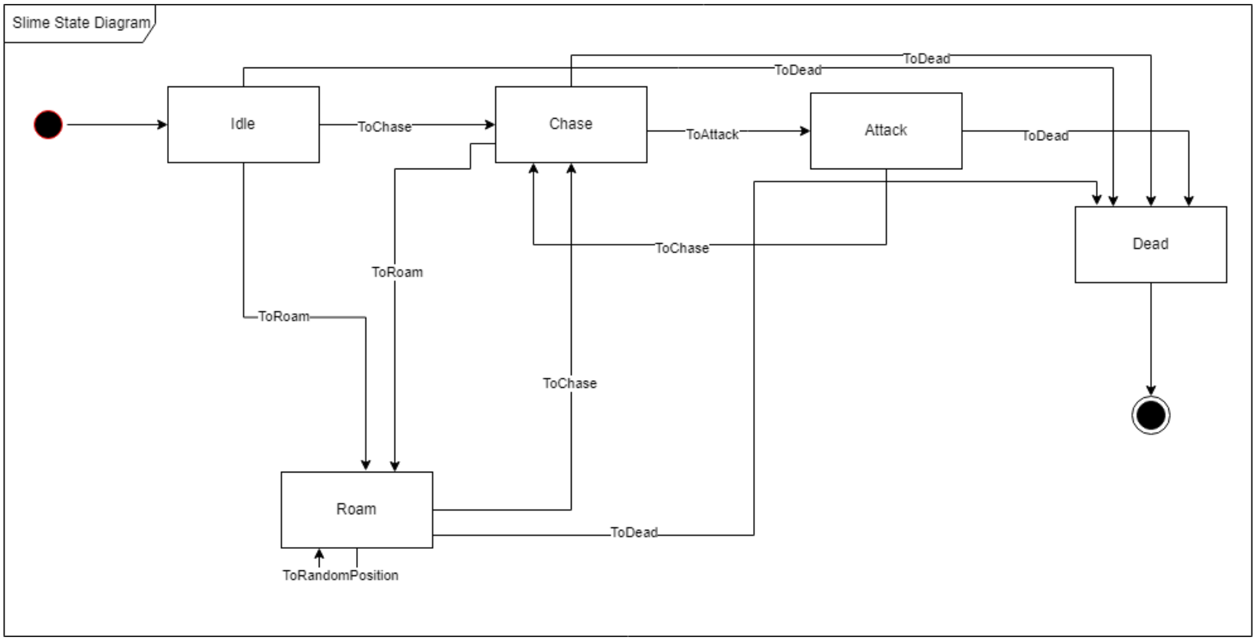


Image 18: Slime State Diagram

Alias	State
s0	Initial state
s1	Idle state
s2	Chase state
s3	Roam state
s4	Attack state
s5	Dead state
s6	Final state

Alias	Action
e0	ToChase
e1	ToRoam
e2	ToRandomPostion
e3	ToDead
e4	ToAttack

No.	State	Action/Event	Result state
1	s0		s1
2	s1	e1	s3
3	s1	e0	s2
4	s1	e3	s5
5	s2	e1	s3
6	s2	e4	s4
7	s2	e3	s5
8	s3	e2	s3
9	s3	e0	s2
10	s3	e3	s5
11	s4	e3	s5
12	s4	e0	s2
13	s6		s6

3. Archer state diagram

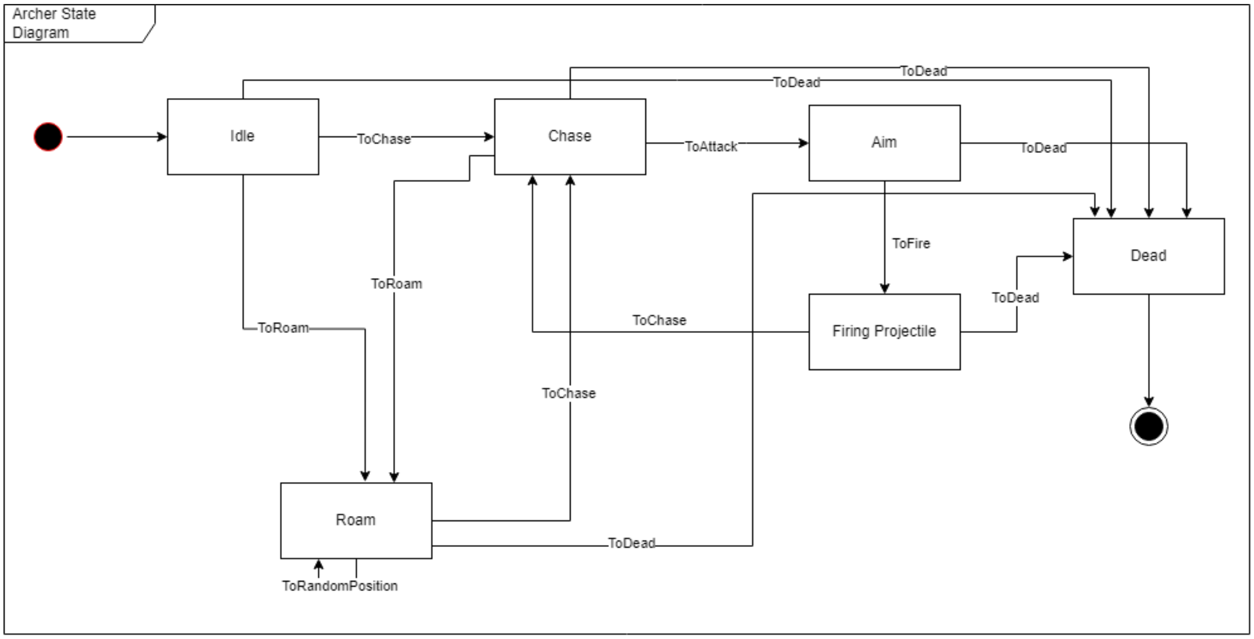


Image 19: Archer State Diagram

Alias	State
s0	Initial state
s1	Idle state
s2	Chase state
s3	Roam state
s4	Aim
s5	Firing projectiles
s6	Dead state
s7	Final state

Alias	Action
e2	ToChase
e3	ToRoam
e4	ToAttack
e5	ToFire
e6	ToDead
e7	ToRandomPosition

No.	State	Action/Event	Result state
0	s0		s1
1	s1	e2	s2
2	s1	e3	s3
3	s1	e6	s6
4	s2	e4	s4
5	s2	e6	s6
6	s3	e7	s3
7	s3	e2	s2
8	s3	e6	s6
9	s4	e5	s5
10	s4	e6	s6
11	s5	e6	s6
12	s6		s7

4. Golem state diagram

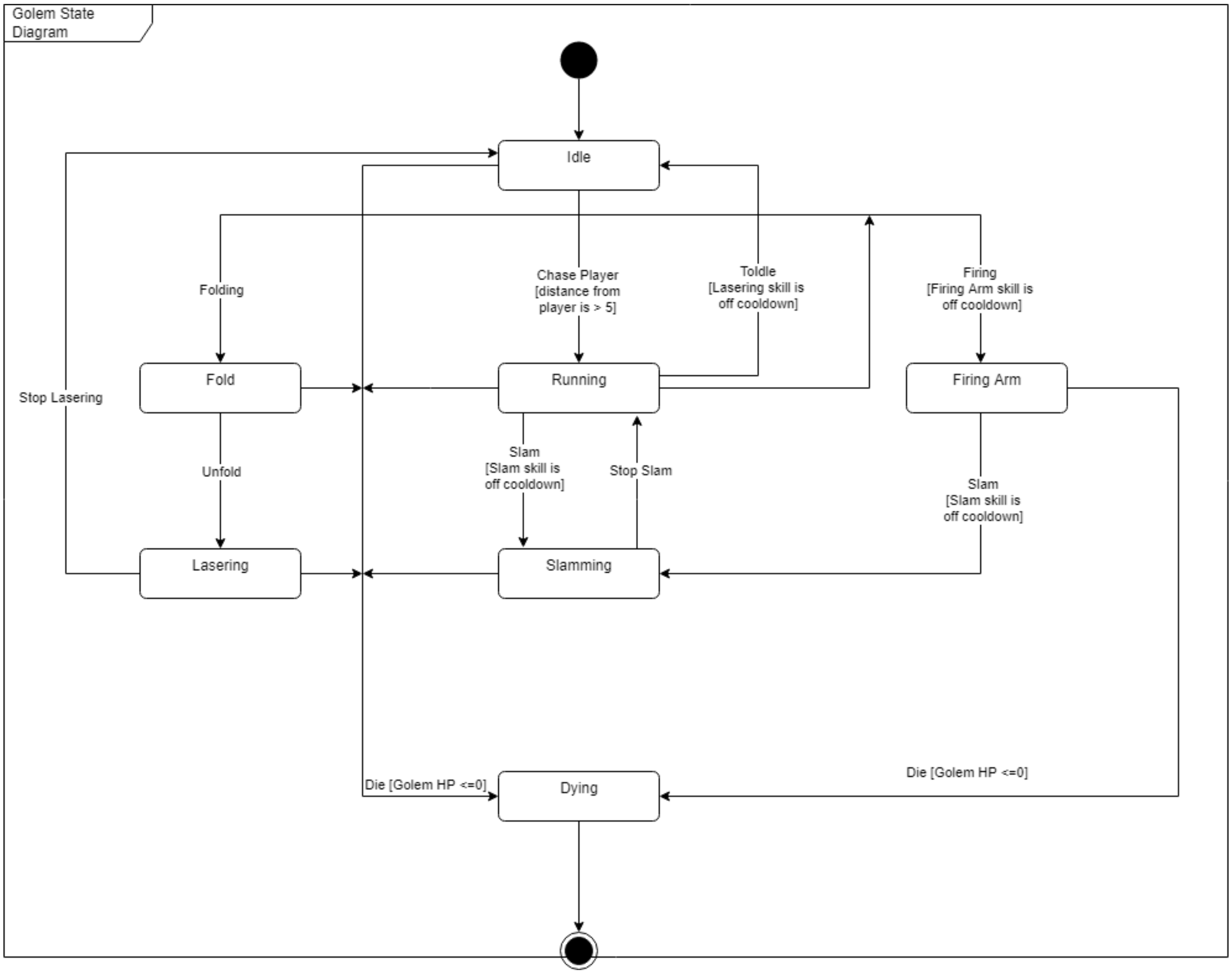

Golem State Diagram
Idle
Folding
Chase Player [distance from player is > 5]
ToIdle [Lasering skill is off cooldown]
Firing [Firing Arm skill is off cooldown]
Stop Lasering
Fold
Running
Firing Arm
Unfold
Slam [Slam skill is off cooldown]
Stop Slam
Slam [Slam skill is off cooldown]
Lasering
Slamming
Die [Golem HP <=0]
Dying
Die [Golem HP <=0]

Image 20: Golem State Diagram

Alias	State
s0	Initial state
s1	Idle
s2	Running
s3	Fold
s4	Firing Arm
s5	Lasering
s6	Slamming
s7	Dying
s8	Final state

Alias	Action
e1	ToIdle
e2	Chase player
e3	Folding

No.	State	Action/Event	Result state
0	s0		s1
1	s1	e2	s2
2	s1	e7	s7
3	s2	e3	s3
4	s2	e4	s4
5	s2	e6	s6
6	s2	e7	s7
7	s3	e1	s1
8	s3	e7	s7
9	s4	e6	s6
10	s4	e7	s7
11	s5	e1	s1
12	s5	e7	s7
13	s6	e8	s2

e4	Firing
e5	Unfold
e6	Slam
e7	Die
e8	StopSlam

14	s6	e7	s7
15	s7		s8

5. Bullet state diagram

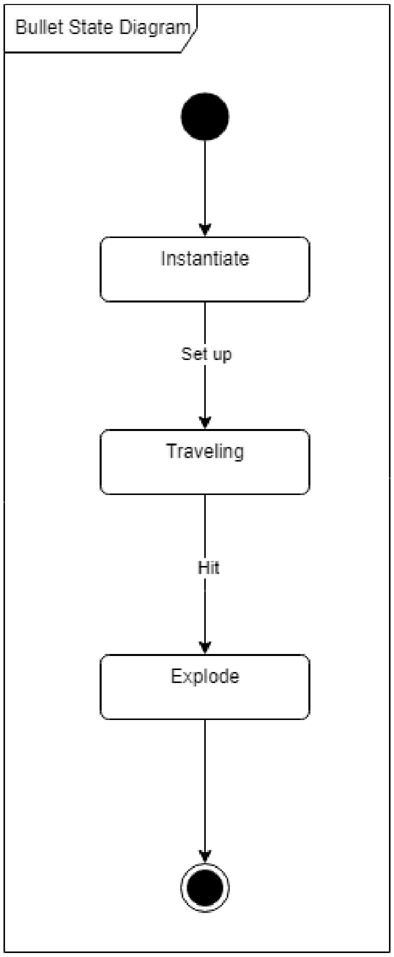


Image 21: Bullet State Diagram

Alias	State
s0	Initial state
s1	instantiate
s2	Traveling
s3	Explode
s4	Final state

No.	State	Action/Event	Result state
0	s0		s1
1	s1	e0	s2
2	s2	e1	s3
3	s3		s4

Alias	Action

e0	Set up
e1	Hit

6. Non-Moving Object state diagram

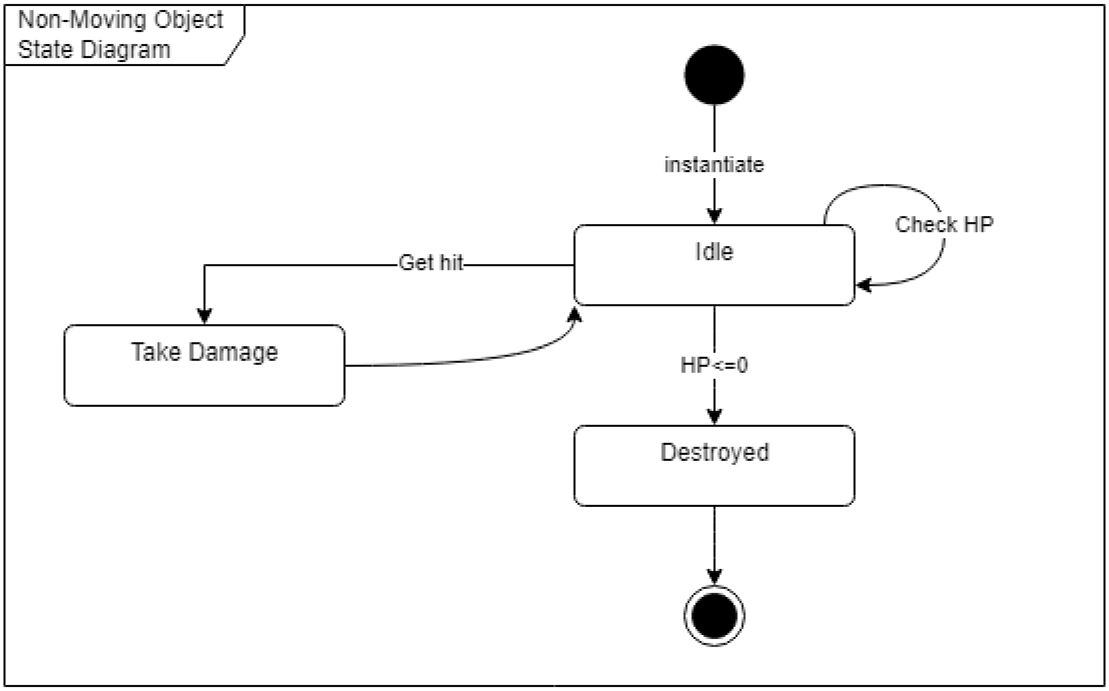


Image 22: Non-Moving Object State Diagram

Alias	State
s0	Initial state
s1	Idle
s2	Take damage
s3	Destroyed
s4	Final state

Alias	Action
e0	Check HP
e1	HP<=0
e2	GetHit

No.	State	Action/Event	Result state
0	s0		s1
1	s1	e0	s1
2	s1	e2	s2
3	s1	e1	s3
4	s2		s1
5	s3		s4

7. ShopNPC state diagram

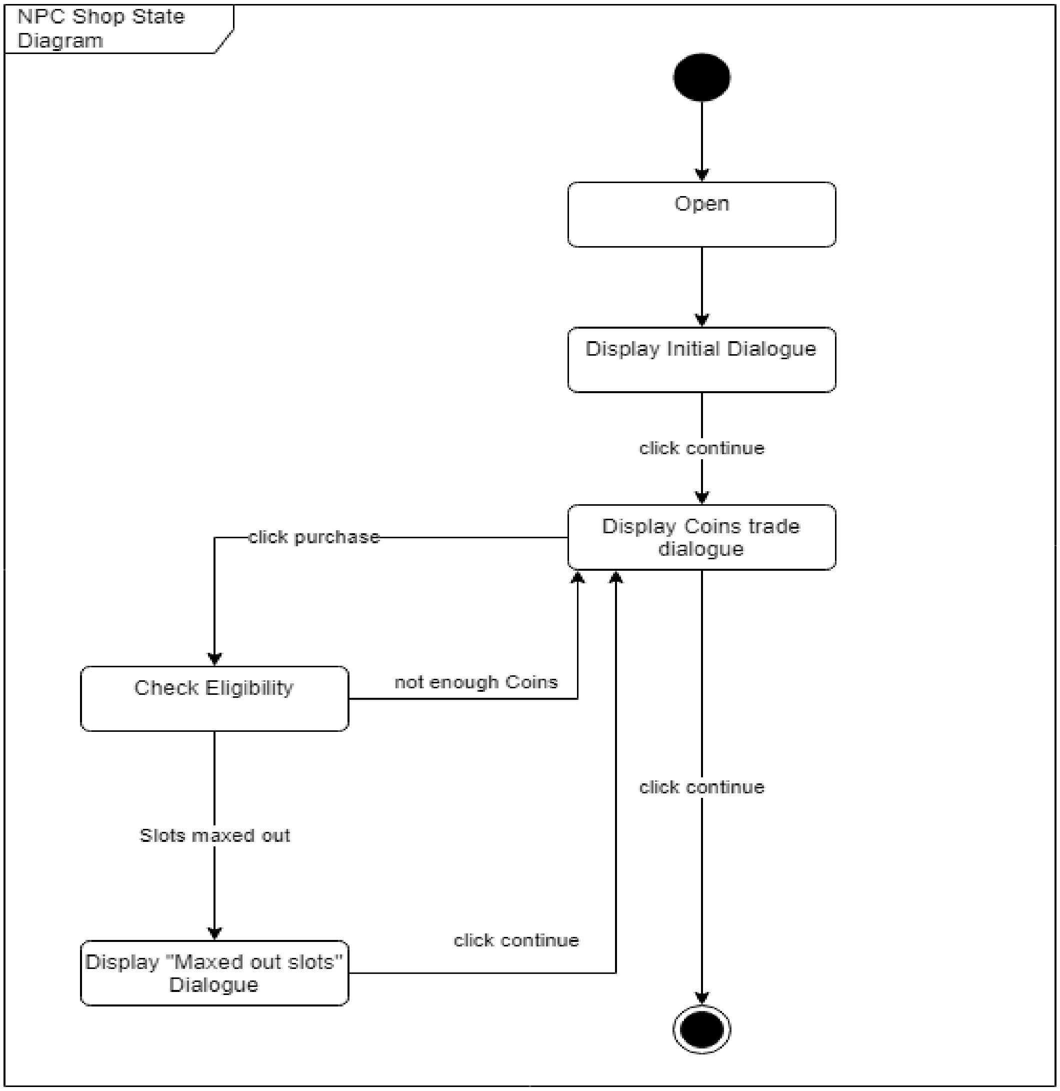


Image 23: Shop NPC State Diagram

Alias	State
s0	Initial state
s1	Open
s2	Display initial dialogue
s3	Display coin trade dialogue
s4	Check eligibility
s5	Display "Maxed out slots" dialogue
s6	Final state

Alias	Action
e0	Click continue
e1	click purchase

No.	State	Action/Event	Result state
0	s0		s1
1	s1		s2
2	s2	e1	s3
3	s3	e1	s4
4	s3	e0	s6
5	s4	e2	s3
6	s4	e3	s5
7	s5	e0	s3

e2	not enough coins
e3	slots maxed out

8. Buff state diagram

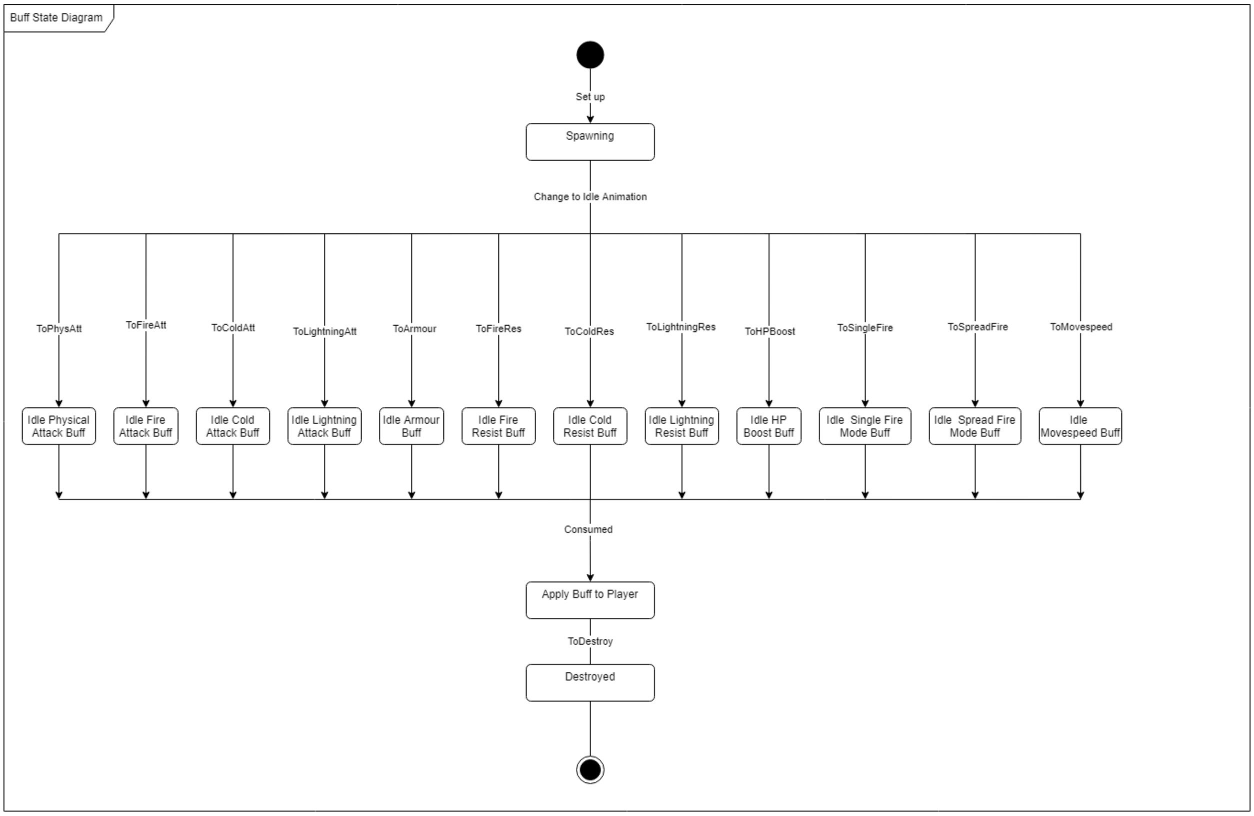


Image 24: Buff State Diagram

Alias	State
s0	Initial state
s1	Spawning
s2	Idle Physical attack buff
s3	idle fire attack buff
s4	idle lightning attack buff
s5	idle cold attack buff
s6	idle armour buff
s7	idle fire resist buff
s8	idle lightning resist buff
s9	idle cold resist buff
s10	idle HP boost buff
s11	idle spread fire mode buff
s12	idle single fire mode buff
s13	idle movespeed buff
s14	Apply buff to player

No.	State	Action/Event	Result state
0	s0	e1	s1
1	s1	e2	s2
2	s1	e3	s3
3	s1	e4	s4
4	s1	e5	s5
5	s1	e6	s6
6	s1	e7	s7
7	s1	e8	s8
8	s1	e9	s9
9	s1	e10	s10
10	s1	e11	s11
11	s1	e12	s12
12	s1	e13	s13
13	s2	e13	s14
14	s3	e14	s15

s15	Destroyed
s16	Final state

Alias	Action
e1	set up
e2	ToPhysAtt
e3	ToFireAtt
e4	ToLightningAtt
e5	ToColdAtt
e6	ToFireRes
e7	ToLightningRes
e8	ToColdRes
e9	ToHPBoost
e10	ToSingleFire
e11	ToSpreadFire
e12	ToMovespeed
e13	Consume
e14	ToDestroy

15	s4	e15	s16
16	s5	e16	s17
17	s6	e17	s18
18	s7	e18	s19
19	s8	e19	s20
20	s9	e20	s21
21	s10	e21	s22
22	s11	e22	s23
23	s12	e23	s24
24	s13	e24	s25
25	s14	e14	s15
26	s15		s16

9. ShieldEnemy state diagram

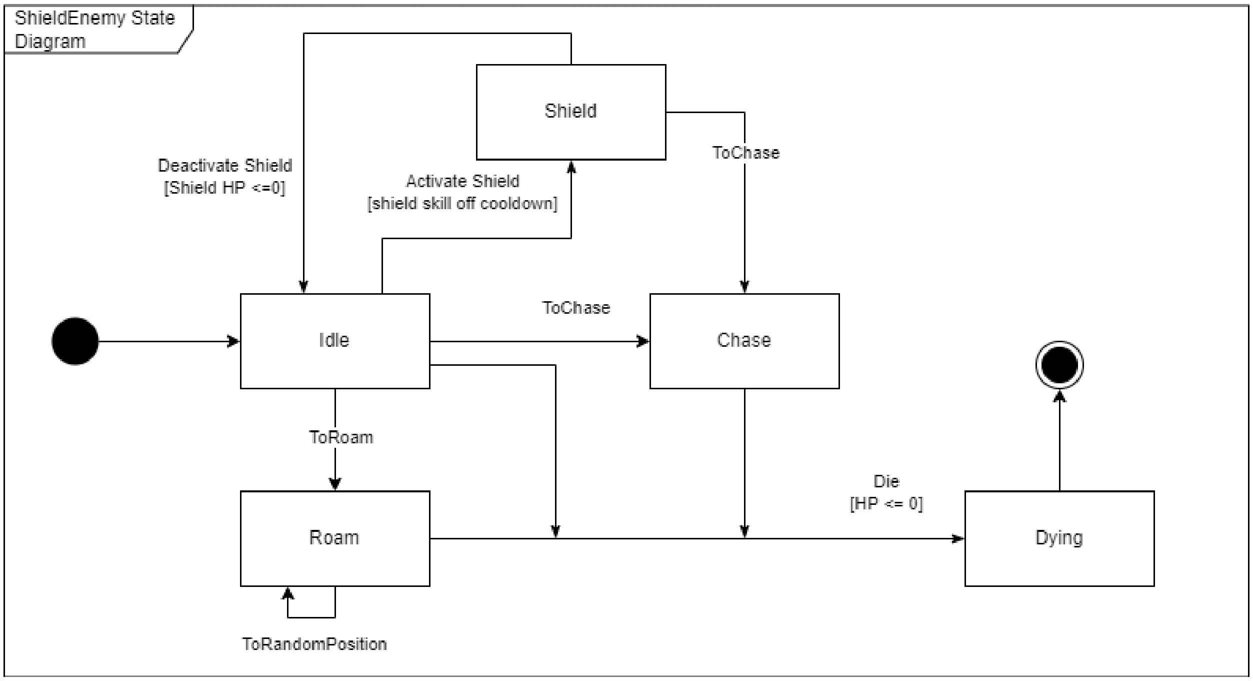

ShieldEnemy State Diagram
Shield
Deactivate Shield [Shield HP <=0]
Activate Shield [shield skill off cooldown]
ToChase
ToChase
Idle
Chase
ToRoam
Roam
Die [HP <= 0]
Dying
ToRandomPosition

Image 25: Shield Enemy State Diagram

Alias	State
s0	Initial state
s1	Idle
s2	Roam

No.	State	Action/Event	Result state
1	s0		s1
2	s1	e1	s3
3	s1	e2	s2

s3	Chase
s4	Shield
s5	Dying
s6	Final state

Alias	Action/Event
e1	ToChase
e2	ToRoam
e3	ToRandomPosition
e4	Activate Shield
e5	Deactivate Shield
e6	Die

4	s1	e4	s4
5	s2	e3	s2
6	s4	e5	s1
7	s1	e6	s5
8	s2	e6	s5
9	s3	e6	s5
10	s5		s6

10. Bomber state diagram

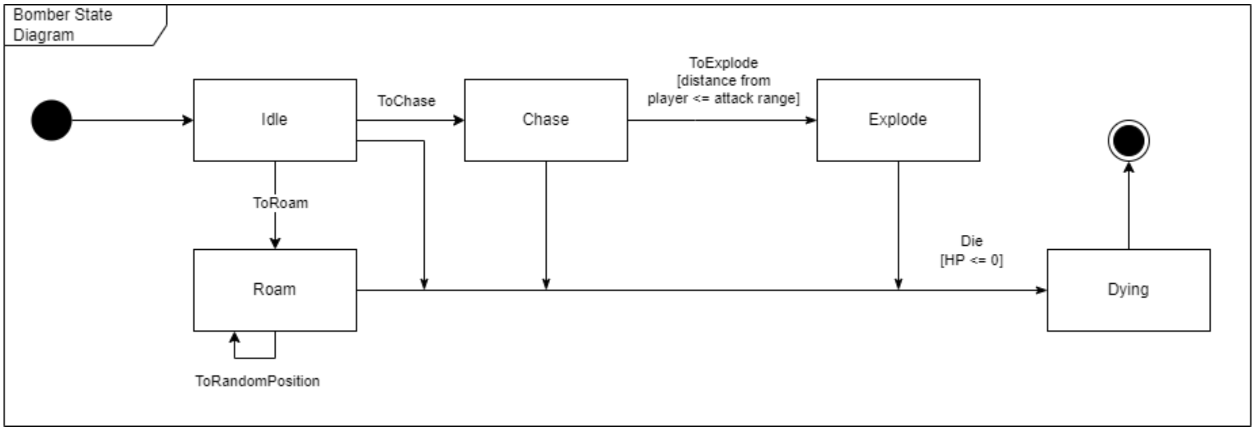


Image 26: Bomber State Diagram

Alias	State
s0	Initial state
s1	Idle
s2	Roam
s3	Chase
s4	Explode
s5	Dying
s6	Final state

Alias	Action/Event
e1	ToChase
e2	ToRoam
e3	ToRandomPosition
e4	ToExplode
e5	Die

No.	State	Action/Event	Result state
1	s0		s1
2	s1	e1	s3
3	s1	e2	s2
4	s3	e4	s4
5	s4	e5	s5
6	s1	e5	s5
7	s2	e5	s5
8	s3	e5	s5
9	s2	e3	s2
10	s5		s6

11. BringerOfDeathBoss state diagram

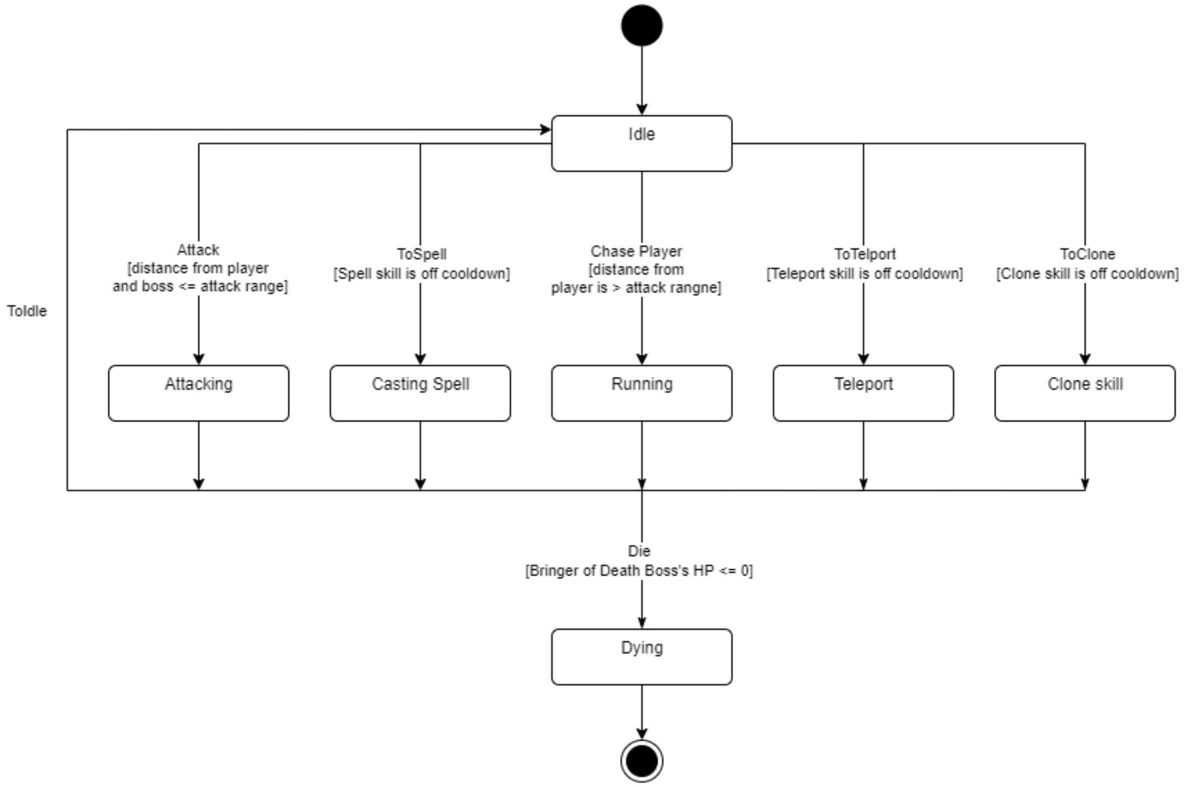

Idle
Attack
[distance from player
and boss <= attack range]
ToSpell
[Spell skill is off cooldown]
Chase Player
[distance from
player is > attack rangne]
ToTelport
[Teleport skill is off cooldown]
ToClone
[Clone skill is off cooldown]
ToIdle
Attacking
Casting Spell
Running
Teleport
Clone skill
Die
[Bringer of Death Boss's HP <= 0]
Dying

Image 27: Bringer of Death Boss State Diagram

Alias	State
s0	Initial state
s1	Idle
s2	Running
s3	Casting Spell
s4	Attacking
s5	Teleport
s6	Clone Skill
s7	Dying
s8	Final state

Alias	Action
e1	ToIdle
e2	Chase player
e3	ToSpell
e4	Attack
e5	ToTeleport
e6	ToClone
e7	Die

No.	State	Action/Event	Result state
0	s0		s1
1	s1	e2	s2
2	s1	e3	s3
3	s1	e4	s4
4	s1	e5	s5
5	s1	e6	s6
6	s2	e1	s1
7	s3	e1	s1
8	s4	e1	s1
9	s5	e1	s1
10	s6	e1	s1
11	s2	e7	s7
12	s3	e7	s7
13	s4	e7	s7
14	s5	e7	s7
15	s6	e7	s7
16	s7		s8

CHAPTER IV: DATA DESIGN

I. Player data

The data is processed in respect of the player’s assigned slot when the user picks a save slot. Each one is named as "Player" + assigned slot + ".savefile".

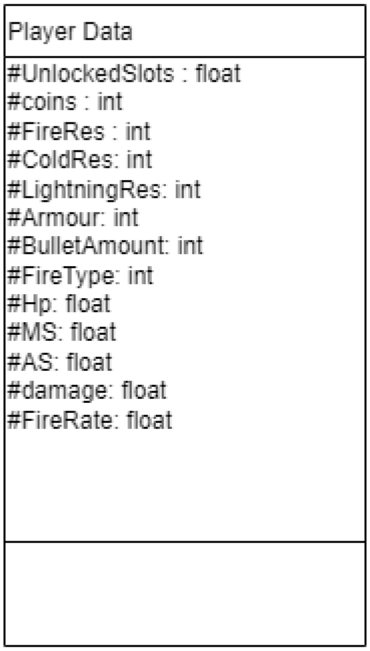
The data is serialized and stored locally in the persistent path and can be deserialized to recover its usage in the game.

Tool used for serialization and deserialization:

- Library JsonUtility
- serialization : ToJson
- deserialization : FromJson

Persistent path location :

C:/Users/Username/AppData/LocalLow/DefaultCompany/DaGame



Player Data

#UnlockedSlots : float
#coins : int
#FireRes : int
#ColdRes: int
#LightningRes: int
#Armour: int
#BulletAmount: int
#FireType: int
#Hp: float
#MS: float
#AS: float
#damage: float
#FireRate: float

II. Modifiers Data

The modifiers data are saved in a json file which is inside the game resource folder for ease of access.

Each modifier for now will be categorized into either “weapon modifier” group or “armour modifier” group and will have some basic properties like

```
{
        "name":
        "weight":,
        "description":,
        "popupMessage":{
                "message”:,
                "value":
        }
}
```

Some modifiers with have their own custom properties like “range” if they have value that can be roll in a range of value. And some will have custom property that change how the player play the game like the “spread shot” modifer which will have properties like “attackSpeed”, “damageReduce” which are multipliers that affect player stats and how player shoot projectiles.

CHAPTER V: STRUCTURAL DESIGN

I. Game design

1. Core Game Loop

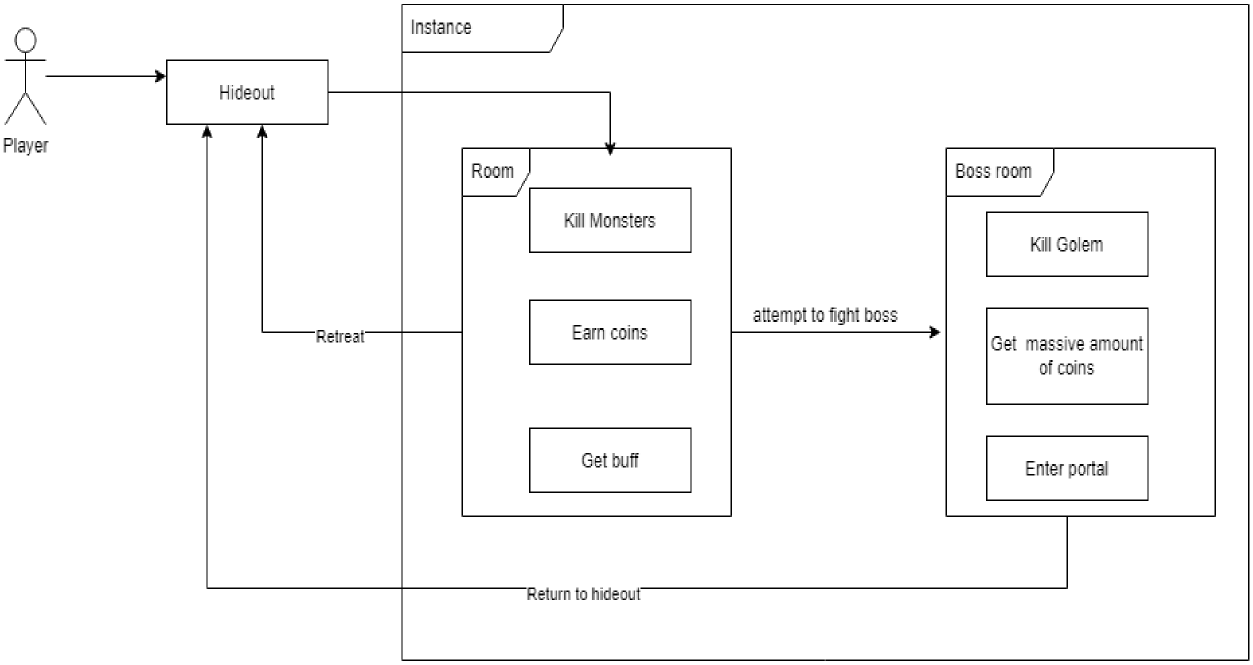


Image 28: The game structural design

The current core game loop is player enter an instance and kill all monster in a room then consume the buff that spawn. After that player move to another room and do the same. At the end of the instance there is a boss room player can enter the boss room and attempt to kill the boss but they can also retreat to hideout and keep 30% of the coins they found. Player can use that coins they earn to unlock slots to get random quirks depending on amount of slots they have to boost their stats before enter an instance. Rinse and repeat.

2. Progression

Player progress through the game by collecting coins to unlock slots and use those slots to get random quirks at the beginning of each instance.

II. Software design

We plan to use the model - view - controller architecture to design the game

- Model - game logic
- View - Graphics, 3D models, etc..
- Controller - Interface to the model

View

- User Action -> Controller
- Read state -> Model.State

Controller

- Write State -> Model.State
- Call Functions - > Model.function

Model

- State
- Functions

III. Design Patterns Use

- Singleton:

Having concerned in the statement project, the game is dense with objects and classes. Most objects interact with the environment and react back to it and the player interacts with most of them also and the other way around. We implement the singleton pattern design for the player, AudioManager and more so that it can be referred globally and simplify the structure of our game.

CHAPTER VI: INTERFACE DESIGN

I. Regulation

- Make user interfaces consistent
- Accept user's errors
- Confirm detrimental actions
- Disable unsuitable functions contextually
- Use the screen effectively

II. List of UI

No.	Name
1	Main menu Screen
2	Player’s Health bar
3	Player stats sheet
4	Number Pop up
5	Player exit to menu and retreat to hideout
6	Tutorial UI
7	Coins counter
8	NPCShop
9	Interact with object buttons
10	Item tooltip
11	Inventory

III. Details description

1. Main menu screen

The users will enter this scene first when they start the application. The scene will have a looped video as background, 1 text object for the game title, 3 buttons : Play, Tutorial and Quit.

Function:

- Play: Open save file slot for the user to choose
- Tutorial : move the user to tutorial scene
- Quit : quit the application

Upon pressing the Play button, the following gameobjects will get enabled : Save container 1 , 2 and 3,load button , create button, back button

Function:

- Save container Hold the information of the file slot such as coin, unlockable slot, display "none" if no save file was found.
- Back button : disable Save container 1 , 2 and 3, back button.
- Create button:If there is an already existing savefile, alert the user that the save file will get overwritten. Or else , create a new save file, assign the index of the slot to the player, make a new save file and send the user to the hideout scene
- Load button:If there is an already existing savefile, assign the index of the slot to the player, make a new save file and send the user to the hideout scene. Or else, notice the user that there is no save file found. If the player wants to proceed, assign the index of the slot to the player, make a new save file and send the user to the hideout scene.

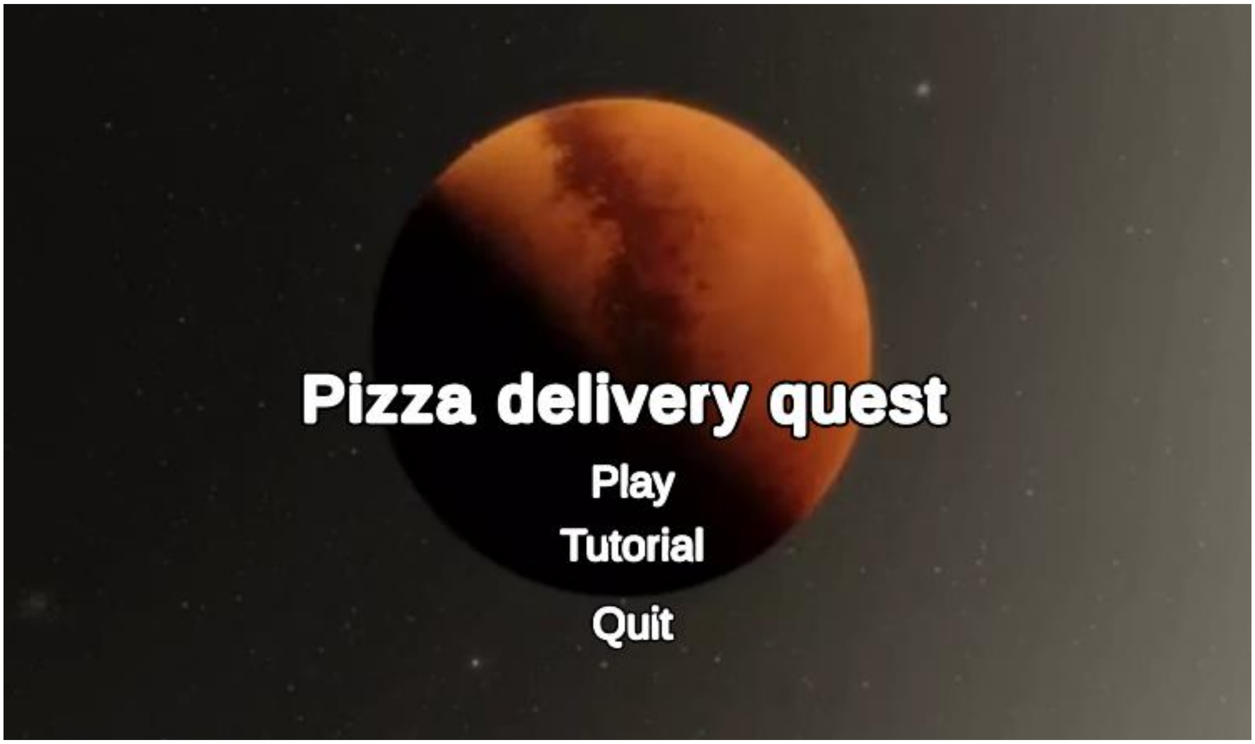


Image 29: Main menu UI

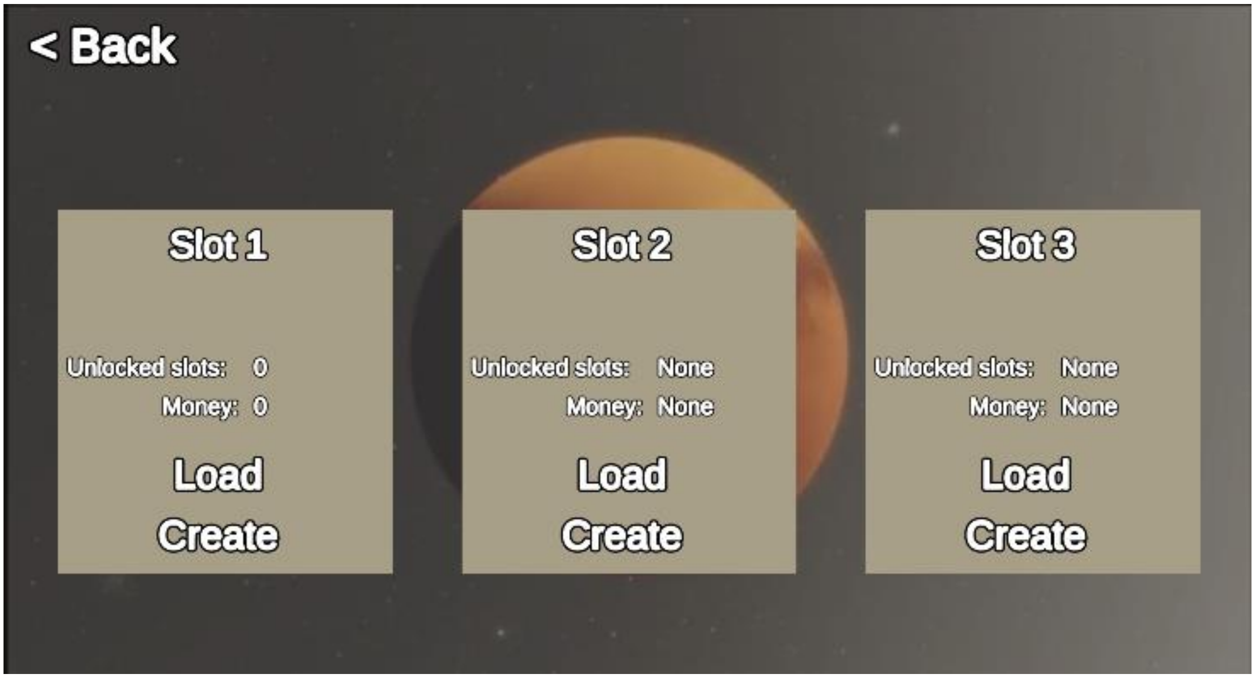


Image 30: Save slots options UI

2. Player's Health bar

2.1. Circular health bar

Active in any scene beside the main menu scene. It is composed of two main parts, the crescent part and the bar part. In the middle of the crescent part there is an illustration of the player and the text displaying current hp and max hp in the middle of the bar part which updates in every frame. The whole thing is stationed in the bottom right corner.

2.2. Status effect bar

Located next to the crescent part and above the bar part of the circular health bar, the status bar will add a status icon when the player receives a status effect and remove the icon whose active time runs out .Those icons are arranged in a grid like layout from left to right and from top to bottom.

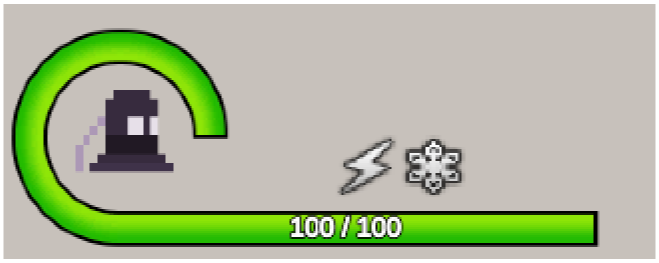
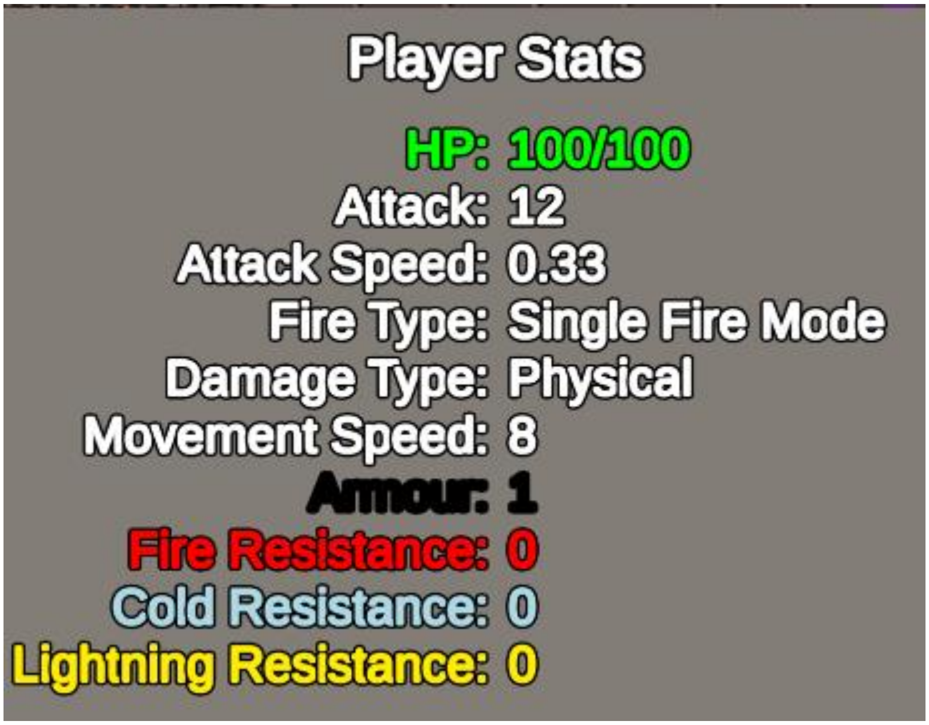


Image 31: Player Health bar and Status Effect bar

3. Player stats sheet

Existing in any scene beside the main menu scene , the player stat sheet is by default disabled. It's activated only when the user is holding the tab button and it holds information like : hp , movespeed , attack speed , damage , armour , fire resistance , cold resistance , lightning resistance, etc in the form of text.



Player Stats

HP: 100/100
Attack: 12
Attack Speed: 0.33
Fire Type: Single Fire Mode
Damage Type: Physical
Movement Speed: 8
Armour: 1
Fire Resistance: 0
Cold Resistance: 0
Lightning Resistance: 0

Image 32: Player Stat Sheet UI

4. Number Pop up

Display a small number indicating the damage an entity is receiving . It appears on the head of the entity and ascends upward then disappears after a fixed amount of time

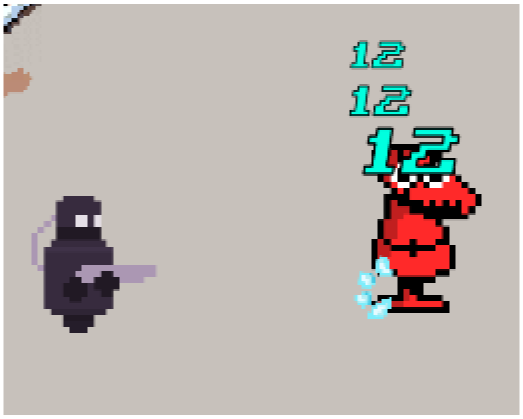


Image 33: Damage Pop Up

5. Player exit to menu and retreat to hideout UI

Esc button will act as a trigger to turn on and off the UI. According to the active scene at the time being, one UI will show up. Player exit to menu will pop up if the player is in the hideout or the tutorial scene , retreat to hideout will pop up if the player is in an instance.

- Player exit to menu
    - Stop the time
    - Button to send the user back to the menu scene

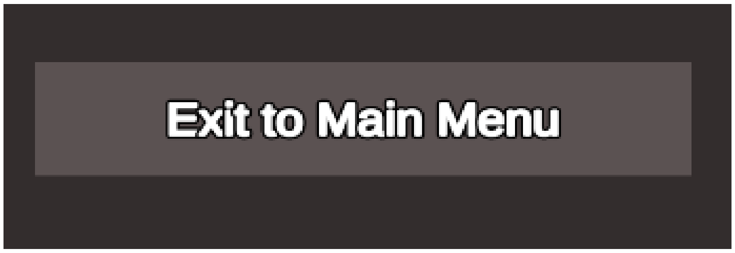


Image 34: Exit to main menu button UI

- Retreat to hideout
    - Text displaying the outcome if the user wants to return back to the hideout
    - Stop the time
    - Yes button to send the user back to the hideout
    - No button disables the retreat to hideout pop up and unfreezes time

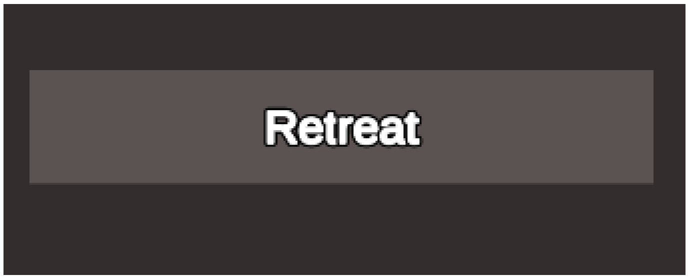


Image 35: Retreat button UI

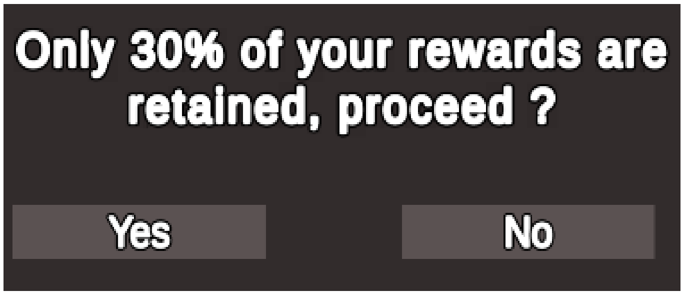

Only 30% of your rewards are retained, proceed ?

Image 36: Retreat Confirmation Button UI

6. Tutorial UI

Return button in the tutorial scene, illustrated as a return arrow and located near the health bar. Pressing the button will reset the tutorial scene.

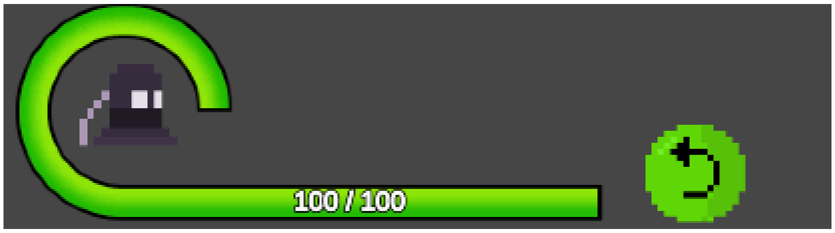


Image 37: Tutorial UI

7. Coins counter

Active in any scene beside menu scene and tutorial scene. An illustration of a coin and text displays the amount the current player is holding. The text is updated every frame. The whole thing is placed in the top left corner of the scene

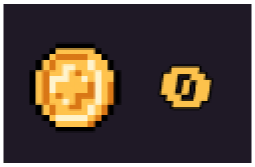
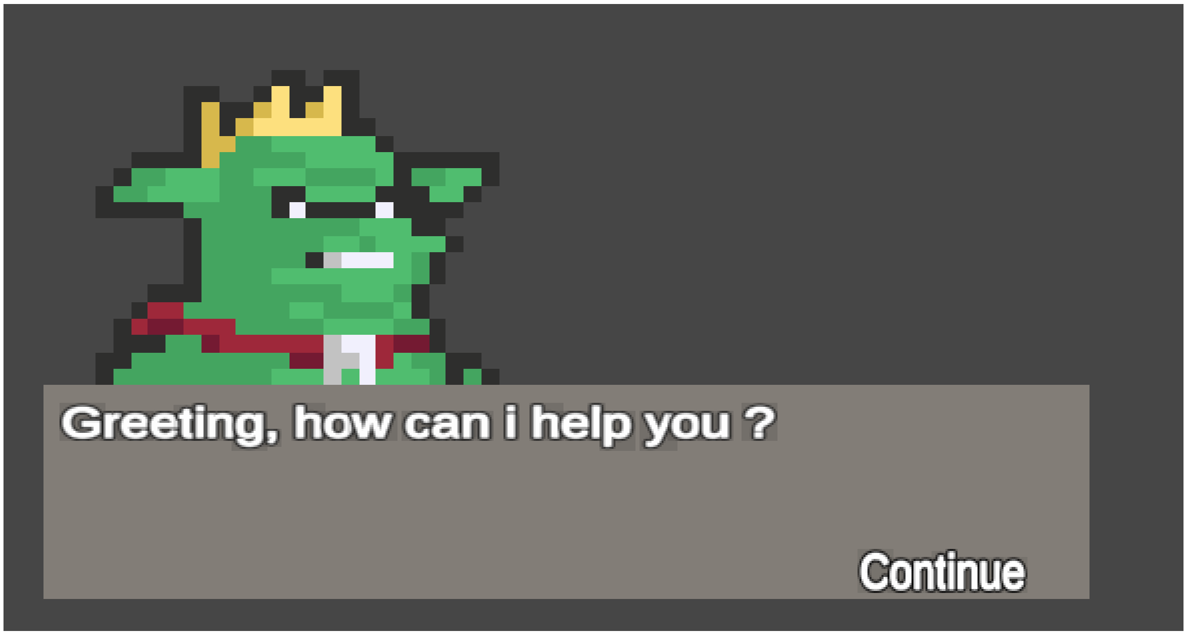


Image 38: Coin Counter UI

8. NPCShop

Only existing in the hideout scene and is activated when the player with it, the NPCShop will contain 2 dialogue containers. Both the dialogue containers are formatted as , the dialogue box, the illustration of the npc on top of the box.

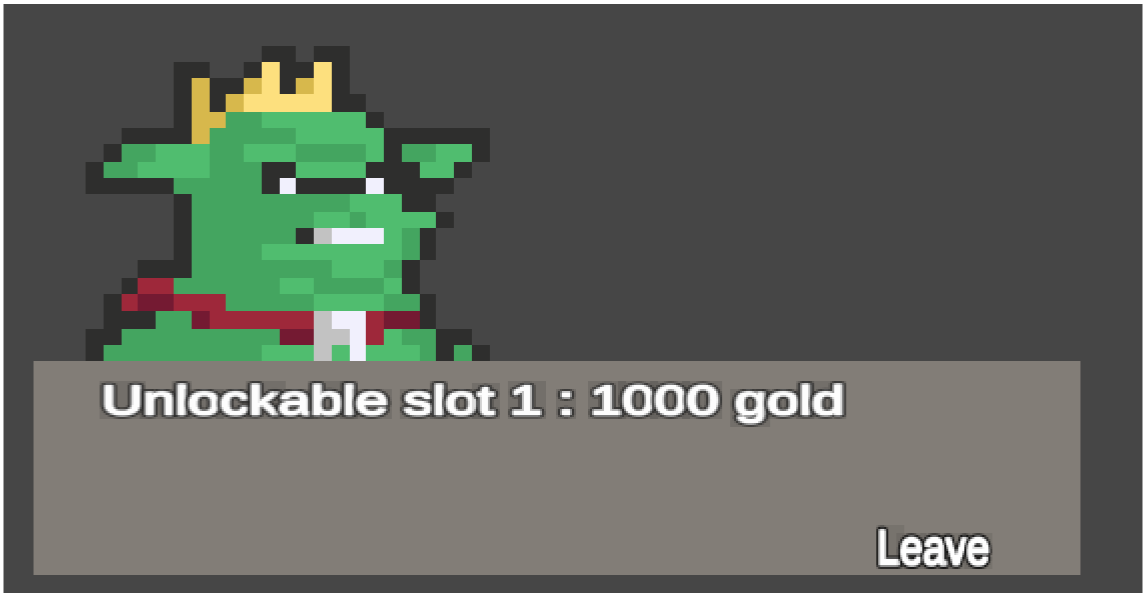
- First dialogue container
  - Text displays the greeting to player
  - Continue button : display the current dialogue box and activate the second one



Greeting, how can i help you ?

Image 39: NPC Dialogue 1

- Second dialogue container
  - Text displays :
    - N-th unlockable slot for the player to player and the price for it
    - If the player has reached the maximum unlockable slot, display " has reached maximum number of unlockable slots"
  - Button to buy the unlockable slot placed near the text displaying the detail
  - Leave button : disable the current dialogue



Unlockable slot 1 : 1000 gold

Image 40: NPC Dialogue 2

9. Interact with object buttons

Illustrated as the key related to the action, the interact with object button is a small icon showing on top of the interactable gameobject when the player is near it and disappears when the player is out of range.

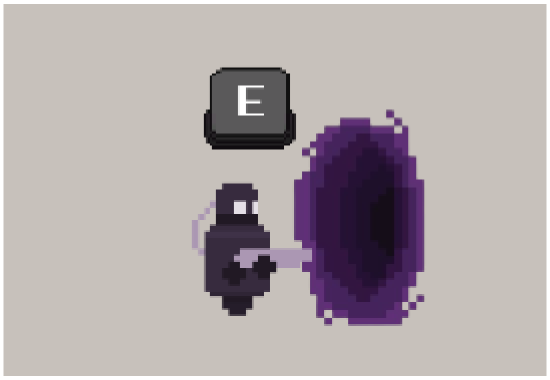
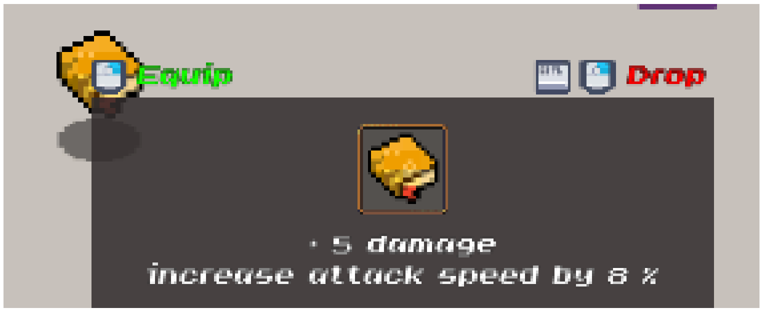


Image 41: Interact Button UI

10. Item tooltip

Display the item modifiers and show what kind of effect will affect player if equiped. There is a "right mouse click" on the top left corner to to show how to equip the item and "ctrl + right mouse click" icon the the top right corner to show how to drop the item on the ground



+ 5 damage
increase attack speed by 8 %

Image 42: Item Tooltip UI

11. Inventory

When player press “I” an Inventory UI appear and show all the items the player currently has equiped and in Inventory. Player can right click an item in the Inventory to equip it to the slot on the left. There are only 3 slots for player to equip so player need to choose what modifiers on an item are more important, and only 20 Inventory slots so player will need to leave some items behind if they’re not as good as others. Player can also right click an equiped item to unequip it. By pressing ctrl + right click on an item inside the Inventory player can throw the item on the ground

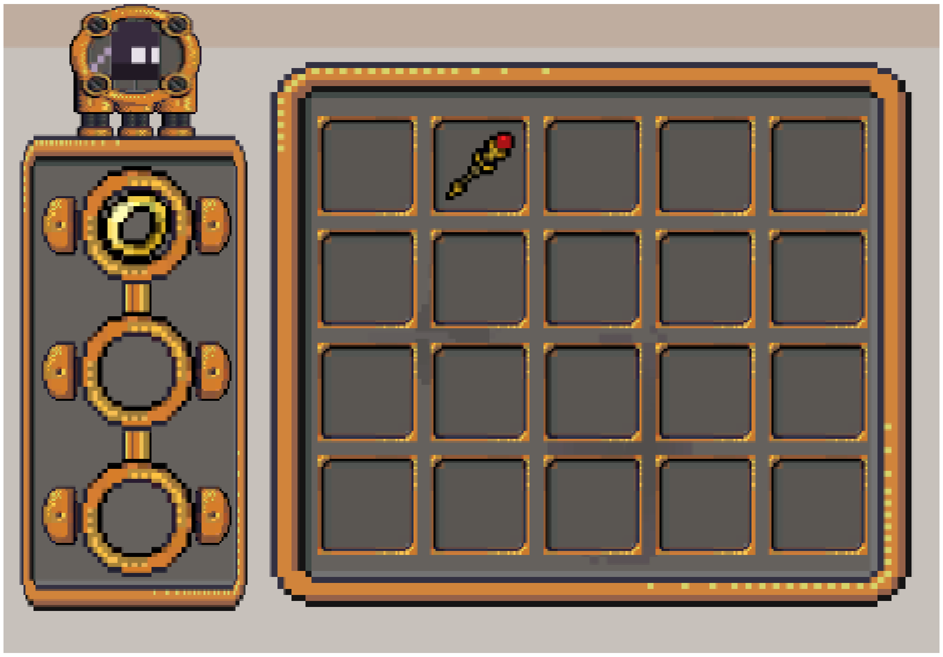


Image 43: Inventory UI

CHAPTER VII: INSTALLATION

I. Unity Hub and Editor

To gain access to the project source code and its editor , such tools are required. The guide to download and installing them visit Installing Unity using the Hub

Unity editor system requirements

Minimum requirements	Windows	macOS	Linux (Support in Preview)
Operating system version	Windows 7 (SP1+), Windows 10 and Windows 11, 64-bit versions only.	High Sierra 10.13+	Ubuntu 20.04, Ubuntu 18.04, and CentOS 7
CPU	X64 architecture with SSE2 instruction set support	X64 architecture with SSE2 instruction set support	X64 architecture with SSE2 instruction set support
Graphics API	DX10, DX11, and DX12-capable GPUs	Metal-capable Intel and AMD GPUs	OpenGL 3.2+ or Vulkan-capable, Nvidia and AMD GPUs.
Additional requirements	Hardware vendor officially supported drivers	Apple officially supported drivers	Gnome desktop environment running on top of X11 windowing system, Nvidia official proprietary graphics driver or AMD Mesa graphics driver. Other configuration and user environment as provided stock with the supported distribution (Kernel, Compositor, etc.)

	For all operating systems, the Unity Editor is supported on workstations or laptop form factors, running without emulation, container or compatibility layer.

II. The game

The game will come in the form of a zip file. Extracting then the game client is ready, the file name is DaGame.exe

The link to download the game :
https://github.com/asdqwe02/Project-NDC/releases/tag/v1.2.5

You can also play the game on some of the io site that host the game like:

- https://simmer.io/@BerryAlmond/project-ndc
- https://berryalmond.itch.io/project-ndc (the password is 5qGQRk7ztNF8S8T)

Recommended system requirements

Operating system	32/64-bit Windows 7 / 8.1 / 10
Processor	Intel Core i5 4460 or over
Memory	6 GB RAM
Graphics	128mb Video Memory, capable of Shader Model 2.0+
DirectX	11
Storage	at least 2GBs

CHAPTER VIII: CONCLUSION

I. Result

- Finished implementing 90% of the planned features
- Easy to understand UI
- Finished testing for the game
- Functional basic game loop
- Can easily expand and add more Buffs, Enemy types, bosses and new features
- Functional Buffs weighting system to use for spawning random Buffs and apply random Quirks
- Destroyable object in the game
- Working Buffs and Enemies
- Finished adding the Bringer of Death Boss Enemy, Bomber Enemy and Shield Enemy
- Sound system is finished and can be easily update if needed
- Item and their modifiers are finished and fully functional

II. Limitations

- Due to time constraints and lack of manpower, we haven't implemented the increase difficultly for next loop if player haven't died
- We lack experience in developing and designing game so the game still feel bland
- Researching take more time than expected
- Lacking sound effect for some Enemy and Bosses, UI
- We haven't followed the model – view – controller structure to the fullest extend
- The game is still simple and straightforward, lacking some exploration aspects of a dungeon crawler type of game
- Some functions still don't work well to our liking and need to be refactored or revamp
- UI is still simplistic
- Lacking menu to control the sound volumn and no menu to change the game resolution

III. Future Scope

- Adding more Enemy types
- More Elemental type
- Adding weapon swapping and more type of weapon and fire type
- More Bosses to fight
- Adding more sound to the game
- Improve UI
- Design more level that promote exploration and looting
- Adding NPC that sell buffs inside an instance instead of just in the hideout
- Add more items type and item modifiers
- Adding more character types with unique skills that player can choose

- Adding more game mechanic that challenge the player skill
- Adding cosmetic items

REFERENCES

1. Unity

- https://docs.unity3d.com/ScriptReference/Quaternion.html
- https://docs.unity3d.com/Manual/script-Serialization.html
- https://docs.unity3d.com/Manual/JSONSerialization.html
- https://docs.unity3d.com/ScriptReference/Application-persistentDataPath.html
- https://docs.unity3d.com/Manual/class-LineRenderer.html
- https://docs.unity3d.com/Manual/Navigation.html
- https://docs.unity3d.com/ScriptReference/Vector3.Lerp.html
- https://docs.unity3d.com/Manual/class-AnimatorController.html
- https://docs.unity3d.com/Manual/Prefabs.html
- https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/UICanvas.html
- https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/UIVisualComponents.html
- https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/UIInteractionComponents.html
- https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/EventSystem.html

2. Unity Basic Tutorial (YouTube)

- https://www.youtube.com/watch?v=FTxQKHG5WCA
- https://www.youtube.com/watch?v=2jTY11Am0Ig
- https://www.youtube.com/watch?v=Lvk6HKmWVwE
- https://www.youtube.com/watch?v=XF05X0fVz_o
- https://www.youtube.com/watch?v=Bc9lmHjqLZc
- https://www.youtube.com/watch?v=ygIC7DcXRCk
- https://www.youtube.com/watch?v=hkaysu1Z-N8&t=728s
- https://www.youtube.com/watch?v=Vrld13ypX_I
- https://www.youtube.com/watch?v=MKjWDtm5eeU
- https://www.youtube.com/watch?v=sPiVz1k-fEs

3. Unity UI tutorial (YouTube/Forums)

- https://www.youtube.com/watch?v=w6_fetj9PIw
- https://www.youtube.com/watch?v=U1LFOTDLh3o
- https://www.youtube.com/watch?v=d_qk7egZ8_c
- https://www.youtube.com/watch?v=_RIsfVOqTaE
- https://www.youtube.com/watch?v=HwdweCX5aMI&t=483s
- https://www.youtube.com/watch?v=YUcvy9PHeXs
- https://answers.unity.com/questions/828315/animated-sprite-in-ui-canvas.html

- https://forum.unity.com/threads/animating-sprites-in-a-canvas.281426/
- https://www.youtube.com/watch?v=VaDhk2eOQXM
- https://www.youtube.com/watch?v=JivuXdrIHK0
- https://answers.unity.com/questions/35579/destroy-object-at-a-set-distance.html
- https://answers.unity.com/questions/1414048/destroy-specific-gameobject-with-name.html
- https://www.codegrepper.com/code-examples/csharp/unity+create+a+child+object
- https://www.youtube.com/watch?v=LNLVOjbrQj4

4. Assets "borrowed"

- https://darkpixel-kronovi.itch.io/mecha-golem-free
- https://penusbmic.itch.io/
- https://elthen.itch.io/2d-pixel-art-goblin-king-sprites
- https://nyknck.itch.io/fx068
- https://nyknck.itch.io/pixel-arteffectfx023
- https://nyknck.itch.io/smoke-effect
- https://nyknck.itch.io/portal
- https://pimen.itch.io/fire-spell-effect-02
- https://elthen.itch.io/pixel-art-destructible-objects
- https://elthen.itch.io/2d-pixel-art-dungeon-collectables

(note: some assets are too old or obscure and we can't find it again)

5. Creating assets/VFX tutorial and tool use to create assets

- https://www.youtube.com/watch?v=rLdA4Amea7Y
- https://www.youtube.com/watch?v=1CXVbCbqKyg
- https://www.youtube.com/watch?v=PoCN8yMoYu0
- https://www.youtube.com/watch?v=OAfpDaAKVSE
- https://www.youtube.com/watch?v=VvjIUIlso9M
- https://www.pixilart.com/draw

6. Path finding

- https://www.youtube.com/watch?v=jvtFUfJ6CP8

7. Buff weight and pool references

- https://www.gamedeveloper.com/design/loot-drop-best-practices
- http://www.vcskicks.com/random-element.php

8. Fire rate reference

- https://joshcgblog.wordpress.com/2015/12/07/2d-unity-game-bullets-and-fire-rate/

9. Player's Dash tutorial

- https://www.youtube.com/watch?v=w4YV8s9Wi3w

   - https://www.youtube.com/watch?v=Bf_5qIt9Gr8

10. *Audio Manager tutorial*
    - https://www.youtube.com/watch?v=6OT43pvUyfY
    - https://www.youtube.com/watch?v=QL29aTa7J5Q
11. *Item tooltip tutorial*
    - https://youtu.be/U1LFOTDLh3o
    - https://youtu.be/HXFoUGw7eKk
    - https://youtu.be/d_qk7egZ8_c
    - https://youtu.be/YUIohCXt_pc

WORKLOAD DISTRIBUTION

Patch 1.0	Nguyễn Nhật Huy	Nguyễn Minh Thiện
Tilemap	60%	40%
Animation	50%	50%
UI	60%	40%
Physics	50%	50%
Interaction	50%	50%
Data design	50%	50%
Testing	30%	70%
Document	50%	50%

Patch 1.2.5	Nguyễn Minh Thiện
Tilemap	100%
Animation	100%
UI	100%
Physics	100%
Interaction	100%
Data design	100%
Testing	100%
Document	100%
Researching	100%