

National College of Ireland

Bachelor of Science (Honours) in Computing (BSHCSD4)

Software Development

2022/2023

Cree Gunning

x19302733

x19302733@student.ncirl.ie

Digital Card Game - Unreal Engine 5

Technical Report

Contents

Executive Summary	2
1.0 Introduction	3
1.1. Background	3
1.2. Aims.....	3
1.3. Technology.....	3
1.4. Structure	4
2.0 System.....	4
2.1. Requirements.....	4
2.1.1. Functional Requirements.....	4
2.1.1.1. Use Case Diagram	5
2.1.1.2. Requirement 1: Card Placement.....	6
2.1.1.3. Description & Priority.....	6
2.1.1.4. Use Case	6
2.1.1.5. Requirement 2: Card Interaction	7
2.1.1.6. Description & Priority.....	7
2.1.1.7. Use Case	7
2.1.1.8. Requirement 3: Combat Phase	9
2.1.1.9. Description & Priority.....	9
2.1.1.10. Use Case	10
2.1.1.11. Requirement 4: Win Condition	11
2.1.1.12. Description & Priority.....	11
2.1.1.13. Use Case	11
2.1.1.14. Requirement 5: Join a Game.....	12
2.1.1.15. Description & Priority.....	12
2.1.1.16. Use Case	12
2.1.1.17. Requirement 6: Adding Cards to Hand	13
2.1.1.18. Description & Priority.....	13
2.1.1.19. Use Case	14
2.1.1.20. Requirement 7: Coin Flip.....	15
2.1.1.21. Description & Priority.....	15
2.1.1.22. Use Case	15
2.1.1.23. Requirement 8: Highlighting Cards	16
2.1.1.24. Description & Priority.....	16

2.1.1.25.	Use Case	16
2.1.1.26.	Requirement 9: Card Preview	17
2.1.1.27.	Description & Priority.....	18
2.1.1.28.	Use Case	18
2.1.1.29.	Requirement 10: Targeting Indicator.....	19
2.1.1.30.	Description & Priority.....	19
2.1.1.31.	Use Case	19
2.1.2.	Data Requirements	21
2.1.3.	User Requirements	21
2.1.4.	Environmental Requirements	21
2.1.5.	Usability Requirements.....	21
2.2.	Design & Architecture	22
2.3.	Implementation	30
2.4.	Graphical User Interface (GUI).....	33
2.5.	Testing.....	36
2.6.	Evaluation	37
3.0	Conclusions	37
4.0	Further Development or Research	37
5.0	References	38
6.0	Project Proposal.....	38
1.0	Objectives.....	39
2.0	Background	40
3.0	State of the Art.....	40
4.0	Technical Approach.....	41
5.0	Technical Details	41
6.0	Special Resources Required	42
7.0	Project Plan	42
8.0	Testing.....	44
7.0	Reflective Journals	44
8.0	Other Materials Used.....	55

Executive Summary

This report will analyse the design and implementation of my Digital Card Game that I have built within Unreal Engine 5 for my final year computing project. It will highlight and

breakdown the core functional requirements that are essential for my game to function at its most basic state. These will include everything from allowing the user to place cards on the game board, how the system handles card interaction, etc. Additional features of the system will also be emphasized that aid the player's overall understanding of the game and its rules, which ultimately adds ease of use functionality to the system.

The results at the end of the project cycle should showcase a fully functioning card game which includes all of the essential mechanics and also additional functionality. It should be able to be played by two players, and a winner should be easily determined by the system when certain criteria/conditions have been met. The results should ultimately show that the system is in a state where core functionality has been achieved, and that development can be continued further into the future.

1.0 Introduction

1.1. Background

I chose to undertake this project as I have an incredible interest in both collectible card games and video games. I've had the idea for this card game for quite some time, and so I think that this is an excellent opportunity to jump in and bring my idea to life. I've already made a paper prototype of this game previously, which greatly aided me in forming the rules behind the game and understanding what the most essential aspects of the game actually are. This knowledge and previous prototype will greatly aid me throughout this project as it ultimately means I already know how the system should function, it's just a matter of adapting the functionality to a digital version of the game and implementing its functionality correctly. I also believe that working on something that I'm genuinely interested in will help motivate me to put my all into the development of this project.

1.2. Aims

The aim of this project is to build a fully functioning digital card game with all the necessary core mechanics included in order for the game to be played out in its entirety and as intended. This includes everything from card interaction, card placement, win conditions, etc. The game will allow two players to face off against each other and battle it out using a variety of different card types with unique abilities.

1.3. Technology

I will use Unreal Engine 5 to achieve my goal of bringing this card game to life. I will use the Blueprint scripting system built into Unreal Engine to handle the game logic. This card game will contain a multitude of different blueprints which will control distinct mechanics for different scenarios and situations, and so it's essential to make sure that they are composed correctly to allow for a smooth and seamless gameplay experience. The blueprint system is based on C++ and you can access this code via an IDE if you wish to do so. However, I chose this technology for the benefit of using its visual scripting features and I intend to stick to this form of programming for the development of the game, as it has a multitude of advantages such as visual testing features, etc.

1.4. Structure

This document contains all the necessary functional requirements that are associated with the design and implementation of my game. These functional requirements will describe the essential components that are needed for this game to operate as intended. They are further broken-down using use case diagrams and a description of the use case. This will clearly outline what each action of the use case represents in relation an actor/user using the system. It also describes the design & architecture of the game system, the implementation of the game, the GUI included in the game, and the techniques I used for testing the functionality of my game.

2.0 System

2.1. Requirements

The following requirements will highlight and breakdown various functionalities of the game system and outline the actions that a user can take throughout the entire duration of a game. It is essential that the user understands how they can interact with the game system and that they gain an adequate understanding of the variety of card options that are presented to them after only a short period of playtime.

2.1.1. Functional Requirements

The following functional requirements will be ranked by highest priority.

1. Card Placement
 - a. Highest Priority
 - i. Allows the user to place cards within the appropriate slots on the game board.
 - ii. This is the highest priority requirement as without having cards on the game board, cards cannot interact with each other, and the game cannot progress and conclude.
2. Card Interaction
 - a. High Priority
 - i. Allows cards to interact with each other based on certain conditions such as card statistics, abilities, and user input.
 - ii. This is just below the priority of Card Placement, as this interaction functionality is the core part of the game that performs the choices that users make. Without this interaction logic, the system will not be able to determine the outcome of the game.
3. Combat Phase:
 - a. High Priority
 - i. System determines when the combat phase is initialised, and which player begins it.
 - ii. This is a high priority requirement as without this phase being initialised, the system can't move onto card interaction and the player's won't be able to attack with their cards.

4. Win Condition:
 - a. High Priority
 - i. System determines which player wins the game based on the outcome of the both user's input and choice of card interactions.
5. Join a Game:
 - a. High Priority
 - i. User navigates through menu to enter a game.
 - ii. High priority as it's extremely important to be able to easily join a game.
6. Adding Cards to Hand
 - a. Medium Priority
 - i. Allows cards to be added to both player's hands at the start of each turn.
 - ii. Without cards being available to players, they cannot be played on the board and the game cannot progress.
7. Coin Flip:
 - a. Medium-Low Priority
 - i. Coin is flipped to determine which player plays their turn 1 first.
 - ii. Not as essential as the above requirements by any means but does allow both users to clearly see the reasoning behind why one player starts their turn before the other.
8. Highlighting Cards:
 - a. Low Priority
 - i. Cards are highlighted with a glowing border if they can be played.
 - ii. Simply aids the user in visually understanding whether a card is playable or not given the current state of the game.
9. Card Preview:
 - a. Low Priority
 - i. When hovering the mouse over a card, a larger preview of the card is displayed.
 - ii. Not as important as the other requirements by any means but does allow both users to clearly read all of the statistics and the abilities of the cards in a larger visual format.
10. Targeting Indicator:
 - a. Low Priority
 - i. When targeting an enemy entity card, a visual indicator is displayed to show exactly what card is targeted.
 - ii. This is a low priority requirement as it just aids the users visually in understanding what attack is about to occur during the combat phase.

2.1.1.1. Use Case Diagram

2.1.1.2. Requirement 1: Card Placement

Use Case for Card Placement via User Input

2.1.1.3. Description & Priority

The user must place specific cards on specific slots each turn by clicking on the card, dragging it to the appropriate slot, and releasing the card. Successful placement of the card is dependent on certain restrictions such as card type, card type slot and Spirit resource available to the player. It has the highest priority as without being able to place a card on the game board, cards then can't interact with each other.

2.1.1.4. Use Case

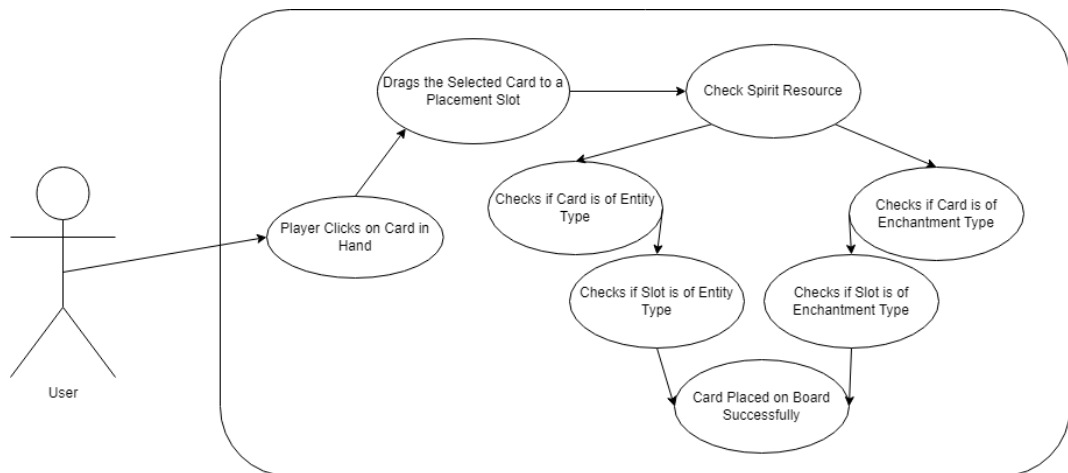
Scope

The scope of this use case is to analyse the steps required to place a card on the game board.

Description

This use case describes the requirements to successfully place a card on the game board taking all restrictions into account.

Use Case Diagram



Flow Description

Precondition

A user must already have a card in their hand in order for it to be then placed on the game board.

Activation

This use case starts when the user clicks on a card in their hand with the mouse.

Main flow

1. The user clicks on a card in their hand.

2. The system will display the card being picked up.
3. The user drags the card to a placement slot.
4. The system checks the user's Spirit resource.
5. The system checks the card type.
6. The system checks the slot type.
7. The system determines that the card and slot type's match up.
8. The user releases the mouse and places the card in the slot.
9. The system recognises the card has been placed and displays it.

Alternate flow

1. The system checks the user's Spirit Resource.
2. The system shows a prompt to the user indicating they don't have enough Spirit to play the chosen card.

Alternate flow

1. The system checks the card type.
2. The system checks the slot type.
3. The system determines that the card and slot type's don't match up.
4. The system returns the card to the user's hand.

Termination

The system displays the card in the slot it was played at on the game board after a successful placement has occurred. (Flipped and face up for Entity Cards, unflipped and face down for Enchantment Cards)

Post condition

The system enters a wait state. The user now has the option to repeat this process and place another card on the board, given that is actually possible (conditions and criteria for placement will be checked again if placement is attempted). Or they the user has the option to end their turn.

[2.1.1.5. Requirement 2: Card Interaction](#)

Use Case for Card Interaction via User Input

[2.1.1.6. Description & Priority](#)

The user must select a card to attack with by clicking on an Entity card on the game board. They then must click on and target an enemy Entity to card, indicating they want to attack that card. The system then subtracts attack from health on both cards. It has a high priority as this is a core component of the gameplay and a key piece of functionality, which ultimately allows the system to identify who has won the game based on the previous interactions.

[2.1.1.7. Use Case](#)

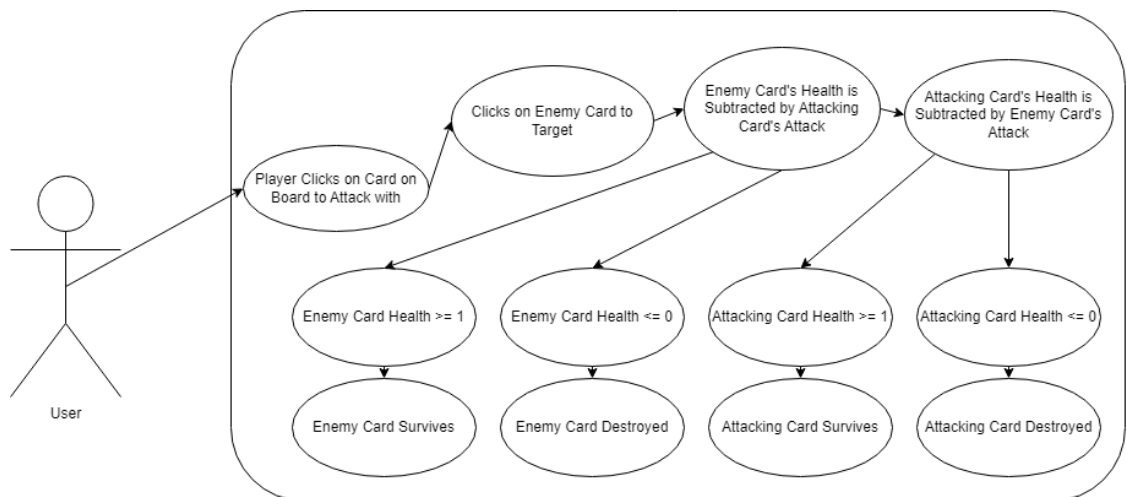
Scope

The scope of this use case is to analyse the steps required to attack an enemy card with a friendly card and have them interact.

Description

This use case describes the requirements to successfully attack an enemy card and have the system determine the outcome of this interaction.

Use Case Diagram



Flow Description

Precondition

A user must already have a friendly Entity card on the board to attack with it.

Activation

This use case starts when the user clicks on an Entity card on the board that they want to attack with using their mouse.

Main flow

1. The user clicks on a friendly card on board to attack with.
2. The user clicks on an enemy card to target.
3. The system subtracts the health of the targeted enemy card by the attack of the attacking card.
4. The system subtracts the health of the attacking card by the attack of the targeted enemy card.
5. The system checks the current health of the targeted enemy card.
6. The system checks the current health of the attacking card.
7. The system determines that the health of the enemy card is ≥ 1 .
8. The system determines that the health of the attacking card is ≥ 1 .
9. The system recognises that the enemy card survives.
10. The system recognises that the attacking card survives.

Alternate flow

1. The system checks the current health of the targeted enemy card.
2. The system checks the current health of the attacking card.
3. The system determines that the health of the enemy card is ≥ 0 .
4. The system determines that the health of the attacking card is ≥ 1 .
5. The system recognises that the enemy card is destroyed and indicates this.
6. The system recognises that the attacking card survives.

Alternate flow

1. The system checks the current health of the targeted enemy card.
2. The system checks the current health of the attacking card.
3. The system determines that the health of the enemy card is ≥ 1 .
4. The system determines that the health of the attacking card is ≥ 0 .
5. The system recognises that the enemy card survives.
6. The system recognises that the attacking card is destroyed and indicates this.

Alternate flow

1. The system checks the current health of the targeted enemy card.
2. The system checks the current health of the attacking card.
3. The system determines that the health of the enemy card is ≥ 0 .
4. The system determines that the health of the attacking card is ≥ 0 .
5. The system recognises that the enemy card is destroyed and indicates this.
6. The system recognises that the attacking card is destroyed and indicates this.

Termination

If a card was destroyed, the system indicates that the card was destroyed. If it survived, it simply continues to be displayed with its new health value.

Post condition

The system enters a wait state until the calculations from the interaction has been resolved. The turn is then passed over to the opponent.

[2.1.1.8. Requirement 3: Combat Phase](#)

Use Case for Determining When The Combat Phase Should be Initialised and Which User Attacks First

[2.1.1.9. Description & Priority](#)

Once both users have played through their five turns, the combat phase begins. Enchantment cards are be flipped and the user who attacks first is determined by the system. It has a high priority as this leads into the card interaction requirement which is a core aspect of the game where a huge part of user input and decision takes place.

2.1.1.10. Use Case

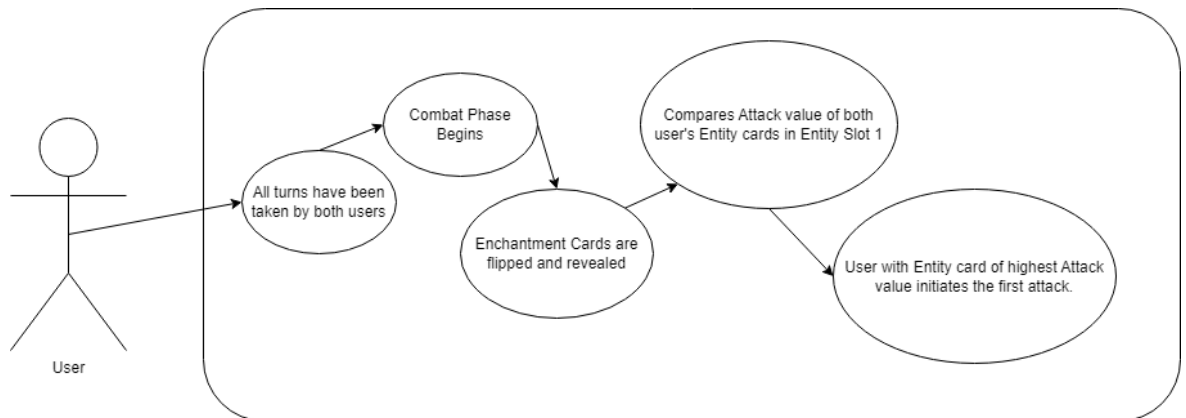
Scope

The scope of this use case is to analyse the steps required for the system to begin the combat phase and to identify the user who attacks first.

Description

This use case describes the requirements of how the system successfully determines when to begin the combat phase and how it determines the user who attacks first.

Use Case Diagram



Flow Description

Precondition

User is on their final turn.

Activation

Both users have played through all five of their turns.

Main flow

1. The system initiates the combat phase.
2. The system checks for enchantment cards on board and reveals them face up.
3. The system checks both card's attack value in Entity Slot 1.
4. The system compares these values and assigns attack priority to the user whose card has the higher attack value.

Termination

The system displays which user's turn it is to attack first.

Post condition

The system enters a wait state until the user initiates an attack or ends their turn.

2.1.1.11. Requirement 4: Win Condition

Use Case for Determining Which Player Wins the Game

2.1.1.12. Description & Priority

The system checks the board state to see if either user has successfully destroyed all of the opponent's cards, whilst still having at least one of their own cards in play. Alternatively, it also checks to see if either user's God card has been defeated. These are the conditions that must be met in order to assign a winner and loser. It has a high priority as it's important that the system clearly identifies the winner of the game and displays this information to the user. The game must resolve at some stage and not continue on forever.

2.1.1.13. Use Case

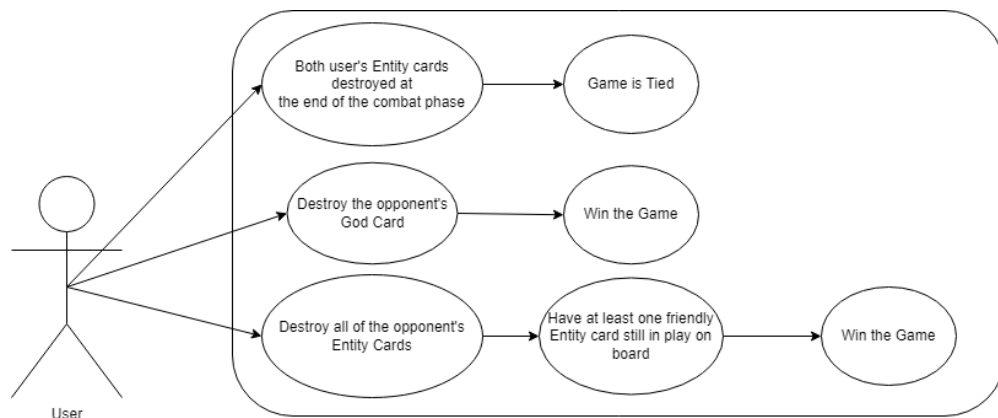
Scope

The scope of this use case is to analyse the steps required for the system to identify the winner of the game.

Description

This use case describes the requirements of how the system successfully determines who the winner of the game is based on certain criteria that has been met based on previous actions of the user.

Use Case Diagram



Flow Description

Precondition

Users must have made previous actions with their cards in order for the system to determine who has won the game.

Activation

This use case starts when the system either of the three conditions have been met and identified.

Main flow

1. The system recognises that the user has destroyed all the opponent's Entity cards.
2. The system recognises that the user still has at least one friendly Entity card alive and in play on their side of the board.
3. The system identifies from these previous conditions that the user has in fact won the game and it displays this to the user.

Alternate flow

1. The system recognises that the user has destroyed the opponent's god card.
2. The system identifies from this previous condition that the user has in fact won the game and it displays this to the user.

Alternate flow

1. The system recognises that both user's Entity cards were destroyed at the same time at the end of the combat phase.
2. The system identifies from this previous condition that both users have tied, and it displays this to both users.

Termination

The system displays the result of the game to both users. Winner is displayed to the winning user and Loser is displayed to the losing user.

Post condition

The system enters a wait state then the game prompts the user if they'd like to exit the game or return to the main menu.

[2.1.1.14. Requirement 5: Join a Game](#)

Use Case for Navigating Through Menu and Joining a Game

[2.1.1.15. Description & Priority](#)

User either creates/hosts their own game and invites another user, or a user joins a game which is hosted by another user. It has a high priority as it is crucial to ensure that a game can be joined with absolute ease.

[2.1.1.16. Use Case](#)

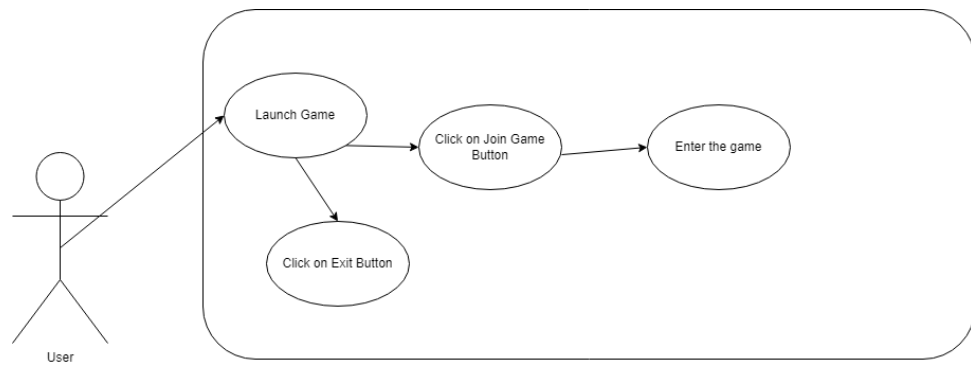
Scope

The scope of this use case is to analyse the steps required for a user to join a game.

Description

This use case describes the requirements of how the system allows the user to navigate through the menu and join a specific game.

Use Case Diagram



Flow Description

Precondition

User's computer must be turned on.

Activation

The game is launched by the user.

Main flow

1. The user navigates through the menu and selects the Join Game Button.
2. The user joins the game.

Alternate flow

1. The user navigates through the menu and selects the Exit Game Button.
2. The system shuts down and the user closes out of the game.

Termination

The system will indicate to the player that they have joined the game and will show a "Waiting for Player" message if another user hasn't joined yet.

Post condition

The system enters a wait state until two users have successfully joined a game.

2.1.1.17. Requirement 6: Adding Cards to Hand

Use Case for Cards being added to user's hand

2.1.1.18. Description & Priority

The system adds three cards to the user's hand at the beginning of the first turn from the user's deck. From turns 2-5, the system adds two cards to the user's hand at the beginning of the turn from the user's deck. It has a high priority as it's essential for the user to consistently have cards available to them throughout the game so that they have the option to be played. Cards are what this game is built around after all.

2.1.1.19. Use Case

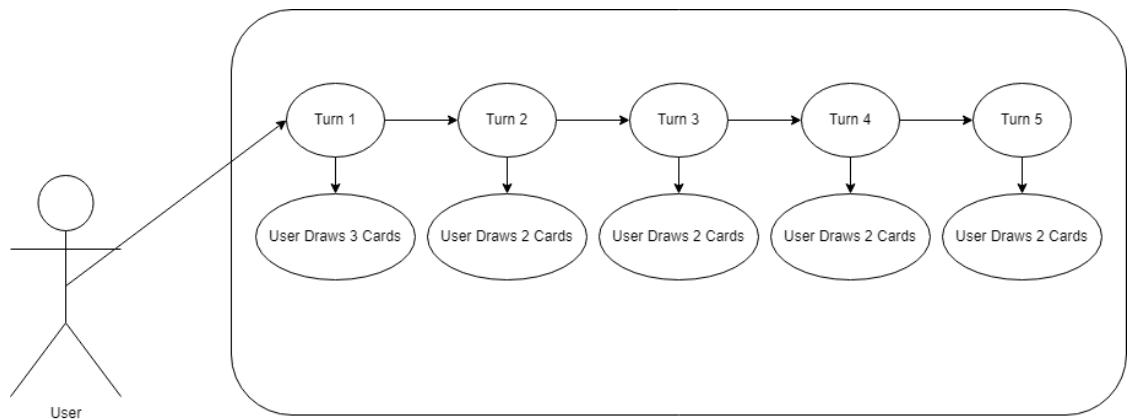
Scope

The scope of this use case is to analyse the steps required for the system to add cards to the user's hand.

Description

This use case describes the requirements of how the system successfully adds cards to the user's hand.

Use Case Diagram



Flow Description

Precondition

It must be the user's turn in order for cards to be added to their hand.

Activation

This use case starts when the system indicates to the user that it is their turn 1.

Main flow

1. The system recognises it's currently the user's turn 1 and displays this to them.
2. The system adds 3 cards to the user's hand.
3. The system recognises it's currently the user's turn 2 and displays this to them.
4. The system adds 2 cards to the user's hand.
5. The system recognises it's currently the user's turn 3 and displays this to them.
6. The system adds 2 cards to the user's hand.
7. The system recognises it's currently the user's turn 4 and displays this to them.
8. The system adds 2 cards to the user's hand.
9. The system recognises it's currently the user's turn 5 and displays this to them.
10. The system adds 2 cards to the user's hand.

Termination

Cards are displayed in the user's hand after they've been drawn from the deck.

Post condition

The system enters a wait state. The user can play a card from their hand that they've drawn if they choose, or they can end their turn.

2.1.1.20. Requirement 7: Coin Flip

Use Case for Determining Which User Takes Their Turn First

2.1.1.21. Description & Priority

Once both users have entered the game, a coin flip begins, and the result determines which user will begin the game and play the first turn. It has a medium-low priority as it's not essential for the system to show the user why a specific user gets to start the game before the other user. This could easily just be randomized at the start of each game.

2.1.1.22. Use Case

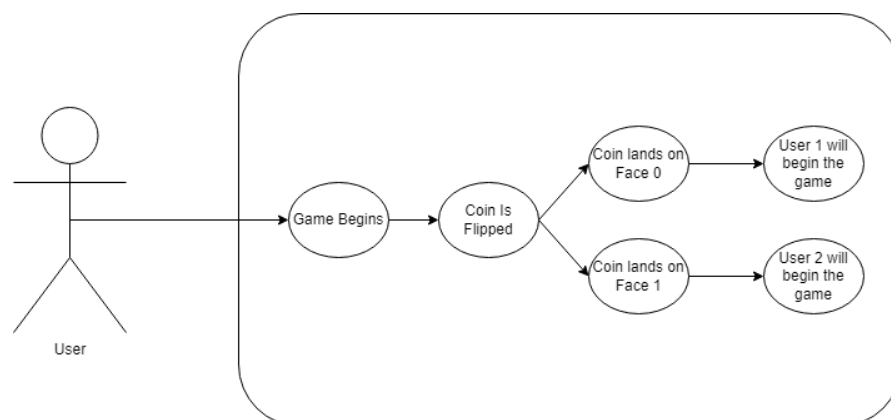
Scope

The scope of this use case is to analyse the steps required for the system to determine which user begins the game first.

Description

This use case describes the requirements of how the system successfully determines who the starting user will be based on the outcome of a coin flip.

Use Case Diagram



Flow Description

Precondition

Both users must be in the game.

Activation

The system initiates starting game phase.

Main flow

1. The system checks the outcome of the coin flip action.
2. The system determines that the result was the coin landed on the Face with 0.
3. The system informs user 1 that it is their turn.

Alternate flow

1. The system determines that the result was the coin landed on the Face with 1.
2. The system informs user 2 that it is their turn.

Termination

The system displays the result of the coin flip to both users and indicates who will be taking the first turn.

Post condition

The system enters a wait state until the user makes an action on their current turn or ends their turn.

[2.1.1.23. Requirement 8: Highlighting Cards](#)

Use Case for Determining which cards in the user's hand should be highlighted.

[2.1.1.24. Description & Priority](#)

Cards are highlighted in each user's hand when they are able to be put into play based on the available resources the user has. This is a low priority use case as it's not necessary for the game to function correctly, however it does aid the user in understanding when cards are considered to be playable and when they are not.

[2.1.1.25. Use Case](#)

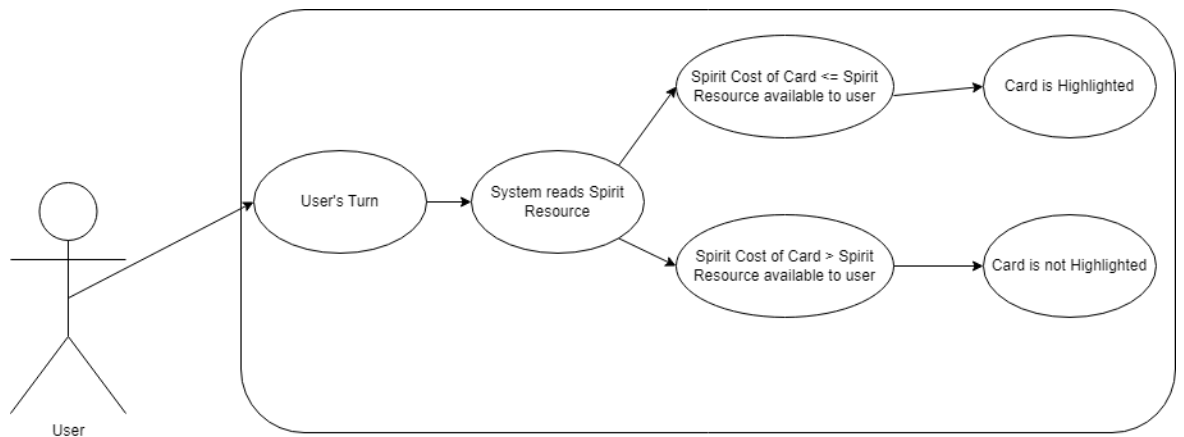
Scope

The scope of this use case is to analyse the steps required for the system to determine which cards should be highlighted in a user's hand.

Description

This use case describes the requirements of how the system successfully determines what cards are highlighted in the user's hand based on the resources they have available to them.

Use Case Diagram



Flow Description

Precondition

Cards must be present within user's hand.

Activation

The system recognises it's the user's turn.

Main flow

1. The system reads the amount of Spirit resource that is available to the user.
2. The system checks the Spirit cost of a card against the amount of Spirit the user has available to spend.
3. The system checks to see if the cost of the card is \leq the amount of Spirit available.
4. The system highlights the card.

Alternate flow

1. The system checks to see if the cost of the card is $>$ the amount of Spirit available.
2. The system does not highlight the card.

Termination

The system highlights all cards in hand that are playable that meet the required conditions.

Post condition

The system enters a wait state until the user makes an action on their current turn or ends their turn.

2.1.1.26. Requirement 9: Card Preview

Use Case for Determining when to show an enlarged preview of a card.

2.1.1.27. Description & Priority

Cards will have a large preview when the user's mouse hovers over a card. This is a low priority use case as it's not necessary for the game to function correctly, however it does aid the user in reading the statistics and abilities of cards clearly and precisely rather than just looking at the on the game board.

2.1.1.28. Use Case

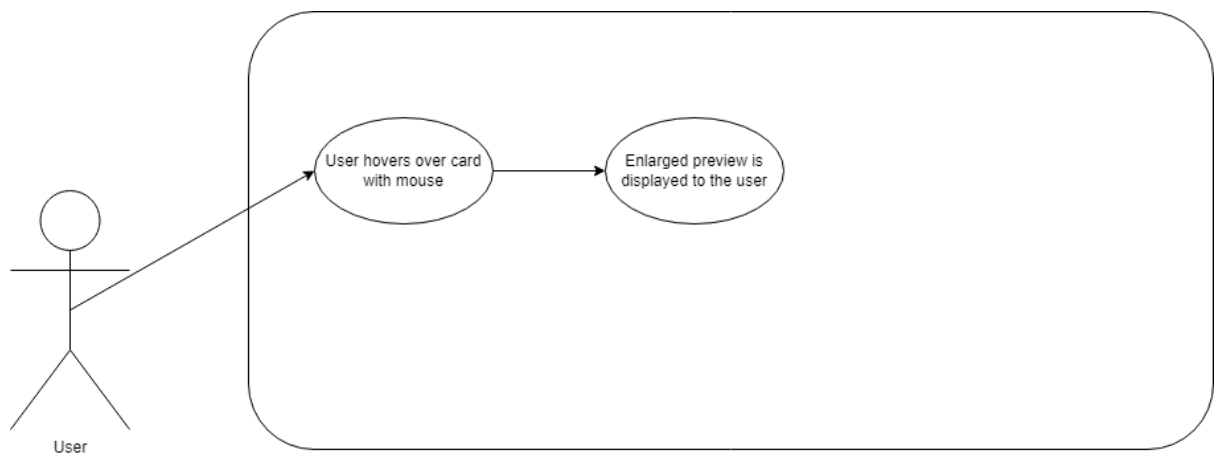
Scope

The scope of this use case is to analyse the steps required for the system to determine when a card preview should be displayed to the user.

Description

This use case describes the requirements of how the system successfully determines what card should be previewed based on whether or not the user's mouse is hovering over said card.

Use Case Diagram



Flow Description

Precondition

Cards must be present in the user's hand and/or the game board.

Activation

The system recognises the user is moving their mouse.

Main flow

1. The system checks to see if the user's mouse is hovering over a card.
2. The system recognises that a card is present at the mouse cursor location.
3. The system displays the enlarged card preview to the user.

Alternate flow

1. The system does not read a card at the mouse cursor location.

2. The system does not display any sort of preview to the user.

Termination

The system stops previewing a card to the user when the mouse cursor is removed from the card location.

Post condition

The system enters a wait state until another card is hovered over by the user's mouse cursor.

[2.1.1.29. Requirement 10: Targeting Indicator](#)

Use Case for Determining when to show the targeting indicator for an attacking card.

[2.1.1.30. Description & Priority](#)

When a friendly card is clicked, a visual targeting indicator is displayed to the user which attaches to the position of the mouse cursor. This enables both users to better understand what card is attacking and what card is being targeted by said attacking card. This use case has low priority as this is simply a visual aid that helps the plays clearly see what action is taking place during the combat phase. It does not affect the functionality of the action.

[2.1.1.31. Use Case](#)

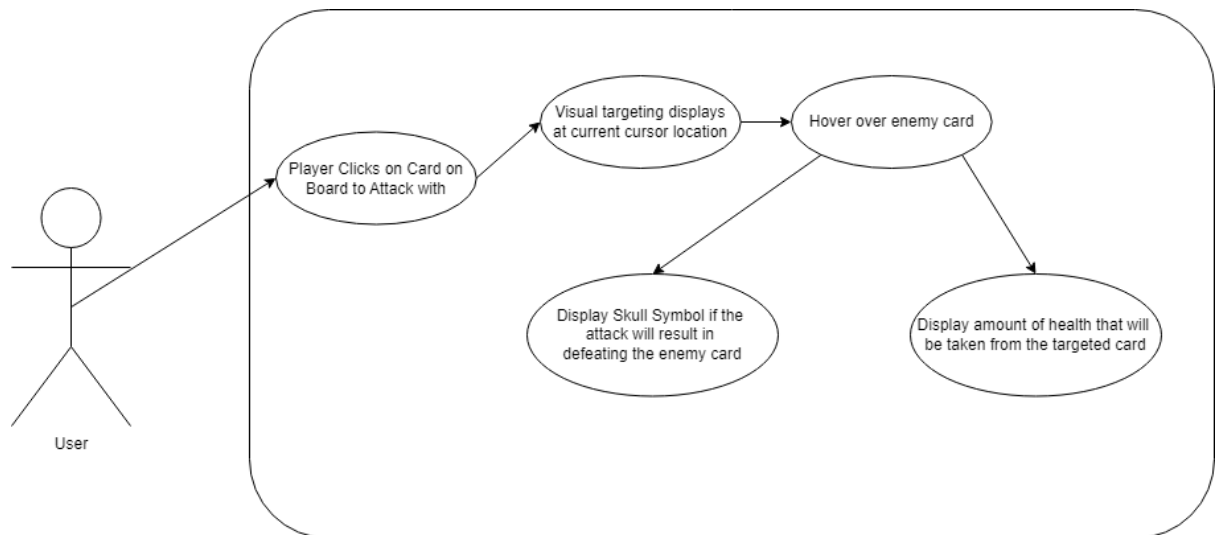
Scope

The scope of this use case is to analyse the steps required for the system to determine when the visual indicator for a targeted card should be enabled.

Description

This use case describes the requirements of how the system successfully determines when to display the visual targeting indicator based on which card has been selected as the attacking card and the current location of the mouse cursor.

Use Case Diagram



Flow Description

Precondition

Combat phase must be initiated.

Activation

The system recognises the user can make an attack.

Main flow

1. The system recognises the user clicked on a friendly card to attack with.
2. The system displays the visual targeting indicator at the current location of the mouse cursor.
3. The system recognises the user is hovering over an enemy card.
4. The system displays a skull beside the targeting indicator if the attack will result in the targeted card becoming defeated.

Alternate flow

1. The system displays the amount of health that will be subtracted from the targeted cards current health if the attack will not result in the targeted card becoming defeated.

Termination

The system stops displaying the visual targeting indicator once an attack has been initiated by the user clicking on an enemy card.

Post condition

The system enters a wait state until another card is selected to attack an opposing card.

2.1.2. Data Requirements

Requirement 1: Data for card statistics (Spirit Cost (Integer), Attack Value (Integer), Health Value (Integer)). Data formed by me.

Requirement 2: Data for card text and ability (Name (String), Description (Body), Type (String)). Data formed by me.

2.1.3. User Requirements

Requirement 1: The user can join a game.

Requirement 2: The user can play a card from their hand.

Requirement 3: The user can place a card on the board.

Requirement 4: The user can end their turn.

Requirement 5: The user can select a card to attack with.

Requirement 6: The user can target a specific card.

2.1.4. Environmental Requirements

User must use a capable PC that is able to support any graphical demands that the game may throw at the system.

2.1.5. Usability Requirements

These requirements outline key usability features that will aid the user in understanding what should be done at a certain time in order to progress through a game.

Requirement 1: The user can use their mouse to hover over any card in their hand or on the board. This will show them an enlarged card preview which will aid them in learning the abilities of cards and familiarizing themselves with the rules and card interactions within the game.

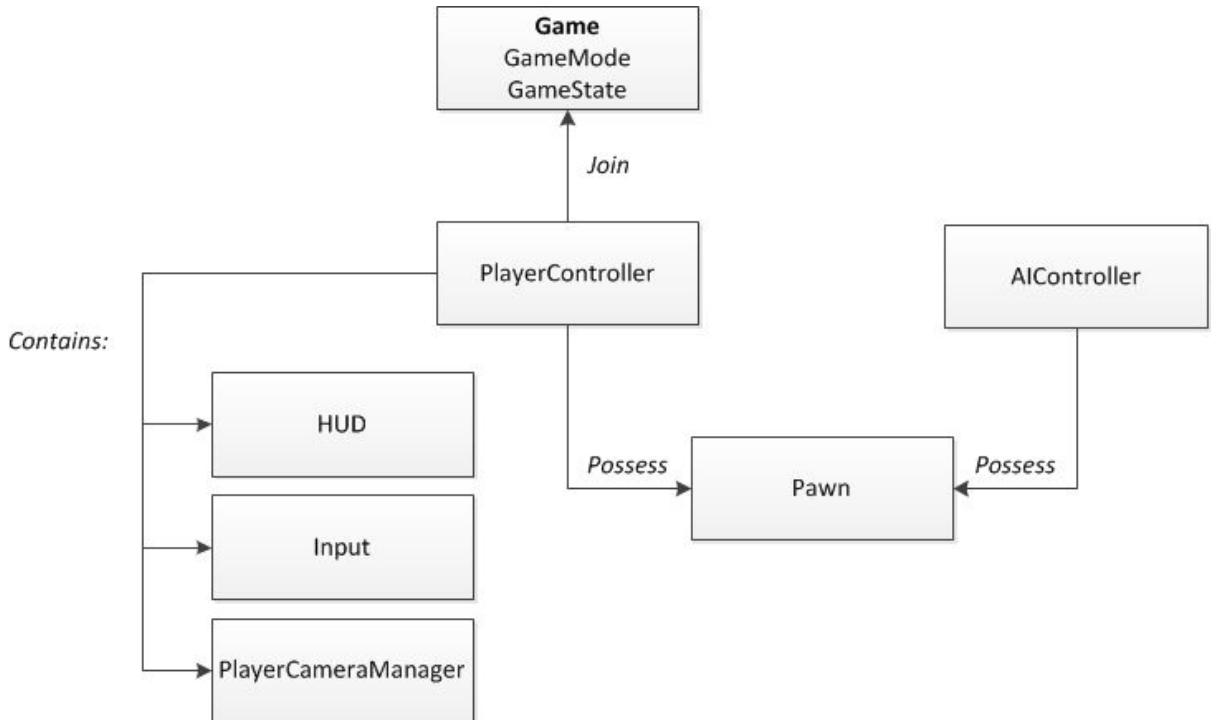
Requirement 2: The user can easily pick up and place cards on the board. Cards will have a glowing border, clearly indicating to the user which cards can be played from the hand given the appropriate criteria is met to allow them to be played. The card slot where the player must place their chosen card will also glow in some regard, again, as a clear indicator for the player to know where the card must be placed.

Requirement 3: The user is prompted with a message in the top right when they try to perform an action that isn't permitted by the system. For example, when trying to play a high Spirit card on turn 1, the user will be prompted with a message stating "Not enough Spirit".

2.2. Design & Architecture

As I have previously mentioned, this project is being developed within Unreal Engine 5 which is an industry standard game engine. It uses a blueprint visual scripting system as it's "language". Blueprints make up all of the functionality of how the game functions as a whole.

Here is a breakdown of the architecture of the core gameplay classes within Unreal Engine 5.



A GameMode and GameState together make up a game. PlayerControllers are associated with players who join the game. These PlayerControllers give players the ability to control in-game pawns (character, sprite, anything) so they can have actual physical presences in the level. In my instance, this is "controlling" where cards are placed. Input controls, a heads-up display, or HUD, and a PlayerCameraManager for managing camera views are also provided by PlayerControllers.

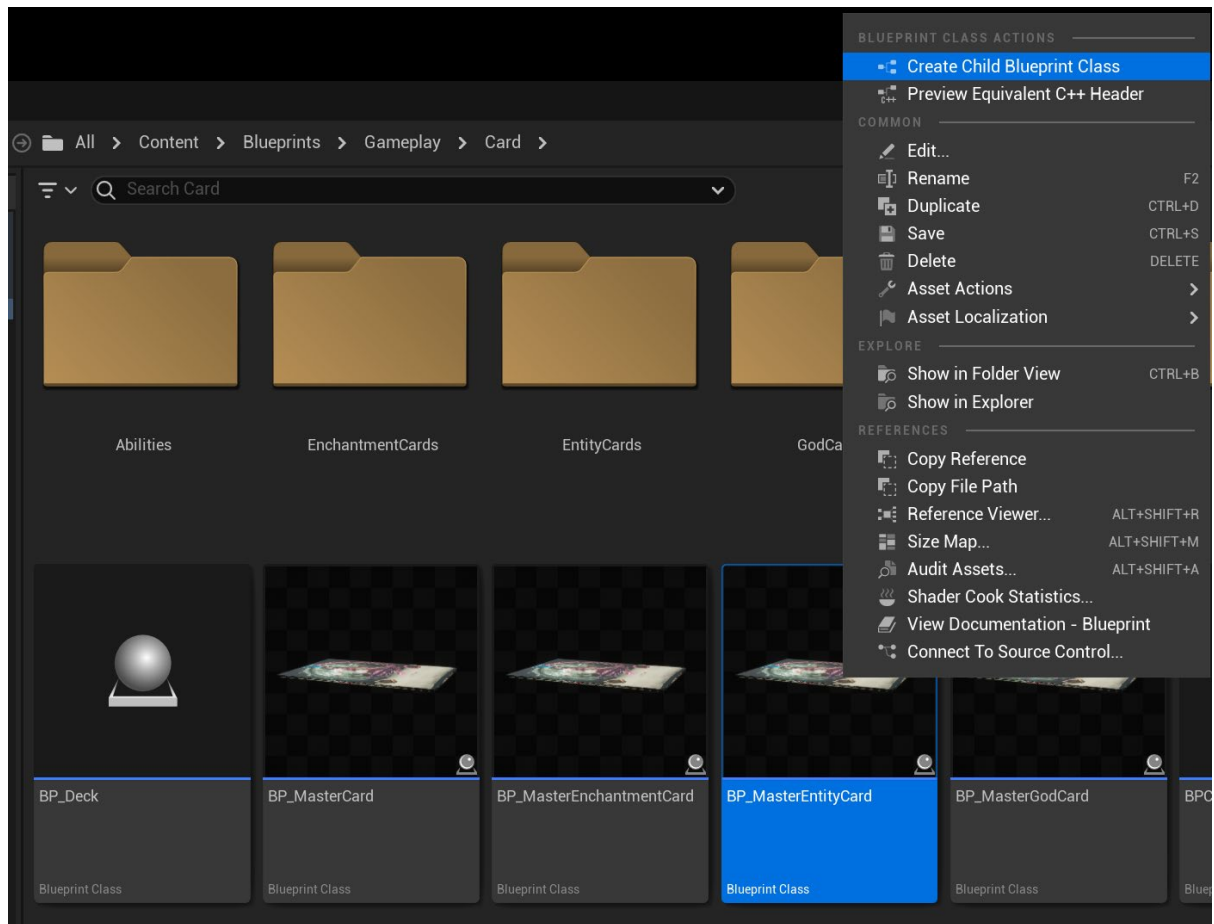
In this section, I'm going to break down the different blueprint folders to showcase exactly what goes into making a digital card game.

Card Blueprints

Entity Cards: Entity cards can be found in the following path.



To create a new entity card, I can either duplicate one of the existing entity cards or make a new child blueprint of the BP_MasterEntityCard located in the Card folder, as shown below.



Once a new entity card is opened, I can first adjust its 3 main values on the blueprint's class defaults:

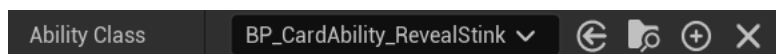
▼ Default	
Spirit Cost	3
Health	4
Attack	3

Spirit Cost: The spirit cost to place this card

Health: The total starting health of the card

Attack: The total starting attack of the card

Besides the 3 values above, every card can also be assigned an ability class, as shown below



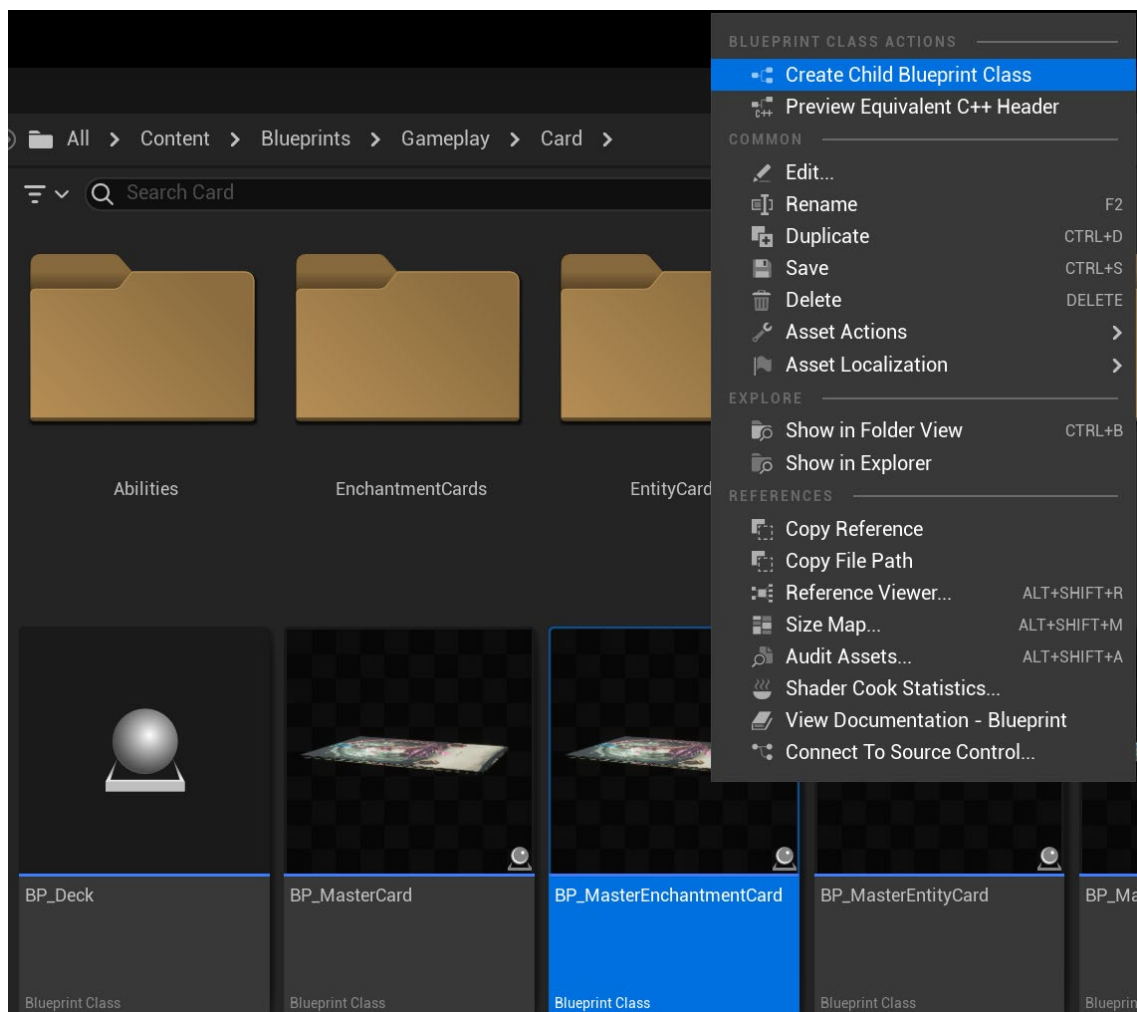
Abilities will be explained in more detail below. If I want an entity card to be able to also attack a god card, I can simply check the Is Relentless Boolean.

Is Relentless

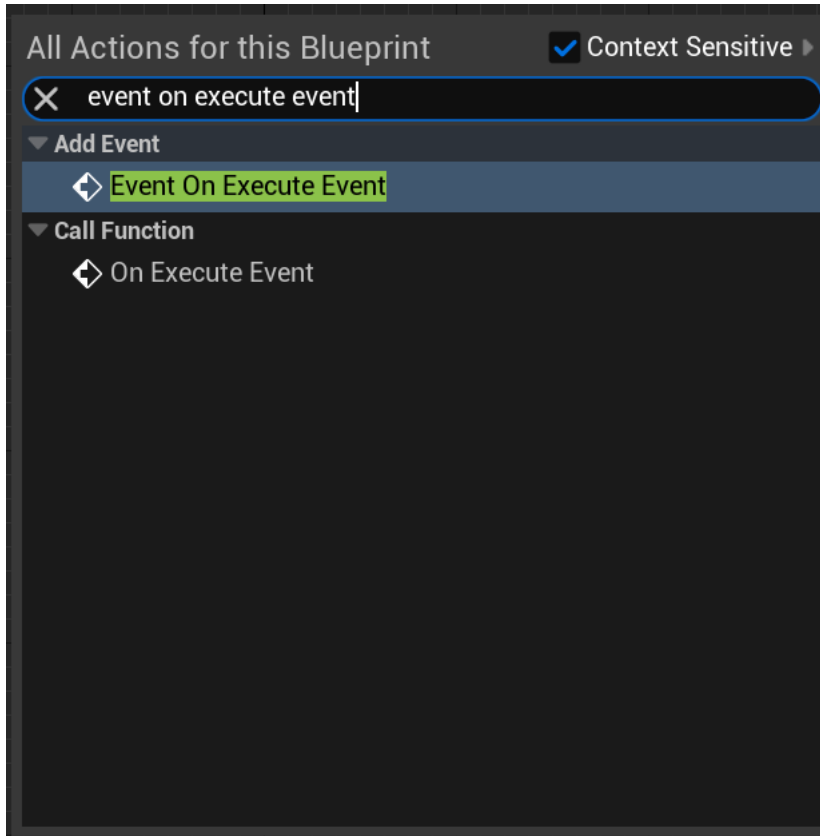
Enchantment Cards: Enchantment cards can be found in the following path.

All > Content > Blueprints > Gameplay > Card > EnchantmentCards

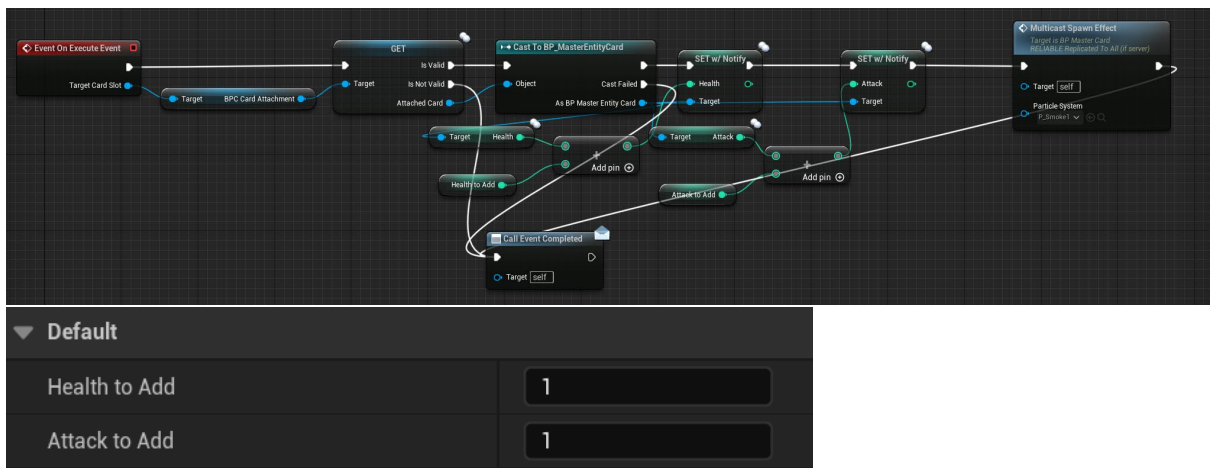
To create a new enchantment card, I can either duplicate one of the existing enchantment cards or make a new child blueprint of the BP_MasterEnchantmentCard located in the Card folder, as shown below



Once a new enchantment card is opened, I can program the new custom event logic for this enchantment card. To do this, I must open the Event Graph and override the event called "OnExecuteEvent", as shown below



Afterwards, I can simply put any logic I want this enchantment card to do.

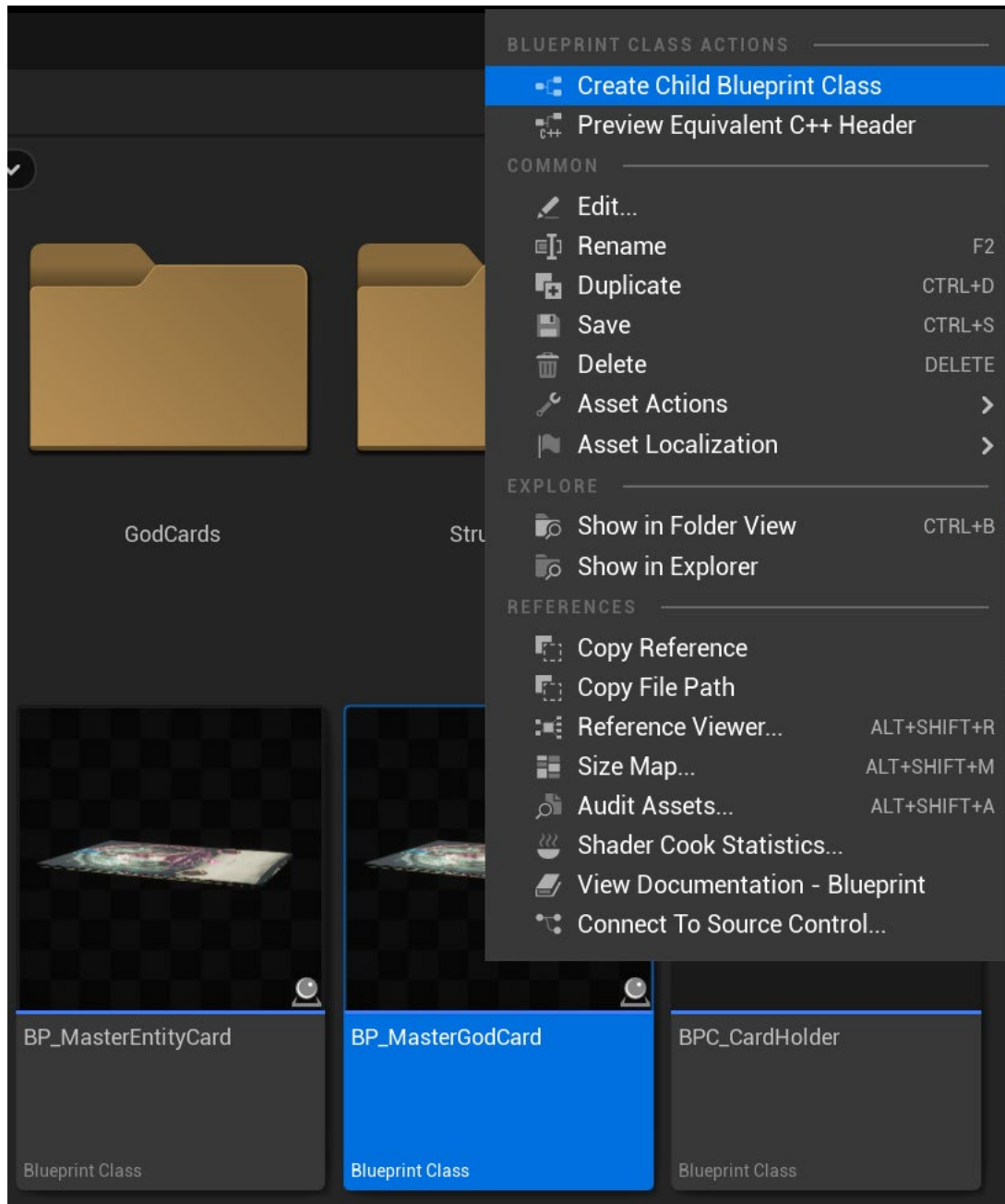


In this example, the Enchantment card is adding 1 Attack (Strength) and 1 Health to its corresponding Entity card.

God Cards: God cards can be found in the following path.



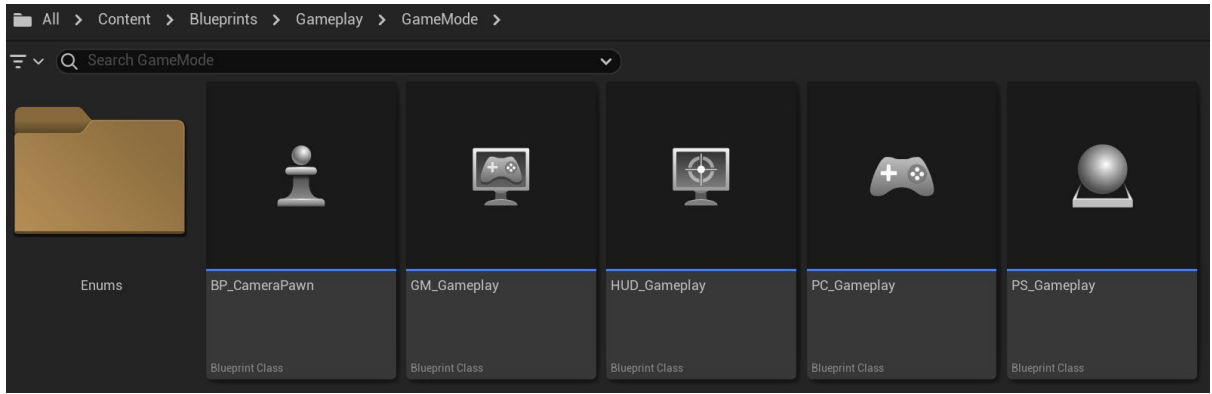
To create a new god card, I can either duplicate one of the existing event cards or make a new child blueprint of the BP_MasterGodCard located in the Card folder, as shown below



God cards are quite similar to entity cards, meaning I can again open and modify their values and ability. Since god cards have starting/passive abilities that execute at the start of the combat phase, I must ensure their assigned ability init type is set to “Lead”, as this ability triggers when a card is put into play. In the case of the god card, this card is put into play at the start of every game.

Gamemode Blueprints

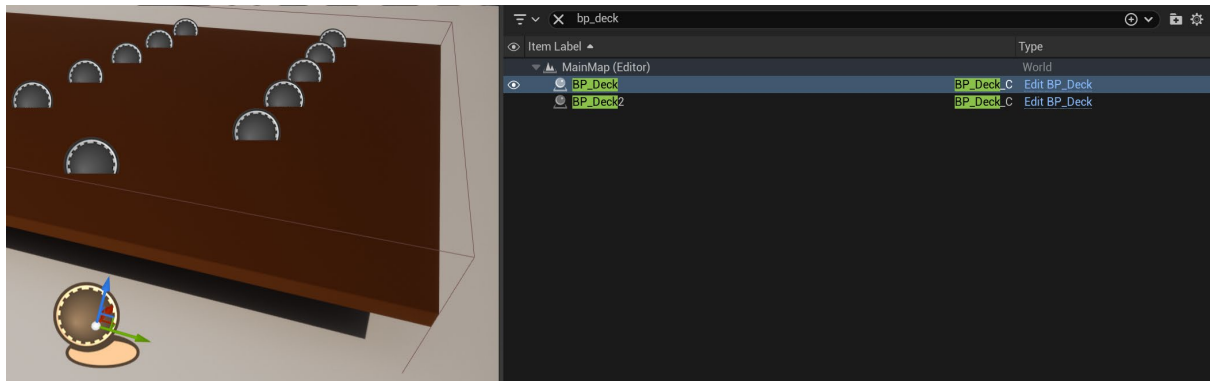
All the main game mode logic can be found in the GM_Gameplay gamemode located in the path below.



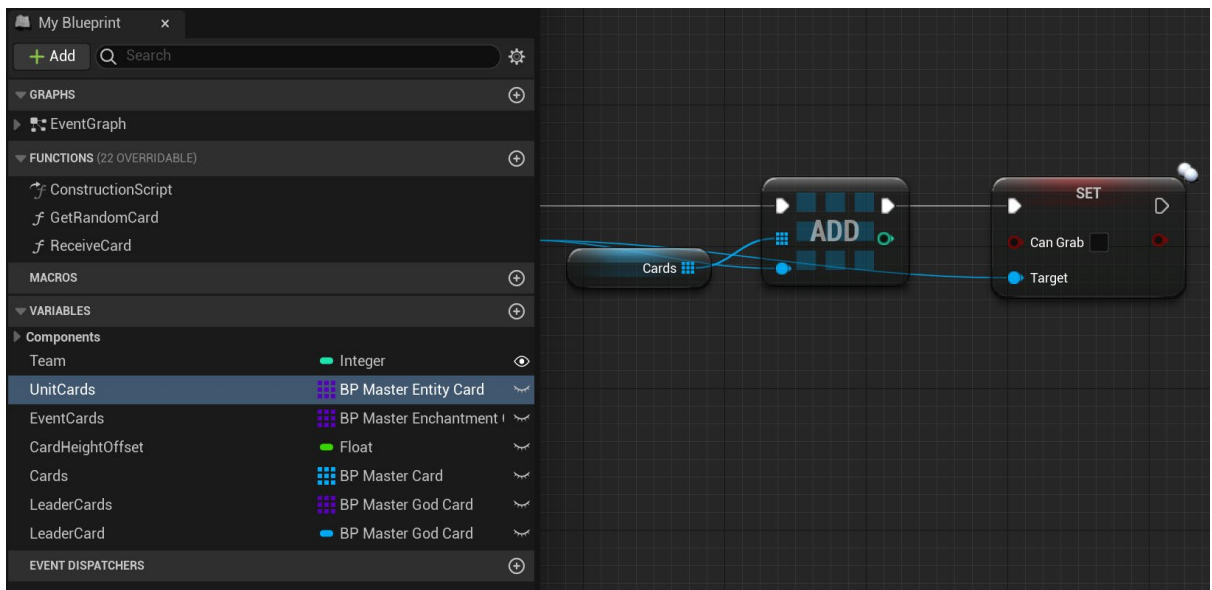
This blueprint contains the logic for all phases, win conditions, card revealing, coin flip etc.

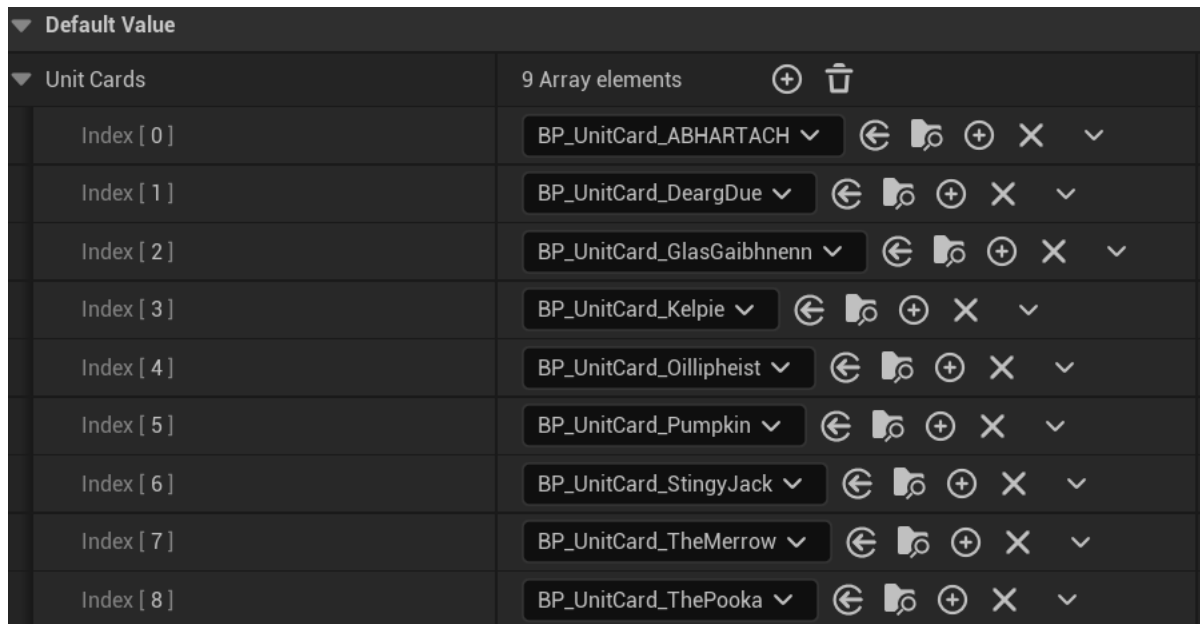
Card Deck Blueprint

Each player has its own deck blueprint located in the world, as shown below.



If I open this blueprint, I can change the starting cards and god cards a player can get, as shown below

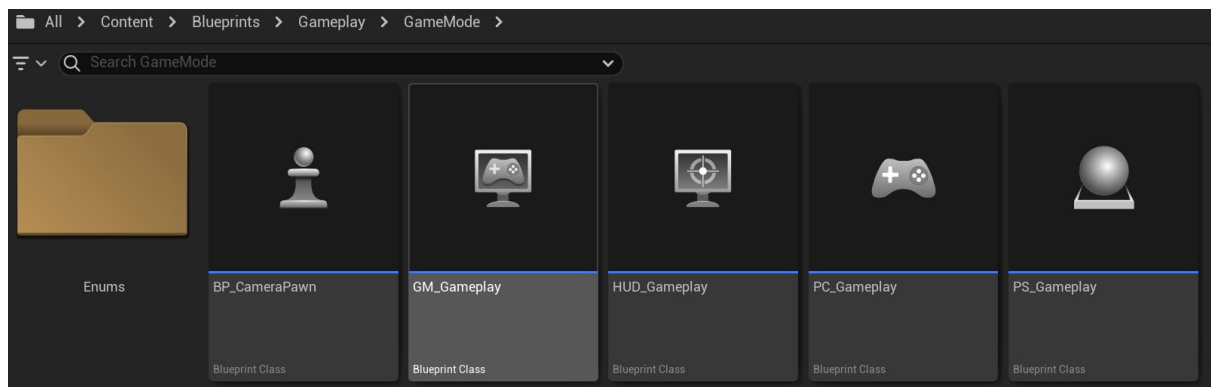




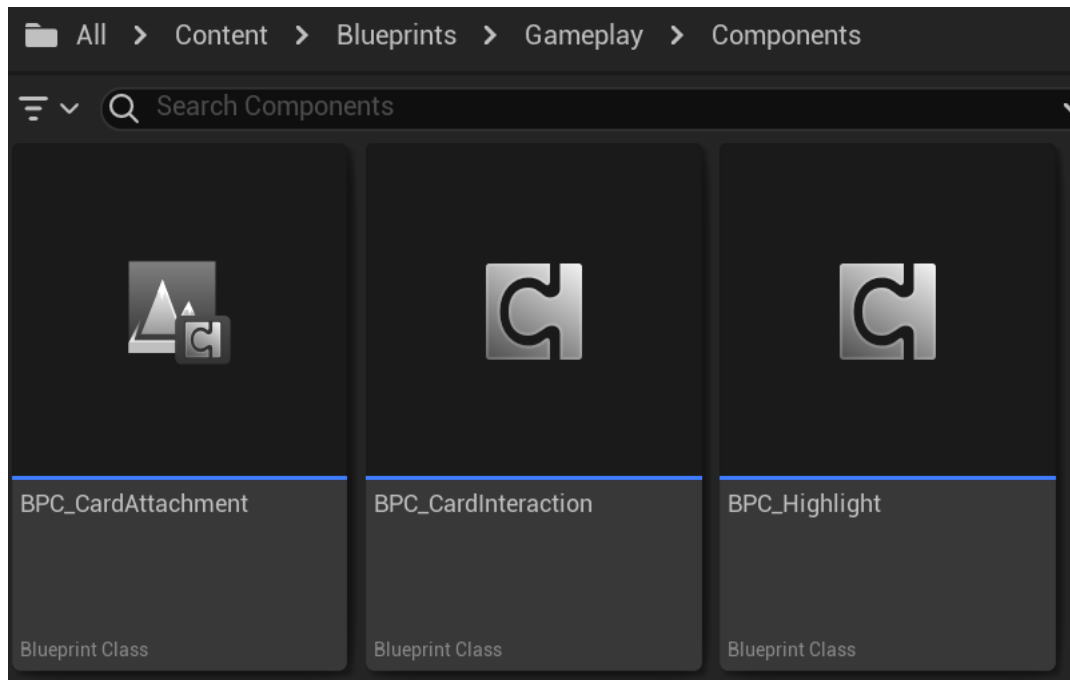
For example, if I made a new entity card, I would have to click the UnitCards variable (**NOTE:** Entity Cards were called Unit Cards early on in the development cycle) and I would add the new entity card class in the list above.

Player Controller and Card Interaction Component

One of the main blueprints is the PC_Gameplay, which can be found in the path shown below



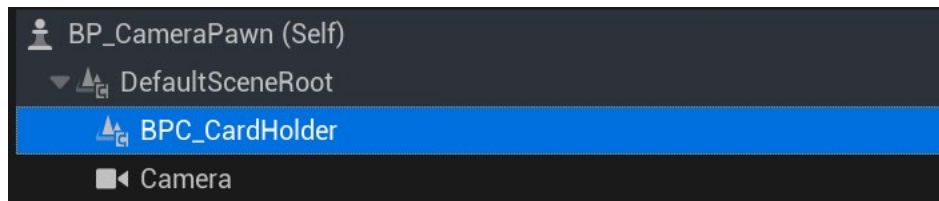
This blueprint contains some multiplayer router events inputs as well as the main component that handles all the card interaction, called BPC_CardInteraction. This component can also be found in the path shown below.



This component contains all the main card interaction logic for both client and other users, to whom the local user's card interactions are efficiently and smoothly replicated.

Card Holder Component (Player's Hand)

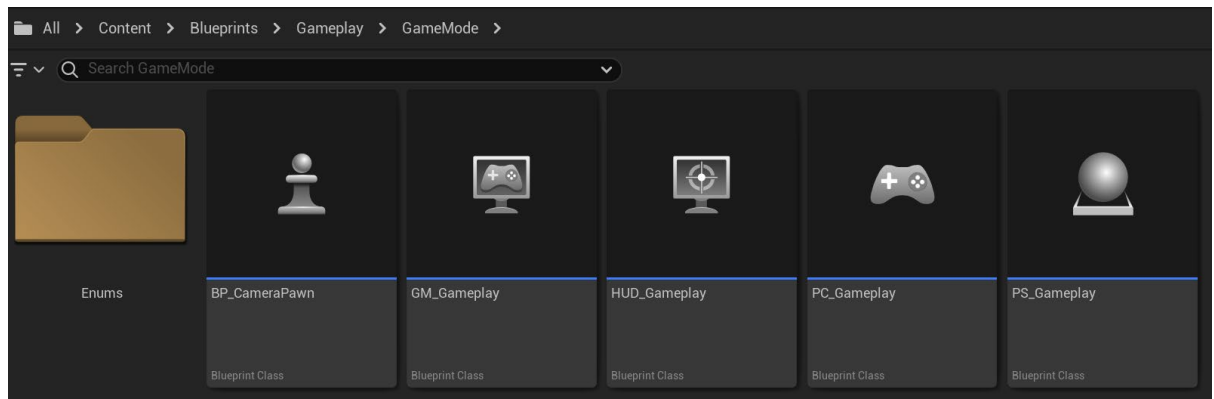
The card holder component is a scene component assigned to the camera pawn blueprint that handles the player's hand. This can be accessed by clicking the BPC_CardHolder component inside the camera pawn blueprint, as shown below



I can adjust some values here for the cards that appear in the player's hand such as Card Move Speed, Card Spacing, Card Width, Card Scaling, etc.

Other Integral Blueprints

HUD_Gameplay, BP_CameraPawn & PS_Gameplay



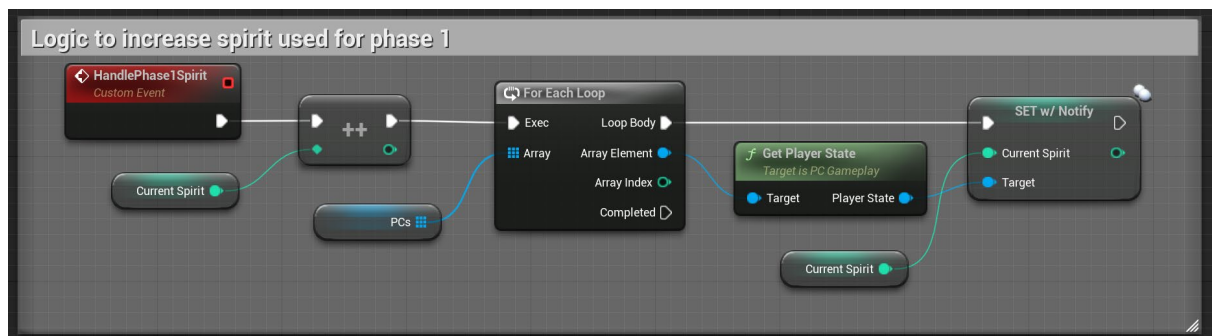
The HUD_Gameplay blueprint handles all of the UI logic of the game.

The BP_CameraPawn is the camera blueprint which all players use.

The PS_Gameplay blueprint is the player state blueprint of the game mode which is used to replicate values between players.

2.3. Implementation

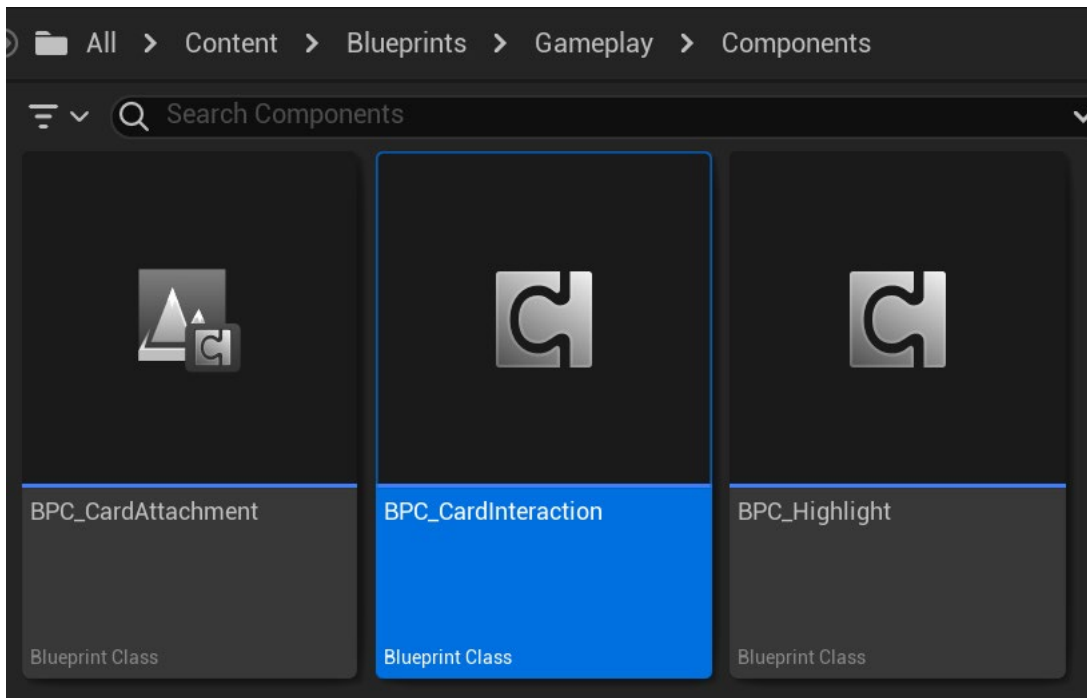
An example of implementation that I would like highlight is the simple custom event I created for increasing and setting the Spirit Counter (resource that allows a user to play an Entity card). I think this will clearly showcase the Blueprint Scripting system (C++ is behind this) within UE5 and help anyone who hasn't used this system to understand how it works.



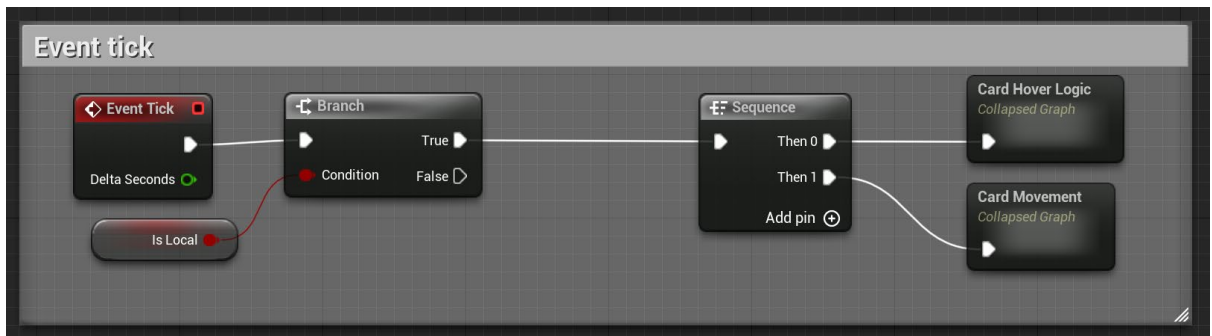
You can see the Custom Event called “HandlePhase1Spirit” in the top right. You can customize the parameters of these custom events any way you like, and they can be also be used within other custom events. For this example, we can see that we grab the “Current Spirit” Integer variable and connect it to the “++” node. This node adds 1 to the specified value and then sets it. We then connect that node to a “For Each Loop” node, which loops through the array of the connected “PCs” variable. This array essentially contains the identifiers for each user/client. This loop allows the stamina of both users to be increased by 1. The “PCs” array then connects to a “Get Player State Function” which tells the system the current state of the player which is an object within this function. The “Player State” Object and the “Current Spirit” variable are then set and sent to the server to be updated when this event is called.

I also want to briefly breakdown a challenging piece of a functionality that I implemented within the game. The multiplayer card interaction system, which enables players to effortlessly grab and move cards while also displaying interactions to the other player/client in a fluid transition and movement, was one of the most challenging design systems created for this project.

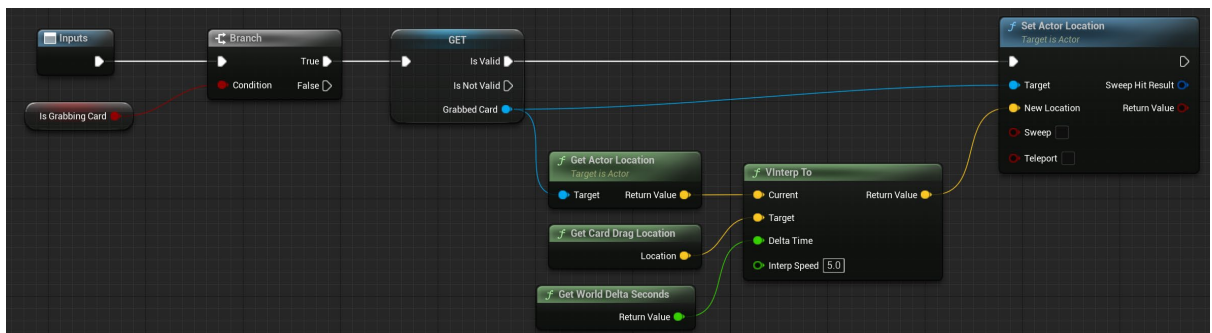
The main card interaction logic is inside the BPC_CardInteraction blueprint component.



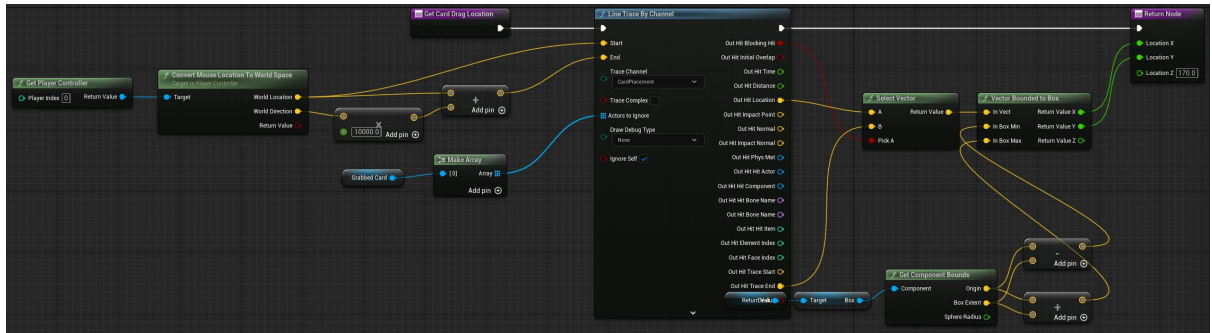
The logic I will break down is how the card movement logic is handled on both, local client and remote clients in multiplayer.



First, event tick will call the “Card Movement” composite node on Event Tick, only for the local client by checking the Is Local variable.

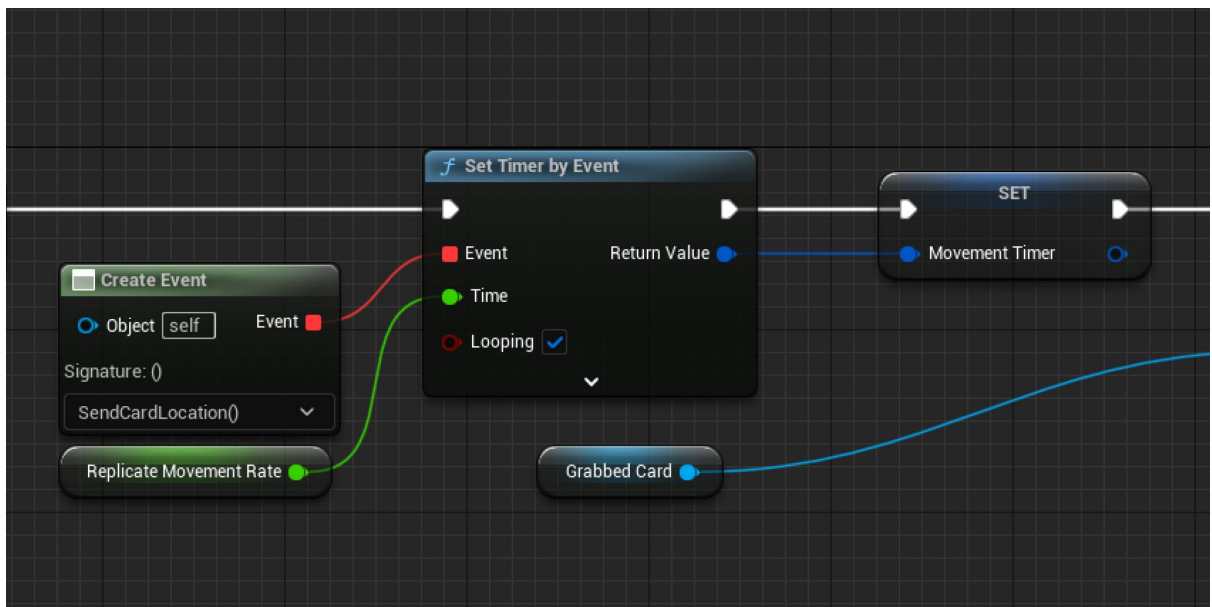


Within the “Card Movement” composite node, if the “IsGrabbingCard” variable is true, the local client will smoothly interpolate the position of the card towards the card drag location.

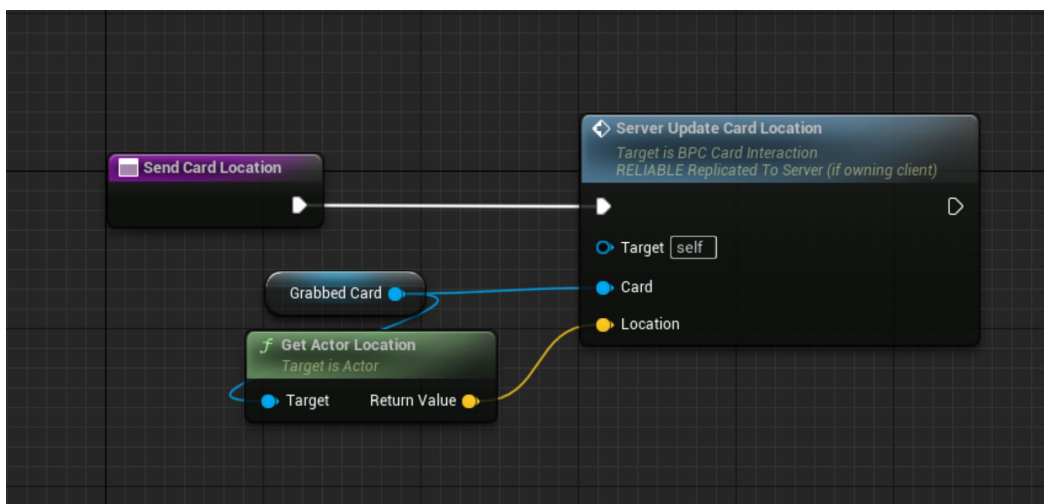


The card drag location, is basically a trace that defines where the card should go based on mouse position. This is how the movement works for the owning client, now let's go over how it works for other clients.

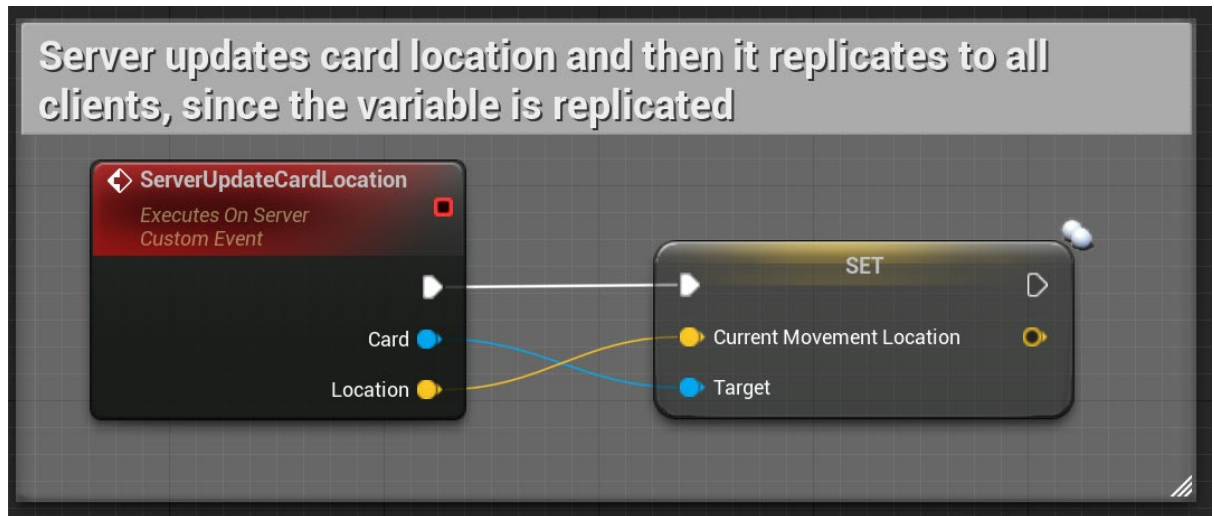
Firstly, when the card interaction happens on the client, this timer is started, which basically defines the rate of the update for the card location between client to server, which will then replicate back to the clients.



That function, calls the nodes below:



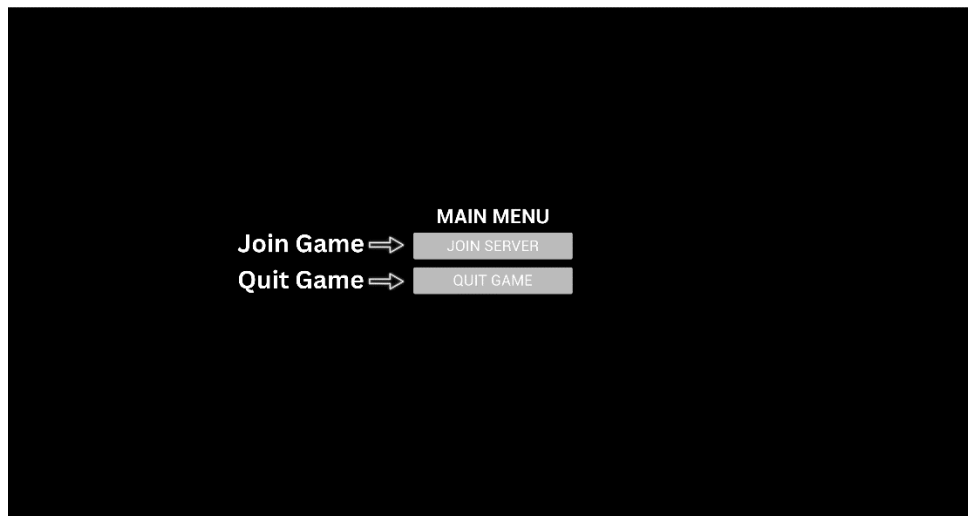
This basically sends a server RPC (remote procedure call) which will pass in the current grabbed card location from the client who controls it.



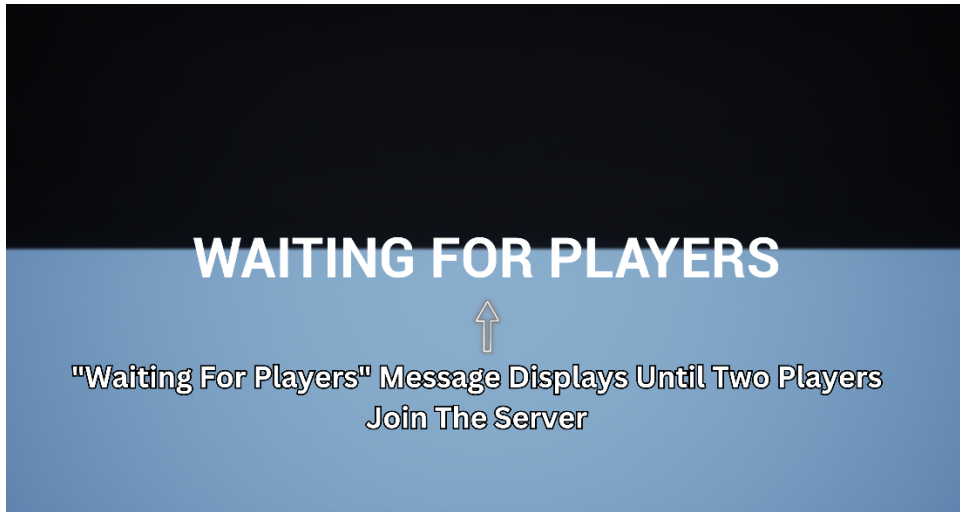
The server will then set the variable, which replicates the target location of the card back to all clients.

2.4. Graphical User Interface (GUI)

Main Menu GUI



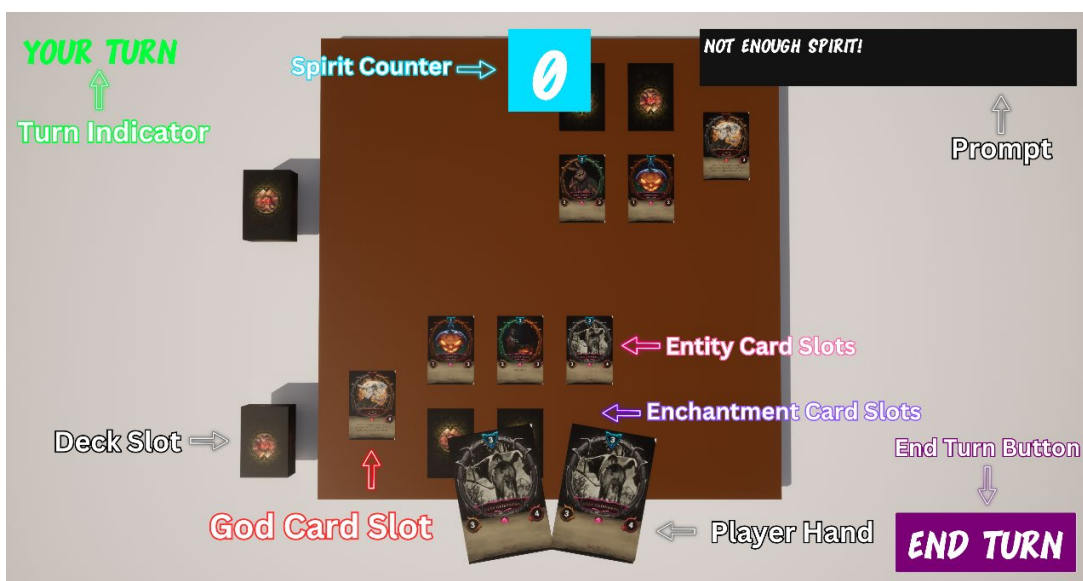
Waiting For Players Screen



Coin Flip Screen



Phase 1 GUI – Placing Cards Across Various Turns



Turn Indicator: Indicator which lets the player know whether it is their turn or the opponent's turn. (Top Left in Green)

Spirit Counter: Displays the amount of Spirit resource the player has available to spend on playing Entity cards. (Top Middle in Blue)

Prompt: Lets the player know that a certain action cannot be performed and the reason why. (Top Right in White)

Deck Slot: Slot which contains all the cards in a player's deck. (Left in White)

God Card Slot: Placement Area for God Card. (Left in Red)

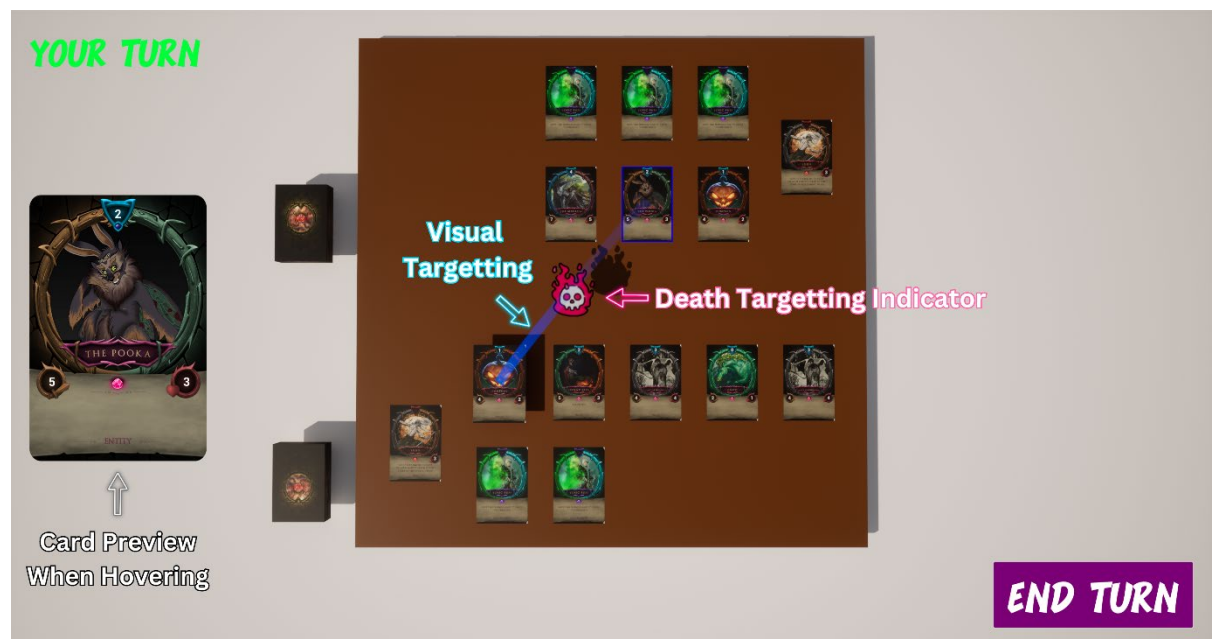
Players Hand: Displays the player's cards that they have drawn from their deck. (Bottom Middle in White)

Entity Card Slots: Slots where entity cards can be placed (Middle of Screen in Pink)

Enchantment Card Slots: Slots where enchantment cards can be placed (Middle of Screen in Purple)

End Turn Button: Once pressed, the player passes their turn over to the opponent. (Bottom Right in Purple)

Phase 2 GUI – Combat Phase

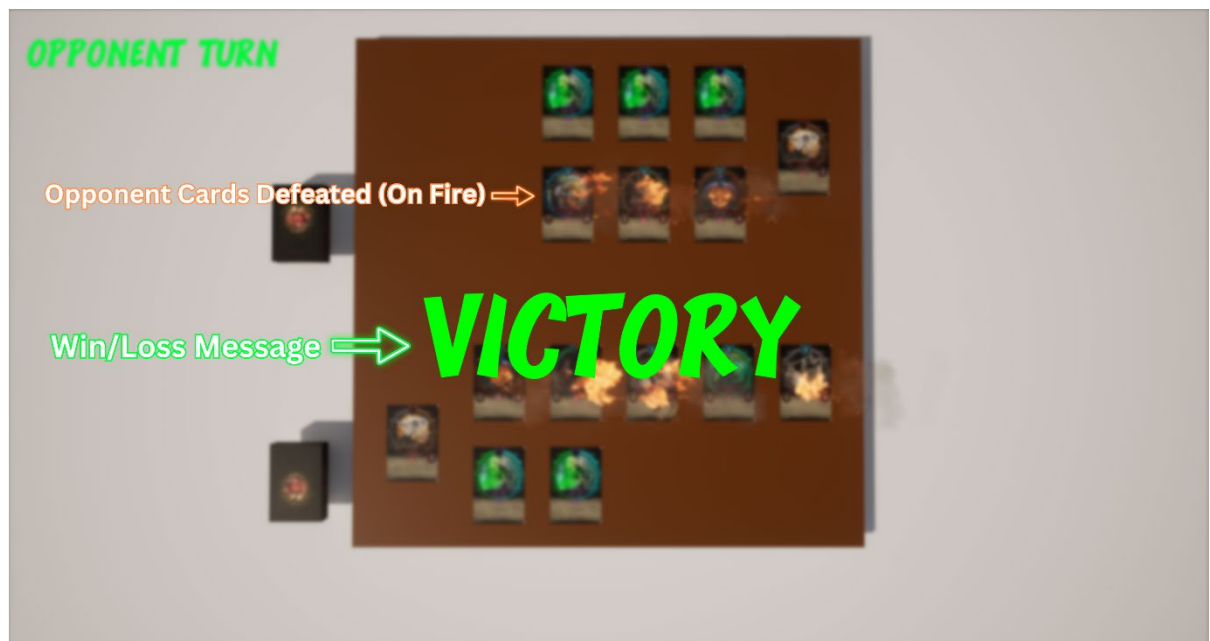


Card Preview When Hovering: Displays a large preview of the card that the mouse is hovering over for better readability. (Left in White)

Visual Targetting: Blue line which visually shows which card is being targeted (Middle of Screen in Blue).

Death Targetting Indicator: Pink Skull Symbol which indicates that a card will be defeated if chosen as the target for the attack (Middle of Screen in Pink).

Winner/Loser Screen



Card Types – Entity, Enchantment, God (Left to Right)



2.5. Testing

Throughout my project, I used the live testing feature within UE5 which allows the user to see the flow of the blueprint as you are playing through the game. This helped me to identify different issues that I found whilst playing through the game, especially when trying to test the different card abilities and card interactions.

My approach to testing was to build the blueprint, test the functionality within the game engine with the blueprint editor open elsewhere, follow the flow of the blueprint as the function played out within the game, and track down any issues that I may have encountered while playing back through the flow of the blueprint. This method worked extremely well for me as it meant that I

could visually track down where I thought the issue was and correct it. I used this through the majority of the development process for the game.

2.6. Evaluation

As I was testing card interactions and abilities, I quickly realised a very important aspect of game design within card games; “balance”. It was through conducting my testing and ensuring that the statistics of the cards were adding up correctly, etc., that I quickly discovered that my game was not very well balanced. Some card abilities were too complex and unnecessary, and it would be better that I spend my time trying to determine the foundational statistics for cards and then leave it in a state where I can easily return to the game and add more cards easily once this balance has been achieved. I’m happy to say that I have left the game in a good state where I can add new cards and abilities and test them accordingly if I want to develop this game further into the future. Please see the design & architecture section for how cards can be added into the game easily.

3.0 Conclusions

Originally, I wanted to include a lot more card abilities within my game, especially abilities attached to specific entity cards. However, during testing some of these abilities, I quickly realized that it was more important to get the basic card interactions functioning correctly first before over complicating the game with abilities that can cause issues further down the line.

But overall, I’m extremely happy with what I’ve been able to develop over the course of the last 9 months. I felt like I really challenged myself to learn a new technology and, in the end, I successfully built a functioning game within my chosen game engine. I learnt a lot about game design throughout the development cycle of this project, and I am sure I will use many of these skills I’ve picked up along the way far into my future.

4.0 Further Development or Research

I’m confident I’ve left this game in a state where I can revisit it in the future and add some of the features that I was unable to include during this initial development cycle such as additional card abilities. Moving forward, my focus would be to add and test additional card abilities while keeping in mind balance design, which was not my main focus while creating the game’s foundation cards. “Card balance” is one of the most difficult aspects to get right because an uneven game would lead to a decline in player base. Fortunately, the method I used to add abilities to cards can be refined to make sure that card statistics and abilities match up accordingly.

Of course, once I’m satisfied with the new functionality I want to add, I’d love to improve the game’s overall aesthetics. This would entail creating better UI components, enhancing card abilities with more visual effects, adding sound effects to particular cards, etc. Once the game’s main components are working properly, these features can always be added.

5.0 References

<https://graphicriver.net/item/trading-card-game-creator-vol-17/31777387>

6.0 Project Proposal



National College of Ireland

Project Proposal

Digital Card Game - Unreal Engine 5

24/10/2022

BSc (Honours) in Computing Software Development

2022/2023

Cree Gunning

X19302733

X19302733@student.ncirl.ie

Contents

1.0	Objectives.....	39
2.0	Background	40
3.0	State of the Art.....	40
4.0	Technical Approach.....	41
5.0	Technical Details	41
6.0	Special Resources Required	42
7.0	Project Plan	42
8.0	Testing.....	44

1.0 Objectives

This project sets out to achieve a fully functioning, playable build of a multiplayer digital trading card game built within Unreal Engine 5 where two players face off against each other in an epic duel. I have numerous objectives that I've set out for myself to accomplish over the course of the game's development cycle.

My initial objective will be to spend the majority of the first month or two researching and learning how to effectively use Unreal Engine 5 to create a digital card game.

From there, I will plan out the development cycle and set out some goals I'd like to achieve by certain stages of the project. First will be basic level design, then I'll tackle the main objective which will be implementing the rules and mechanics for turns, card interaction, winning a game, multiplayer, etc., using Unreal Engines blueprint system. I'll test the game over the course of development, however, when I'm happy with the core gameplay, I would like to allocate a specific time for in depth testing in order to iron out any bugs or issues that may affect the playability and flow of the game.

I have a lot of creative and well-developed concepts I'd love to include during this project, however, I believe that it is crucial to keep my focus on the development of the core mechanics of the game throughout the entirety of the project, as this is what will ultimately determine what the final outcome will be in terms of functionality, playability and structure.

If I'm confident I've completed my main objectives before the project is due, I will then see if I have the time to develop the game further and implement my additional concepts.

2.0 Background

I chose to undertake this project as I have an immense passion for both video games and trading card games. I've had this idea for a unique type of card game in my mind for a while now and so I think that this is the perfect opportunity to jump into the deep end and challenge myself to produce a real, playable version of it via Unreal Engine 5. I believe that working on something I'm genuinely interested in will help motivate me to put my all into this project.

Regarding the objectives I mentioned above in Section 1.0, I will learn and enhance my knowledge of how to develop a game within Unreal Engine 5 through the means of different video courses available to me. I will primarily use Udemy, as I've purchased courses from this site previously for other technology/software and I've found them to be extremely useful in gaining as much understanding as possible.

I will use what I learned from the video courses and tutorials to begin the level design of the game, and like I mentioned above, I'll dive deep into making my own custom blueprints (blueprints are essentially the code in UE5) which will contain all the core functionality and mechanics behind my game. This will be where most of my time will be spent during this project. UE5 is also designed to handle a lot of testing, as it has a useful feature that allows you to view the flow/process of your blueprints live as you are playing through the game, which allows you easily identify any issues that may be present within the blueprint.

3.0 State of the Art

There are many other digital card games that currently exist on the market such as Hearthstone (Blizzard), Legends of Runeterra (Riot Games), Magic the Gathering (Magic Digital Studio), etc. However, I'm confident that my own unique twist of the traditional card game format will allow my project to stand out. Many players get extremely overwhelmed when approaching a new card game for the first time, so my aim is to produce an easy to pick up and well formatted game for everyone to enjoy.

The core differences in my game are the board state/layout, the uniquely structured card slot system, the combat phase taking place after players have played all their turns, etc. To briefly expand and explain on this, in the other card games I've mentioned above, there is nearly always some sort of action that must be taken by the player to interact with your opponents cards each turn. However, in my game, you must plan your turns strategically, as you won't be interacting with the opponent's cards until later in the game after all turns have been played out. I will explain a lot about the rules and go more in depth about how the games plays out and its unique aspects within Section 8.0 Project Plan of this proposal. I

have so many creative and innovative approaches I'd love to include within this final project on top of these differences, so I'll aim to add a few more of these in if time permits. That being said, I'm confident that the core functionality and differences of this game will be apparent, nevertheless.

4.0 Technical Approach

My approach to the development of this project will be to clearly layout and plan what are the essential aspects of the game that allow core functionality. Luckily, I have a lot of the game logic and rules planned out already, so it will just be a case of transposing this information into a technical, development roadmap. It's extremely easy to say how a game should work, but the implementation is quite different. I will ensure that I identify the core aspects of the game that are needed in order for it to function at the most basic level (turn logic, card placement, card interaction, win condition, etc.), and focus my attention on these requirements. My aim will be to pick a specific task, assign myself a timeframe to complete it, break it down into as many parts as possible so that I can fully grasp what needs to be done in order to complete that task, and to repeat this process as I continue to work through the game's development. By doing this, I'll have a game that's quite barebones, however it will be at a stage where I know it functions as intended. From there, I can focus on adding to the game's mechanics and introducing new concepts such as giving existing cards abilities, etc. If I follow this technical approach, I'm sure that I can identify all the of the essential requirements that will need to be tackled throughout development of the game.

5.0 Technical Details

The implementation language that I will use for this project will be the blueprint system which is built into Unreal Engine 5. This is a complete visual and gameplay scripting system based on the concept of using a node-based interface to create gameplay elements from within the Unreal Editor. As with many common scripting languages, it is used to define object-oriented (OO) classes or objects in the engine. It is this blueprint system which will enable my game to function as intended. It will control everything from determining what current state of the game the player is currently in (what turn they're on, how much energy they have, what cards they can or can't play, etc.), the card logic interaction and getting them to properly affect gameplay (allowing the game to know when a card comes into play or when it gets destroyed), etc. This card game will be extremely heavy with multiple different blueprints controlling distinct mechanics for different scenarios and situations, and so it's essential to make sure that they are constructed properly to allow a seamless gameplay experience. This will be the key element behind programming this game. As I go through this project, I will identify what the essential blueprints will be in order for a playable version of the game to run. As stated before, this will be a lot of the logic for card interaction with other elements within the game such as the game board and other cards. These are the most critical technical details that I will focus on throughout the development of the project.

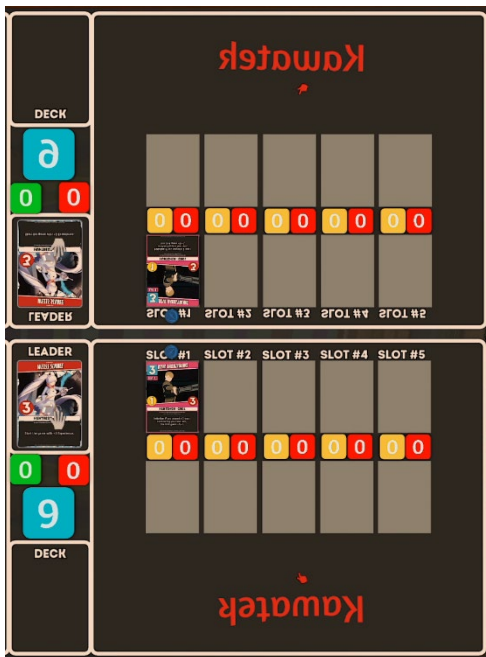
6.0 Special Resources Required

The special resources that will be required for this project will be the use of Unreal Engine 5 to develop the game of course, be that the aforementioned blueprint system and also the visual and testing elements that are included within the engine. Any art assets and visual effects/animations that may be needed for both the card design (main character art, boarder, etc.) and game board design may also be included as special resources.

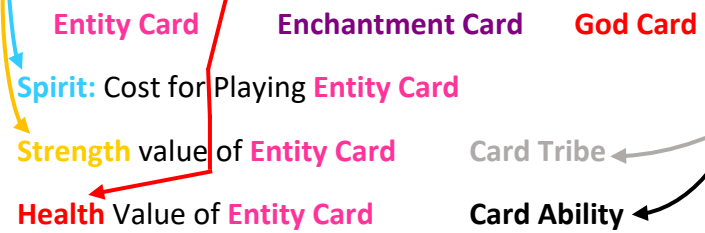
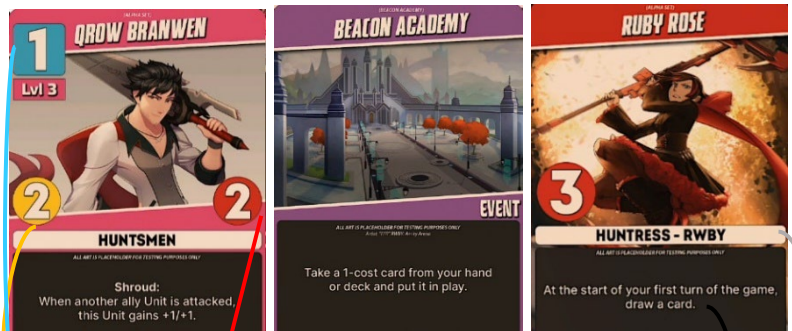
7.0 Project Plan

Brief Mock Up of Game and Rules

Mock Up Board State



Mock Up Cards



RULES

Flip a coin to see which player goes first.

There are five slots on the board, cards must be played from left to right.

5 spaces for **Entity Cards** at the top.

5 spaces for **Enchantment Cards** below.

Any number of **Entity Cards** can be played during a given turn.

Only one **Enchantment Card** can be played per turn.

5 turns for each player.

Spirit (Cost to play an **Entity Card**) increases by one each turn.

Entity Cards cost stamina to play.

Enchantment Cards cost nothing to play.

Entity Cards have **Strength** (Attack) & **Health** values.

Enchantment Cards have an ability that will affect the game in some way (E.g., Give an ally **Entity** +1 Strength).

Combat Phase

Combat phase begins after each player has taken their five turns.

The player who goes first is determined firstly by whoever has the most **Entities** on the board. If players have an equal number of **Entities**, the player who goes first is determined by whoever's **Entity** in slot 1 has the highest strength value.

Enchantment Cards are flipped/revealed first, from left to right (player who begins the combat phase cards will reveal first). Then the opponents **Enchantment Cards** will be revealed.

Entities can then attack opponent's **Entities** from left to right (when a card attacks a card, **strength** of attacking card is subtracted from **health** of card being attacked and vice versa).

When a **Entity** is defeated/destroyed it remains in the slot it was played, just becomes out of play (indication it's been defeated).

Win Condition

First player to defeat all opponent's **Entities** win.

If both player's **Entities** are all defeated, it's a tie.

Entity Abilities (Implement if time permits)

Lead: Ability/Effect triggers when a card comes into play.

Demise: Ability/Effect triggers when a card is defeated/destroyed.

Thrust: Ability/Effect triggers when the **Entity** initiates an attack.

Rough Timeline & Milestones

September – October/November: Research how to use Unreal Engine 5 and all of its core functionalities (Level Design, Coding/Blueprint System, Assets/Meshes (Objects within the game), etc.)

November – December: Assembly of barebone assets and level/game board design (Making an object/shape to represent cards, layout the board and slots correctly, allow cards to be picked up by the player and placed on board, etc.)

January – February: Core game mechanics and basic logic introduced using blueprints (Implement turns for each player, allowing cards to be placed on the appropriate slots, hand of cards for each player, drawing cards from deck, card statistics, etc.)

February – March: Continue to improve upon core functionality of the game in order to achieve a playable version of an entire game match. (Combat phase, order which cards come into play, order which cards come into play, win condition implemented, functioning multiplayer build, etc.)

March – April: Aim to have the core game complete around this stage of development and add in the final concepts (**Enchantment Cards**, give **Entity Cards** some basic abilities, etc.)

April – May: If time permits, add additional concepts into the game (described below)

Additional Win Condition: Introduce **God Cards** into the game. They have 3 health and grant the player a special bonus effect at the start on the game. The player can now win by either defeating all opponents' **Entities** or defeating the opponent's **God**.

Allocated time for final, in-depth testing before project deadline.

8.0 Testing

Again, as I mentioned previously, I will be testing the game vigorously throughout the development cycle in order to ensure that each blueprint is functioning correct and that the game plays out as intended. These tests will include everything from making sure that both players get 5 turns each to play cards, that cards can be placed in the appropriate slots (given the current turn/game state), that cards interact correctly with each other (strength of attacking card successfully reduces the health of the card being targeted), that the game can indeed be won by a player, and so forth. Once core functionality has been fully tested and I am satisfied that the game's mechanics play out correctly (an entire game can be played and won by a player successfully), I will evaluate any remaining time I may have left and decide whether to add more aspects to the game, but I'd like to ensure that these can also be appropriately tested and not just thrown in at the end.

7.0 Reflective Journals

Supervision & Reflection Template

Student Name	Cree Gunning
Student Number	X19302733
Course	BSH Computing (Year 4) (Software)
Supervisor	Emer Thornbury

Month: October

What?

Reflect on what has happened in your project this month?

Over the past month, I've researched many different ideas on what I could base my final project on. I landed on the idea of creating a digital card game within a game development engine. I've always had a great passion for card games, and I could clearly identify a gap in the card game genre where I believe my unique idea would slot in nicely and bring something fresh and new to the scene. I believe that my passion for this idea will really motivate me throughout the development of this project to produce a well-structured, complex, and fully functioning video game. I've also just submitted my project pitch and project proposal this month.

So What?

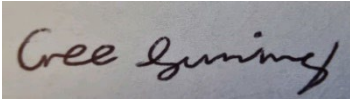
Consider what that meant for your project progress. What were your successes? What challenges still remain?

It's terrific that I chose such a great idea for my final project that I truly believe in, as it will really motivate me to work hard on the game's development over the next few months. I've had this idea for a card game in the back of my mind for a while, so a lot of the planning, rules and logic of the game have already been thoroughly thought out by myself. This actually takes some pressure off as I believe that I'm already well prepared and organized coming into this project, and so I can then just focus on implementing these layers and functionality into the game rather than having to come up with the entire concept and rules behind the game at this current stage. The challenges that still remain are how I will break down the development cycle of the project and manage my time effectively to achieve a fully functioning game by the time May comes around.

Now What?

What can you do to address outstanding challenges?

Now that I have decided on my project idea and I've submitted both my project pitch and my project proposal, I can begin to focus on learning the game engine. It's extremely important that I gain a good understanding of each individual aspect of this engine, so that I then have the ability to develop my project to a high standard of quality and complexity. Once I've learned the many different attributes of my chosen game engine, I can then effectively identify the essential elements that will make up the games' core functionality and focus the majority of my attention on continuing to develop these until the game plays as intended.

Student Signature	

Supervision & Reflection Template




Student Name	Cree Gunning
Student Number	X19302733
Course	BSH Computing (Year 4) (Software)
Supervisor	Emer Thornbury

Month: November

What?

Reflect on what has happened in your project this month?

Over the past month, I've continued to improve my understanding of Unreal Engine 5 and all its various aspects via video tutorials and documentation online. Alongside this, I began work on the actual project by making a rough game board view within the game engine software. I've also continued to plan out the development stages of my project by clearly identifying the functional requirements that are needed for the core game to function correctly. I've started to work on mock-up templates for the card face and card back that will be used within the game (see below). These will most likely be changed down the line but are handy to use for prototyping purposes.

So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

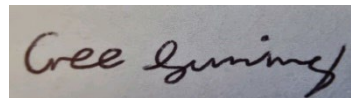
Ensuring that I fully understand how to use Unreal Engine 5 before jumping straight into my project has allowed me to better plan out the development stages and really try to optimize how I will work on each aspect of this project moving forward. Similarly, having the functional requirements laid out now gives me a clear view of what the most important tasks that should be tackled are, in order to reach that core functionality stage. Planning is an extremely important process when developing a video game, so I think it's super important to not gloss over this. The only challenges I faced this month were regarding jumping into the engine initially. It was still quite overwhelming even though I had watched plenty of informative tutorials and videos explaining how to navigate it, etc. This is true of every new type of software/program however, so I'm confident that I'll be grand with it as I continue to develop the project.

Now What?

What can you do to address outstanding challenges?

If I feel that I don't understand any particular aspect of the engine, I can always go back and refresh my memory by watching the tutorials again. There is so much excellent documentation for Unreal Engine 5 that I'm confident I'll be able to solve the majority of issues I might encounter along the way.

Student Signature



December - Supervision & Reflection Template

Student Name	Cree Gunning
Student Number	X19302733

Course	BSH Computing (Year 4) (Software)
Supervisor	Emer Thornbury

Month: December

What?

Reflect on what has happened in your project this month?

Over the past month, I completed my Mid-Point Presentation and Documentation. I highlighted the main functional requirements for my game within my documentation, explaining each use case in detail using diagrams and flow descriptions. My presentation highlighted my project plan and progress to date and requirements engineering, amongst other topics such as the tools and technologies I'm using within my project. I also including a demonstration of my game within this presentation which clearly displays the progress I have made over the past few months.

So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

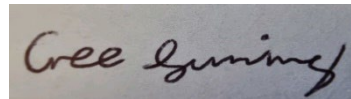
Filling out all of the documentation for the Mid-Point Presentation allowed me to clearly visualize all of the main functional requirements that are essential for the game to function as intended. It also allows me to get a head start on the documentation at the half way point of developing this project rather than at the end of the development cycle. The challenges that remain now are to ensure that I can indeed include all of these functional requirements that I have highlighted within my project successfully. Luckily, I have primarily worked on a lot of the main functionality of the game so far, so many of these functional requirements are covered already. But there are also many areas that need to be worked on such as the multiplayer aspects of the game, ensuring that two players can connect to a game together through a menu/UI system, etc.

Now What?

What can you do to address outstanding challenges?

If I am struggling to successfully include a key functional requirement within my project, I can take the time to research any issues I'm having and find a solution to the problem. It's important to take the time now to build upon these requirements and make sure I can incorporate all of them into the game sooner rather than later.

Student Signature



January - Supervision & Reflection Template

Student Name	Cree Gunning
Student Number	X19302733
Course	BSH Computing (Year 4) (Software)
Supervisor	Emer Thornbury

Month: January

What?

Reflect on what has happened in your project this month?

Over the past month, I have identified and planned out the most important aspects of my project that need to be worked on leading up to the final submission. These include a fully functioning menu/UI system, multiplayer integration, dynamic viewing of elements within the game, and additional card interaction and functionality. The visuals of my game are also an important aspect to keep in mind, however, I will focus on this in more detail closer to the end of my project development cycle when I'm also conducting my final testing. So the past month has essentially consisted of myself writing up an excel spreadsheet of cards and abilities, researching multiplayer integration within Unreal Engine 5, and continuing to work on a clean and appealing menu/UI.

So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

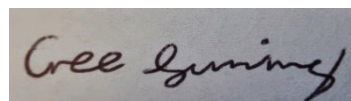
I have been able to work on categorizing abilities for my cards within my game and further develop this functionality. I have successfully tested some baseline abilities within a demo of my game so far, but these still need a lot of tweaking to work consistently and be different depending on specific card attributes. A big challenge that remains is integrating the multiplayer system as I want this to be a simple function that players can use to join into a game with ease. This is a little more complex than I first thought, so I think more research will be required to fully understand the best way to go about implementing this. Dynamic viewing of the main game board is being worked on at the moment and isn't too tricky thankfully.

Now What?

What can you do to address outstanding challenges?

Once again, I can address any of the challenges I have by referencing the video/tutorial material I've researched already, and I can take my time to ensure that I am integrating this functionality correctly and precisely. There are many people who have been in the same situation as myself when it comes to game development, so I'm sure that any problem I encounter can be tackled effectively and efficiently.

Student Signature



Supervision & Reflection Template

Student Name	Cree Gunning
Student Number	X19302733
Course	BSH Computing (Year 4) (Software)
Supervisor	Emer Thornbury

Month: February

What?

Reflect on what has happened in your project this month?

Over the past month, I have added additional functionality to my game such as dynamic viewing of cards so that they are presented in the correct orientation for each player. I have also added card preview functionality to my game so that when you hover over a card it will show you a zoomed in preview of the card for better readability. I'm continuing to improve some UI elements currently and I'm also focused on adding more cards and abilities to the game. Also continuing to work on multiplayer functionality.

So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

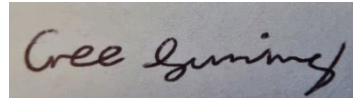
Adding the functionalities that are outlined above allows for a more user friendly experience which is incredibly important. It also helps new players to grasp the concept of the game more easily and learn the mechanics of the game as they continue to play. The multiplayer functionality is still quite challenging, however, this is my primary focus at the moment and I'm confident I will have it working correctly with time.

Now What?

What can you do to address outstanding challenges?

I am continuing to reference tutorials and guides on how to best setup the multiplayer system for my game using resources such as YouTube. Once again, it's important that I understand the concept behind how this works and how its integrated before just blindly trying to execute it.

Student Signature



Supervision & Reflection Template

Student Name	Cree Gunning
Student Number	X19302733
Course	BSH Computing (Year 4) (Software)
Supervisor	Emer Thornbury

Month: March

What?

Reflect on what has happened in your project this month?

This month my main focus was on developing the multiplayer system and ensuring that everything was setup correctly before "building" the project. I followed some video tutorials that I found on how best to execute this and I was able to get the server setup correctly. I made a basic GUI for the main menu of the game that allowed a player to join a game or exit the game. Once the player joins the game, they will be put into a waiting state/game lobby where a message is displayed to them reading "Waiting For Players". Once another instance of the game is running and the join game button is pressed within this instance, the game will then start and players will be able to battle each other.

I also added some more cards to the game such as new enchatment cards and new entity cards. Most of were just basic cards with minimal abilities.



So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

This was a huge leap forward in terms of project progress. The multiplayer system is such an integral part of the game since you cannot play this game as a single player. Getting this implemented correctly was definitely a challenge, but I'm glad that I took my time to follow some guides and execute the setup of it accordingly.

I haven't applied the abilities to some of the new enchantment cards that I added into the game so I still need to focus on setting these up correctly to interact with other cards.

Now What?

What can you do to address outstanding challenges?

I can use my existing enchantment cards as a basis for the abilities of the new cards and alter them accordingly so that the effect matches the text description of the ability on the card.

Student Signature

Student Name	Cree Gunning
Student Number	X19302733
Course	BSH Computing (Year 4) (Software)
Supervisor	Emer Thornbury

Month: April

What?

Reflect on what has happened in your project this month?

Leading into the last month and a half of the development cycle of this project, my main focus has been testing. I have been testing a few more card abilities and ensuring that they are working as intended. I have cut back on the original number of cards that I wanted to have in the game, as I wanted to ensure that I had a good base set of cards that were easy to implement and showcase as my project comes to a close. I decided that it is more important to leave the card design in a state that is easy to return to in the future than to over complicate the final project with multiple card abilities that ultimately end up confusing the user. I want it to be straightforward in terms of usability and understanding for the player. I also added some simple visual elements that help the player to better understand certain actions in the game, e.g., visual targeting indicator when an attacking card is targeting an enemy card.

So What?

Consider what that meant for your project progress. What were your successes? What challenges still remain?

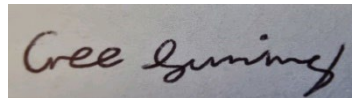
Deciding to remove some abilities from the game has been a tough but necessary decision I feel. Of course, I have this vision in my head of what this game could be and all the cool and unique abilities that I could include in it. However, I really had to ground myself and just look at the game from an outsider's perspective. I think that this is the right decision as it means that the game is less complicated for a first time user and once again, I have left it in a very good state of development so that when I return to develop it more in the future, I can easily add more abilities and cards to the game. The only few challenges that remain now are to continue to test the last few enchantment card abilities to ensure that the effects are being applied correctly to the entity cards and that the numbers are being calculated correctly.

Now What?

What can you do to address outstanding challenges?

I will keep using the live testing featuring within Unreal Engine 5 to ensure that I spot any issue or errors within my testing phase to close out the development of my project.

Student Signature

A photograph of a handwritten signature in black ink on a light-colored surface. The signature reads "Cree Luning" in a cursive script.

8.0 Other Materials Used