**SRS**

Software Requirements Specification (SRS)

Project Goose Gauntlet

Team: Group 3

Authors: Will Susi, Danny Machado, Lucas Torres, Christian Ambrosini, Wilson Van

Customer: Middle School Educators

Instructor: Dr. Daly

# Introduction

The sections of this Software Requirements Specification (SRS) will include all relevant and important information to the project. This section will provide a brief introduction to the project and lay out the organization of the following sections and subsections. This section will include the following subsections:

* 1.1 Purpose
* 1.2 Scope
* 1.3 Definitions, acronyms, and abbreviations
* 1.4 Organization

**1.1 Purpose**

This subsection will explain the project’s goals, intended purpose, target audience, and how the project will achieve these objectives.

**1.2 Scope**

This subsection will explain the scope of the project. It will detail the software products implemented in the project’s creation and how these products were applied, including the benefits, objectives and goals of each one listed. The intended purpose of the software product will also be documented here with its intended use and limitations.

**1.3 Definitions, acronyms, and abbreviations**

This subsection will define all of the terms, acronyms, and abbreviations needed to understand this SRS. All terms that do not have a clear meaning will be explained here.

**1.4 Organization**

This subsection will give an overview of the rest of the SRS after section 1. It will describe what the rest of the SRS contains as well as the structure of each section and its corresponding subsections. A brief description will be provided for each subsection with what it hopes to exhibit.

## Purpose

The purpose of this document is to provide a detailed and comprehensive description of the game for all parties involved. The document describes the methods and diagrams used throughout the design process and expresses functional and non-functional requirements of the game. The game was designed with the intended audience being middle school students.

## Scope

The sole product being produced is the computer game “Goose Gauntlet”. This software is intended to be an educational video game for middle school students. The game focuses on improving English word vocabulary in compliance with Massachusetts Department of Education and Common Core. The game develops this skill by forming the correlation between forming more complex words and performing better within the game.

## Definitions, acronyms, and abbreviations

* Enemy: A character that is hostile towards the player.
* Level: An area within the game where the player encounters 1-3 enemies.
* World: A grouping of levels of the same theme.
* Playthrough: A playthrough is the time frame from when the player first selects their difficulty to when the player sees the end screen.
* Card: An item that has a letter attribute and score attribute that the player uses to create words.
* Deck: A collection of cards that the player has collected throughout their playthrough.
* Turn: A time period where a character is able to do an action.
* Hand: A collection of cards which is available to the player on a given turn.
* Tray: A collection of cards that were selected from the player’s hand that is used to create a word to submit.
* Score: A number representation of how well the player did during their playthrough of the game. Score is calculated by getting the total amount of damage the player has done throughout their playthrough.

## Organization

The rest of the document will be organized as follows:

* 2 Overall Description
  + 2.1 Product Perspective
  + 2.2 Product Functions
  + 2.3 User Characteristics
  + 2.4 Constraints
  + 2.5 Assumptions and Dependencies
  + 2.6 Apportioning of Requirements
* 3 Specific Requirements
* 4 Modeling Requirements
  + 4.1 Use Cases
  + 4.2 Class Diagram
  + 4.3 Sequence Diagram
  + 4.4 State Diagram
* 5 Prototype
  + 5.1 How to Run Prototype
  + 5.2 Sample Scenarios
* 6 References
* 7 Point of Contact

# Overall Description

This section is broken into several different subsections discussing parts of Goose Gauntlet and how it will operate. In subsection 2.1 we will give context for the product and describe its communication with any interfaces. In subsection 2.2 we will summarize the major functionalities of the game in simple terms without going into any specific decisions. In subsection 2.3 we discuss what is expected from the user in order to be able to operate the game. In subsection 2.4 will discuss any constraints or restrictions of the game. In subsection 2.5 elaborates on any assumptions we have on the user/machine that will be using/running the game. Subsection 2.6 will talk about any future features we did not implement yet due to time constraints or complexity issues.

## Product Perspective

Goose Gauntlet is a stand-alone game that can be enjoyed by either students in the classroom or teenagers and adults at home. Goose Gauntlet mixes a fun and colorful fantasy theme with the complexities of a word making game. Anyone looking to learn or practice their English spelling and vocabulary could use this game.

Goose Gauntlet will be able to be played on any Windows 10 and above system as long as the user has it downloaded on their computer, or on a web browser. The software will have these constraints:

* System Interfaces:
  + The software will run completely on its own and is not part of some bigger system.
* User Interfaces:
  + The user will interact with the software entirely through the mouse.
* Hardware Interfaces:
  + The software will require a computer capable of outputting graphics, either through internal graphics or a graphics card.
  + The software will require a monitor to see the graphical output.
  + The software will require a mouse for user input.
* Software Interfaces:
  + The software does not require any other software besides at least a Windows 10 operating system.
  + If the user plays the game through on the web, it is required they use a web browser that supports HTML 5.
* Communication Interfaces:
  + Requires no communication with any other system.
* Memory Constraints:
  + The game will require less than a gigabyte on a hard drive.
  + If played through a web browser, it will only require a few megabytes for game save data.

## Product Functions

Our game will function as follows: the user will either download the game’s executable file from the game’s website or play the game in a browser from the website. Starting the game will load any previous saved data within the game’s file (highscores and a dictionary of words). The game will then load into the main menu screen where the user will be able to navigate other menus. These menus are listed below with their functionality:

* Play: The user is sent to a difficulty selection screen where they are prompted between an Easy, Medium and Hard difficulty which will then proceed to load the game at its first world at the selected difficulty.
* Leaderboard: A screen with previous scores will be shown to the user. The user will also be able to see the previous scores of a given difficulty.
* How to Play: A screen with the instructions on how the game is played will be shown to the user.
* Quit: Closes the game.

Once the user has chosen to Play, and picked a difficulty, the game will load the user into the first world and first level, its scenery, along with an enemy. The game then draws 8 cards the player has access to, and provides a tray to put the cards into. The goal of the player is to form a word by selecting cards in their hand, which places them in the tray. Damage will be done to the enemy based on the word formed. The longer the word, the more letters being used, and the more difficult the letters, the more damage the word will deal. If the player has dealt enough damage to defeat the enemy, the player beats that level and moves on. Otherwise, the enemy will then take its turn to deal damage to the player. If the damage the player receives reduces the health of the player to 0, the player loses. If neither the player or the enemy is defeated, the process repeats with a new hand for the player.

If the player beats the level they progress to the next level and the process is repeated with a new enemy. If the player beats the 5th level, the player progresses onto the next world. If the player beats the 5th level of the 3rd world the player beats the game.

If a player is on easy or medium difficulty, and is having trouble creating a word, then they can use a hint which will assist them in forming a word.

Whether the player loses or beats the game, the player is brought to an ending screen where their score will be displayed, and if it is high enough, will be added to the leaderboard.

## User Characteristics

The user of the game is expected to have at a least middle school level of reading and writing comprehension. The player does not need to have an extensive vocabulary to enjoy the game. The game encourages growth of vocabulary by guiding the user when they may feel stuck and challenges a user who wants something more difficult.

The user should also know the basics of navigating a computer. The mouse is the primary device that handles the user's choices/inputs. If the user can perform basic tasks such as browsing the web or opening programs, then they have enough skill with the computer to operate the software.

## Constraints

The game was designed to be played via computer/laptop. There is no support for mobile devices and there are currently no future plans to provide this support. The game also only supports the English language, and there are currently no future plans to provide support for any other language. If the user would like to play via browser, they also must have a stable internet connection. Otherwise the user can play the game offline via a downloadable executable.

## Assumptions and Dependencies

The user is assumed to have a computer/laptop with the understanding of how to operate it to run the game off of. It is also assumed the user has an understanding of the English language. If the user would like to play via browser, they are assumed to have a stable internet connection.

If the user has a desktop, the desktop should be able to display graphical output and the user is expected to have a monitor. The user should also have a mouse to be able to interact with the game itself.

In terms of software, if the user runs the game natively then their computer should at least have a version of Windows 10 or above. If the user runs the game on a browser, they should be using a web browser that supports HTML 5.

## Apportioning of Requirements

Everything in this document plans to be implemented. However, if unforeseeable obstacles arise, either due to complexity or time, the hint/difficulty system will be the last feature to be implemented. This is because it is the least important aspect of the workings of the system and can be omitted if need be.

These features, although initially discussed, will not be implemented either due to their complexity or due to time constraints:

* Endless Mode
* Custom Starting Deck
* Card Reward Choices for Beating Boss Fights
* Variable Number of Enemies

# Specific Requirements

1. When opening the game, the player will be shown the “Main Menu” screen with options.
   1. The player will have a “Play”option. Upon selecting “Play” the program will navigate to a difficulty selection screen.
   2. The player will have a “Leaderboard” option. Upon selecting “Leaderboard” the program will navigate to the Leaderboard screen.
   3. The player will have a “How To Play” option. Upon selecting “How To Play” the program will navigate to the How To Play screen.
   4. The player will have a “Quit” option. Upon selecting “Quit” the program will terminate.
2. The How To Play screen will display an info-graphic detailing all of the mechanics the player will need to know to play the game.
   1. The player will be able to return to the “Main Menu” from this screen.
3. The Leaderboard screen will display the highest scoring scores for each difficulty
   1. The Leaderboard screen will have sub-Leaderboards for each difficulty.
   2. The information displayed in each leaderboard will hold the top 3 playthroughs for that difficulty.
      1. A playthrough entry will contain the score the player achieved, the last World and Level the player successfully completes, and the date and time the player finished.
   3. If the player has fewer playthroughs in a respective difficulty than the number of shown highscores, only those playthroughs will be displayed.
   4. The player will be able to return to the “Main Menu” from this screen.
4. Upon entering the difficulty selection screen, the player will be prompted with three difficulties. When one of the difficulties is selected, the program will navigate to the 1st World.
   1. The difficulties the player can choose from will be “Easy”, “Medium”, and “Hard.”
   2. The player will be able to return to the “Main Menu” from this screen.
5. The playthrough will start once the program navigates to the 1st World.
   1. A playthrough is broken up into 3 distinct Worlds.
   2. Each World will have its own theme and aesthetic.
      1. World 1 will be a forest theme.
      2. World 2 will be a swamp theme.
      3. World 3 will be a castle theme.
   3. Each World is further broken down into 5 Levels.
      1. A Level contains an enemy.
      2. Once the player defeats the enemy in the Level, the program will navigate to the next Level.
         1. The Level navigation will be depicted by the player’s character walking to the right off the screen into black. The player will then appear in the next Level walking out from the left side of the screen.
      3. The fifth Level will contain a boss enemy that is distinctly stronger from the rest of the enemies.
      4. Once the player defeats the boss enemy in the fifth Level, the program will navigate to the next World.
   4. If the player loses at any point, or finishes the fifth Level in the Final World, they are brought to an Ending screen.
      1. The Ending screen will show the players total score, the difficulty of the playthrough, and the World they last successfully completed in a row.
      2. If the player's final score is higher than that of any of the scores on the respective difficulty’s Leaderboard, that score is added to the Leaderboard.
6. In each Level the player will be able to see their character and any enemy characters they encounter in the battle system in front of a designed scene for each World.
   1. The scene of every Level will consist of 2 dynamic midgrounds, a static background, and a foreground that follow the themes described in point 5.2.
   2. The player will be able to return to the “Main Menu” from this screen.
      1. If the player attempts to leave, the game will prompt the player if they would like to “Give Up” on the current playthrough.
         1. If the player selects “No” the playthrough will continue.
         2. If the player selects “Yes” the player will be sent to the Ending screen.
   3. Characters will be displayed in front of the scene.
      1. The player’s character will be displayed on the left side of the screen and enemy characters will be displayed on the right.
         1. Under any character a health bar will display their current health value.
7. In each Level the player will be provided with a deck, a hand, a tray, and a discard pile.
   1. The deck consists of a mix of cards that have letters on them.
      1. The mix of letters will have a balance of vowels and consonants.
   2. The deck is shuffled at the beginning of each Level.
   3. The hand will consist of 8 cards the player has access to on that turn.
      1. Once the current turn has been concluded, any cards the player played along with any cards remaining in the players hand will be sent to the discard pile.
   4. The discard pile will consist of the cards used and unused from all the previous turn’s hands.
      1. When there are fewer cards remaining in the deck than the player's hand size, the player draws the remaining cards in the deck, and then the discard pile is shuffled to create the new deck. Any cards needed to fill the player’s hand size are then drawn from the new deck.
   5. The tray is the area where selected cards will be displayed.
8. The player and the enemy will alternate turns until one is defeated.
   1. The program will engage the battle system when it navigates to each Level.
   2. A turn for the player consists of several steps.
      1. The player draws a hand of cards from the deck.
      2. The player then has to attempt to form a word with the given cards.
         1. The player can interact with the cards in their hand and any that are in the tray.
            1. Selecting a card in their hand will place it in the tray, appending it to any previous cards that remain there.

That same card will be grayed out in the hand and unselectable

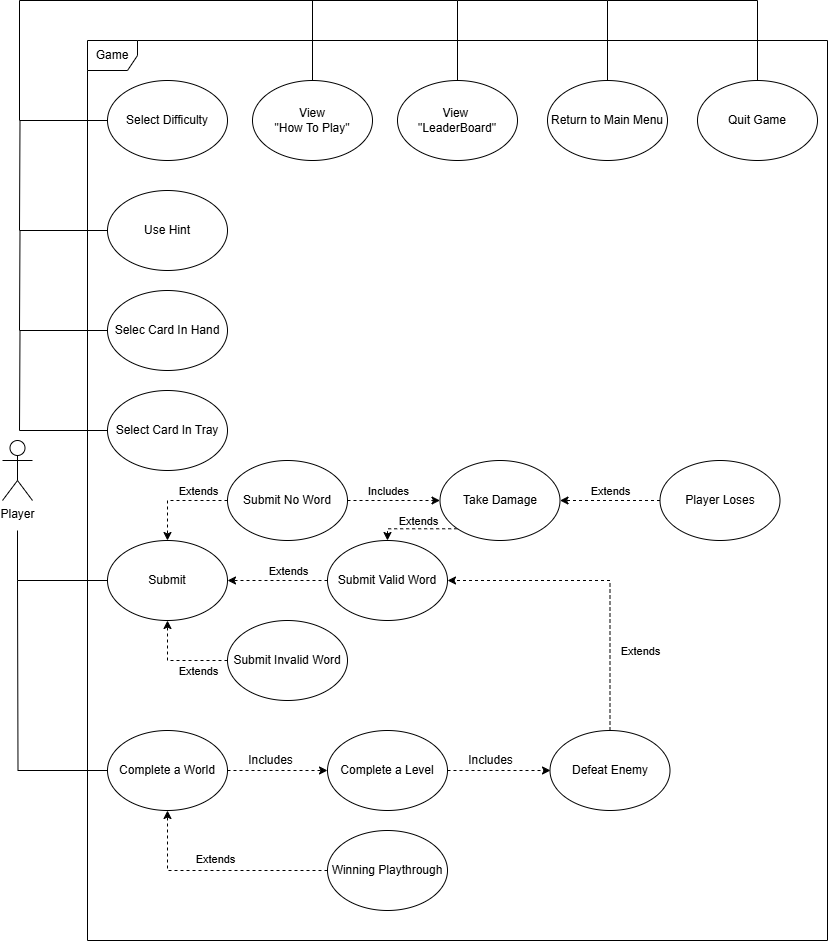
* + - * 1. Selecting a card in the tray will remove it from the tray and return the corresponding card in the hand to its normal state.
      1. The difficulty of the letters used to create the word will decide the total damage inflicted on the enemy for that turn.
    1. When the player wishes to end their turn they will submit to lock in their word.
       1. If the player did not enter any letters into their tray, they forfeit their turn, and no damage is dealt to the enemy.
       2. If their word is invalid, the game will tell the player the word is invalid, and allow them another attempt to form a word.
       3. If their word is valid, the appropriate damage is dealt to the enemy.
    2. The player's hand and any letters they used are sent to the discard pile.
  1. During the enemy’s turn, the enemy will deal an amount of damage in a given range dictated by the World and Level.
  2. The player or the enemy is defeated when their health reaches 0.
     1. When the player has a health of 0 the game ends and the player is sent to the Ending screen.
     2. When an enemy has a health of 0 they disappear from the battle.
     3. Once the enemy is defeated and the Level is cleared, the player's health is fully restored to its max value.
        1. Once a World is cleared the player also gets rewards.
           1. The player will gain a max health increase.
           2. The player will be able to choose one card to add to their deck from a selection of random cards.

1. There will be a Hint System while the player engages with the battle system.
   1. The player will be prompted to use a hint if they don't complete their turn within a certain amount of time.
   2. There will be a counter that displays the amount of hints the player has left to use.
      1. If the player has no remaining hints they will be unable to use a hint.
   3. If there are no possible words that can be spelled using the player’s current hand, attempting to use a hint will display a message in the center screen saying “No possible words.”
   4. If a word can be spelled using the player’s current hand, using a hint will either provide a new hint to the player or complete a prior hint.
      1. If a hint is not already active, using a hint will decrement the remaining hint counter and highlight the cards in the player’s hand which can be used to spell a particular word.
      2. If a hint is already active in the player’s hand, using a hint will complete the hint by arranging the cards into the order that spells the possible word in the tray. (This will not decrement the remaining hint counter.)
   5. The amount of hints available to the player and how the hints work are determined by the current game difficulty.
      1. The “Easy” difficulty will have these properties during gameplay.
         1. There will be 2 hints that the player can use during each World.
         2. The “Hint System” will have full functionality.
      2. The “Medium” difficulty will have these properties during gameplay.
         1. There will be 1 hint that the player can use during their current World.
         2. The “Hint System” will only have the functionality to provide a hint by highlighting letters, and will not complete hints.
      3. The “Hard” difficulty will have these properties during gameplay.
         1. The “Hint System” will be disabled.

# Modeling Requirements

## Use Case Diagram

The following use case diagram shows an overview of the actors, in our case the Player, and their interactions with the game.



| Use Case Name: | View “LeaderBoard” |
| --- | --- |
| Actors: | Player |
| Description: | The player navigates to the LeaderBoard screen to view highscores. |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |

| Use Case Name: | View “How To Play” |
| --- | --- |
| Actors: | Player |
| Description: | The player navigates to the How To Play screen to learn how to play the game. |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |

| Use Case Name: | Quit Game |
| --- | --- |
| Actors: | Player |
| Description: | The player navigates to Quit to exit the game. |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |

| Use Case Name: | Return to Main Menu |
| --- | --- |
| Actors: | Player |
| Description: | The player returns to the Main Menu. |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |

| Use Case Name: | Select Difficulty |
| --- | --- |
| Actors: | Player |
| Description: | The player chooses their desired difficulty for their Playthrough. |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |

| Use Case Name: | Use Hint |
| --- | --- |
| Actors: | Player |
| Description: | The player uses a Hint to receive assistance during the current turn. |
| Type: | Primary |
| Includes: | Find Words in Hand |
| Extends: | N/A |

| Use Case Name: | Select Card In Hand |
| --- | --- |
| Actors: | Player |
| Description: | The player selects a Card in the Hand and the Card is moved to the Tray. |
| Type: | Primary |
| Includes: | Put Card in Tray |
| Extends: | N/A |

| Use Case Name: | Select Card in Tray |
| --- | --- |
| Actors: | Player |
| Description: | The player selects a card in the Tray and the Card is moved to the Hand. |
| Type: | Primary |
| Includes: | Put Card in Tray |
| Extends: | N/A |

| Use Case Name: | Submit |
| --- | --- |
| Actors: | Player |
| Description: | The player submits the Cards currently in the Tray to complete their turn. |
| Type: | Primary |
| Includes: | N/A |
| Extends: | N/A |

| Use Case Name: | Submit Invalid Word |
| --- | --- |
| Actors: | Player |
| Description: | The player submits a word in the Tray, but it is not recognised and the Cards are sent back to the Hand. |
| Type: | Secondary |
| Includes: | N/A |
| Extends: | Submit |

| Use Case Name: | Submit No Word |
| --- | --- |
| Actors: | Player |
| Description: | The player submits an empty Tray because they cannot construct a word, and wishes to end the turn. |
| Type: | Secondary |
| Includes: | Take Damage |
| Extends: | Submit |

| Use Case Name: | Submit Valid Word |
| --- | --- |
| Actors: | Player |
| Description: | The player submits a word in the Tray, and the appropriate damage is dealt to the enemy. |
| Type: | Secondary |
| Includes: | N/A |
| Extends: | Submit |

| Use Case Name: | Complete A World |
| --- | --- |
| Actors: | Player |
| Description: | The Player finishes the last (5th) Level in the World. |
| Type: | Primary |
| Includes: | Complete a Level |
| Extends: | N/A |

| Use Case Name: | Complete a Level |
| --- | --- |
| Actors: | Player |
| Description: | The Player completes a level by defeating its enemy. |
| Type: | Secondary |
| Includes: | Defeat Enemy |
| Extends: | N/A |

| Use Case Name: | Defeat Enemy |
| --- | --- |
| Actors: | Player |
| Description: | The Player has dealt enough total damage to reduce the health value of an enemy to zero. |
| Type: | Tertiary |
| Includes: | N/A |
| Extends: | Submit Valid Word |

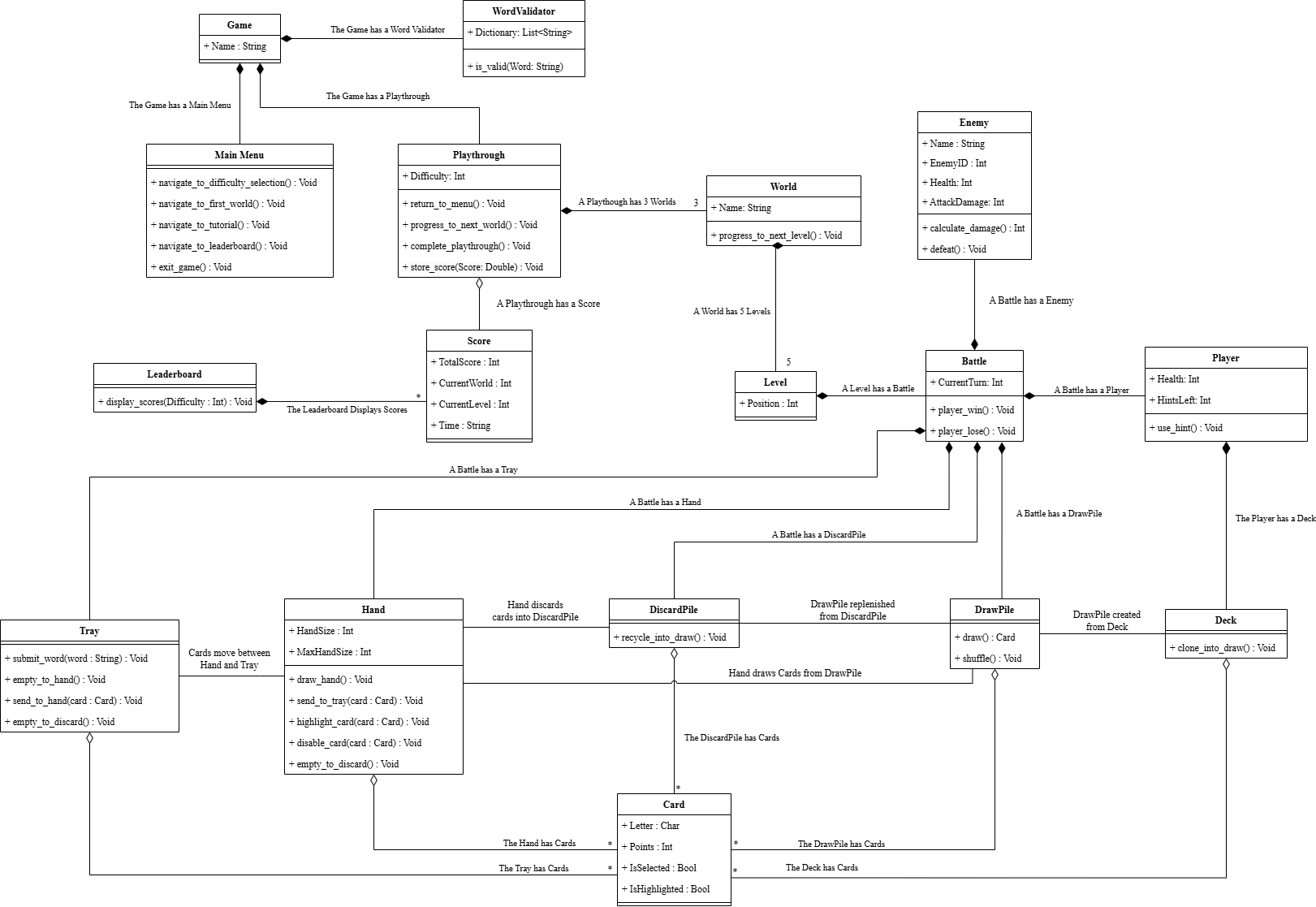
| Use Case Name: | Winning Playthrough |
| --- | --- |
| Actors: | Player |
| Description: | The Player beats all Worlds. |
| Type: | Secondary |
| Includes: | N/A |
| Extends: | Complete a World |

| Use Case Name: | Take Damage |
| --- | --- |
| Actors: | Player |
| Description: | The player takes damage from an enemy. |
| Type: | Tertiary |
| Includes: | Submit No Word |
| Extends: | Submit Valid Word |

| Use Case Name: | Player Loses |
| --- | --- |
| Actors: | Player |
| Description: | The player's health value is zero and the player loses. |
| Type: | Quaternary |
| Includes: | N/A |
| Extends: | Take Damage |

## Class Diagram

The following class diagram shows an overview of each class, its respective attributes and operations, and their relationships between each other in the game.



| **Element Name** | | **Description** |
| --- | --- | --- |
| Battle | | An encounter between the Player and an Enemy. |
| **Attributes** |  |  |
|  | CurrentTurn: Int | The current turn of the Battle. |
| **Operations** |  |  |
|  | player\_win() : Void | Determines if the Player has won the Battle. |
|  | player\_lose() : Void | Determines if the Player has lost the Battle. |
|  | damage\_enemy(word : String) : Void | Deals damage to the Enemy based on a given word. |
|  | damage\_player (damage : Int) : Void | Deals damage to Player. |
| **Relationships** | Is part of a Level.  Has a Player.  Has an Enemy.  Has a DrawPile.  Has a DiscardPile.  Has a Hand.  Has a Tray. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Card | | An item the Player uses to create words. |
| **Attributes** |  |  |
|  | Letter: Char | The letter that the Card represents. |
|  | Points: Int | The amount of points the Card is worth. |
|  | IsSelected: Bool | Whether or not the Player has selected the Card. |
|  | IsHighlighted: Bool | Whether or not the Card is highlighted. |
| **Operations** |  |  |
|  | N/A | N/A |
| **Relationships** | Is part of Deck.  Is part of DrawPile.  Is part of DiscardPile.  Is part of Hand.  Is part of Tray. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Deck | | A collection of Cards the Player has throughout the Playthrough. |
| **Attributes** |  |  |
|  | N/A | N/A |
| **Operations** |  |  |
|  | clone\_into\_draw() : Void | Copies the Cards in the Deck into the DrawPile. |
| **Relationships** | Has any number of Cards.  Is part of Player. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| DiscardPile | | A collection of Cards discarded from previous turns. |
| **Attributes** |  |  |
|  | N/A | N/A |
| **Operations** |  |  |
|  | recycle\_into\_draw() : Void | The DiscardPile is emptied and the Cards are placed in the DrawPile. |
| **Relationships** | Has any number of Cards.  Is part of a Battle. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| DrawPile | | A collection of cards that the Player will draw from. |
| **Attributes** |  |  |
|  | N/A | N/A |
| **Operations** |  |  |
|  | draw() : Void | A card from the DrawPile will be added to the Hand. |
|  | shuffle() : Void | The order of the Cards in the DrawPile are randomized. |
| **Relationships** | Has any number of Cards.  Is part of a Battle. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Enemy | | The opponent the Player will face during a Battle. |
| **Attributes** |  |  |
|  | Name : String | The name of the Enemy. |
|  | EnemyID : Int | A number to signify the type of enemy. |
|  | Health : Int | The amount of life points the Enemy has. |
|  | AttackDamage : Int | The damage the enemy inflicts. |
| **Operations** |  |  |
|  | calculate\_damage() : Int | Determines how much damage the Enemy will inflict on the Player. |
|  | defeat() : Void | Disables the Enemy. |
| **Relationships** | Is part of a Battle. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Game | | Overarching parent class. |
| **Attributes** |  |  |
|  | Name : String | Name of the Game. |
| **Operations** |  |  |
|  | N/A | N/A |
| **Relationships** | Has a Main Menu.  Has a Playthrough. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Hand | | A collection of Cards the Player has access to during their turn. |
| **Attributes** |  |  |
|  | HandSize : Int | The number of cards in the Hand. |
|  | MaxHandSize : Int | The maximum number of cards the Hand can hold. |
| **Operations** |  |  |
|  | draw\_hand() | Draws a set of Cards from the DrawPile to fill the Hand. |
|  | send\_to\_tray(card : Card) : Void | Sends a given Card to the Tray. |
|  | highlight\_card(card : Card) : Void | Highlights a given Card in the Hand. |
|  | disable\_card(card : Card) : Void | Disables a given Card from being selected in the Hand |
|  | empty\_to\_discard() : Void | Moves all the Cards in the Hand to the DiscardPile. |
| **Relationships** | Has 8 Cards. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Leaderboard | | Holds and displays the top Scores of previous Playthroughs. |
| **Attributes** |  |  |
|  | N/A | N/A |
| **Operations** |  |  |
|  | display\_scores (Difficulty : Int) : Void | Displays the top Scores of the given difficulty. |
| **Relationships** | Has Scores. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Level | | Represents a Level. |
| **Attributes** |  |  |
|  | Position : Int | The position in the World the Level occupies. |
| **Operations** |  |  |
|  | N/A | N/A |
| **Relationships** | Is part of a World.  Has a Battle. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Main Menu | | Introduction screen with options of navigation. |
| **Attributes** |  |  |
|  | N/A | N/A |
| **Operations** |  |  |
|  | navigate\_to\_difficulty\_selection() : Void | Sends the Player to the difficulty selection screen. |
|  | navigate\_to\_first\_world() : Void | Sends the Player to the first World at a given difficulty. |
|  | navigate\_to\_tutorial() : Void | Sends the Player to the How To Play screen. |
|  | navigate\_to\_leaderboard() : Void | Sends the Player to the Leaderboard screen. |
|  | exit\_game() : Void | Closes the game. |
| **Relationships** | Is part of the Game. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Player | | The user of the game. |
| **Attributes** |  |  |
|  | Health: Int | The amount of health the Player has. |
|  | HintsLeft: Int | The amount of Hints the Player has remaining. |
| **Operations** |  |  |
|  | use\_hint() : Void | Provides the Player with varying assistance based on the difficulty. |
| **Relationships** | Is part of a Battle.  Has a Deck. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Playthrough | | Represents a single Playthrough of the game. |
| **Attributes** |  |  |
|  | Difficulty : Int | The difficulty the Player selects. |
| **Operations** |  |  |
|  | return\_to\_menu() : Void | Completes the Playthrough and returns the Player to the Main Menu. |
|  | progress\_to\_next\_world() : Void | Progresses the Playthrough onto the next World or completes the Playthrough if it’s the last World. |
|  | complete\_playthrough() : Void | Completes the Playthrough for the player, showing them an end screen, and returning them to the Main Menu. |
|  | store\_score(Score : Double) | Stores the final Score. |
| **Relationships** | Is part of the Game.  Has 3 Levels.  Has a Score. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Score | | A representation of how much damage the player has dealt in a single playthrough. |
| **Attributes** |  |  |
|  | TotalScore: Int | The amount of damage a Player has dealt in a Playthrough. |
|  | CurrentWorld: Int | The current World the Player is playing in the Playthrough. |
|  | CurrentLevel: Int | The current Level the Player is playing in the Playthrough. |
|  | Time: String | The amount of time that has elapsed in the Playthrough. |
| **Operations** |  |  |
|  | N/A | N/A |
| **Relationships** | Is part of a Playthrough.  Is part of the Leaderboard. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| Tray | | A representation of the Cards the Player has selected from the Hand. |
| **Attributes** |  |  |
|  | N/A | N/A |
| **Operations** |  |  |
|  | get\_word() : String | Forms a word from the Cards in the Tray. |
|  | empty\_to\_hand() : Void | The Card(s) in the Tray are sent to the Hand. |
|  | send\_to\_hand(card : Card) : Void | A given Card is sent from the Tray to the Hand. |
|  | empty\_to\_discard() : Void | The Cards in the Tray are sent to the DiscardPile. |
| **Relationships** | Has 0..8 Cards.  Is part of a Battle. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| WordValidator | | Handles the list of valid words. |
| **Attributes** |  |  |
|  | Dictionary : List<String> | Holds a list of valid words. |
| **Operations** |  |  |
|  | is\_valid (word: String): Boolean | Checks if a given word is found in the valid words list. |
| **Relationships** | Game has a WordValidator. | |

| **Element Name** | | **Description** |
| --- | --- | --- |
| World | | Represents a world. |
| **Attributes** |  |  |
|  | Name : String | The name of the world. |
| **Operations** |  |  |
|  | progress\_to\_next\_level() | Progresses the Playthrough onto the next Level of the World or completes the Playthrough if it’s the last World. |
| **Relationships** | Has 5 Levels. | |

## Representative Scenarios of System

The following representative scenarios describe the major processes of the game.

* + 1. Gameplay

The Player has chosen a difficulty and the Playthrough is created/started. The Playthrough loops, starting on the first World and first Level of that World, and loops until the Player completes the last Level of the last World (Level 5, World 3) or is defeated. Once the Player enters a level, the Battle starts. The Battle creates a DrawPile, DiscardPile, Hand, and Tray. The DrawPile is populated by cloning the Player’s Deck and then shuffling it. The DiscardPile, Hand, and Tray are initialized empty. The Battle loops until either the Player or the Enemy is defeated. Whether the Player is defeated or completes the last Level of the Lost World, the Playthorugh completes, the Score is shown to the User, stored in the Leaderboard if high enough, and the User is returned to the Main menu.

* + 1. Turn (Hard Difficulty)

The Battle has started and its contents have already been initialized/created. There have been any number of turns before this. Battle draws Cards from the DrawPile to populate the Hand size. If the DrawPile is emptied before filling the Hand size, the DiscardPile is shuffled back into the DrawPile and the remaining Cards are drawn to fill the Hand size.

The control is turned over to the user and they are able to do any number of actions until they hit the “Submit” button. The user may select any Cards in the Hand to send to the Tray in an attempt to form a word, or select any Cards that have been moved to the Tray to return them to the Hand.

If the user submits no word, meaning the Tray is empty, no damage is dealt to the enemy and they forfeit their turn. If the user submits an invalid word the Cards are kicked back to the Hand and their turn resumes. If the user submits a valid word, the amount of damage is calculated based on the word, and dealt to the Enemy. If the Enemy’s health reaches 0, the Player wins that Battle and progresses accordingly.

If the enemy is not defeated, their turn commences. Battle calculates a damage value based on the enemy and deals it to the Player. If the Player’s health reaches 0, the Player loses that Battle and progresses accordingly.

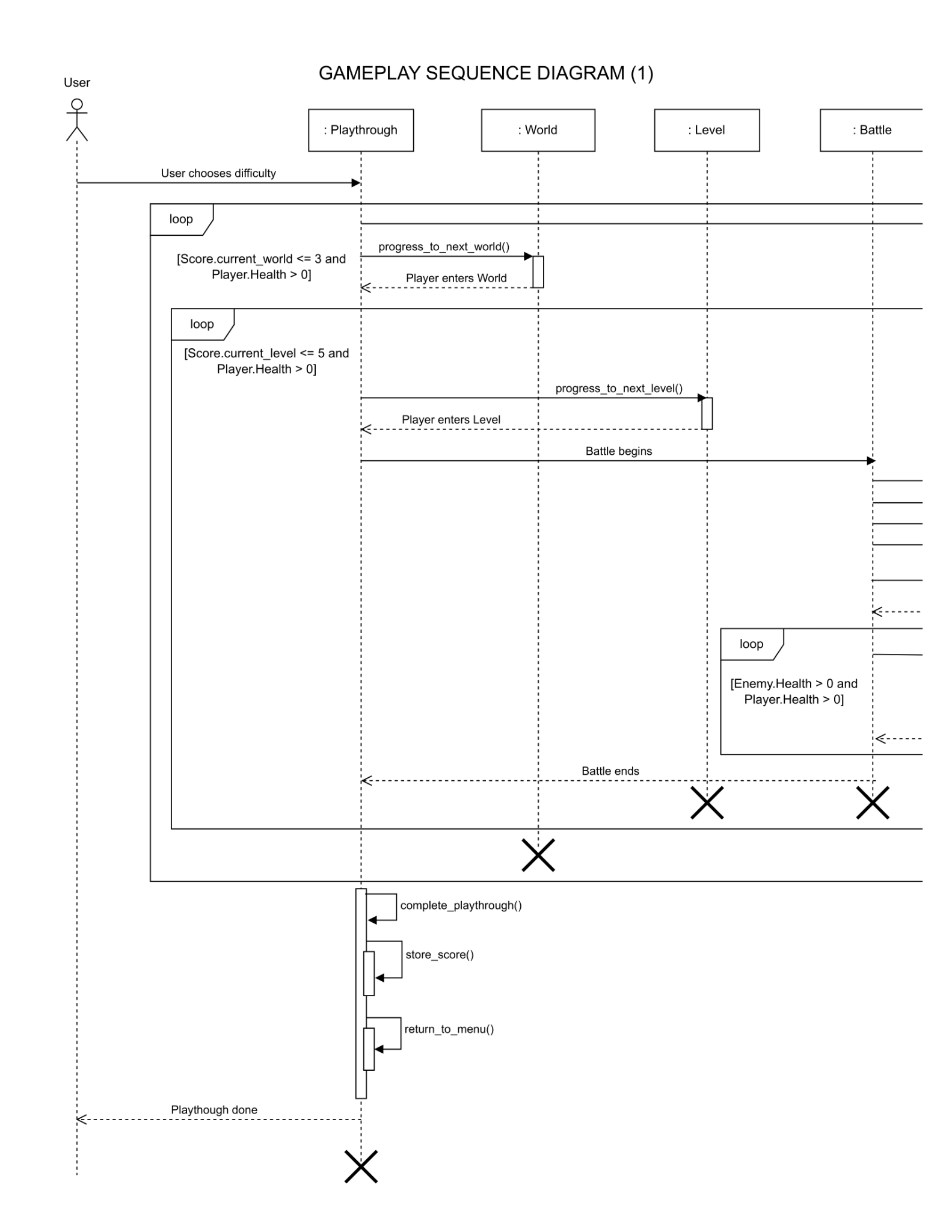
## Sequence Diagrams

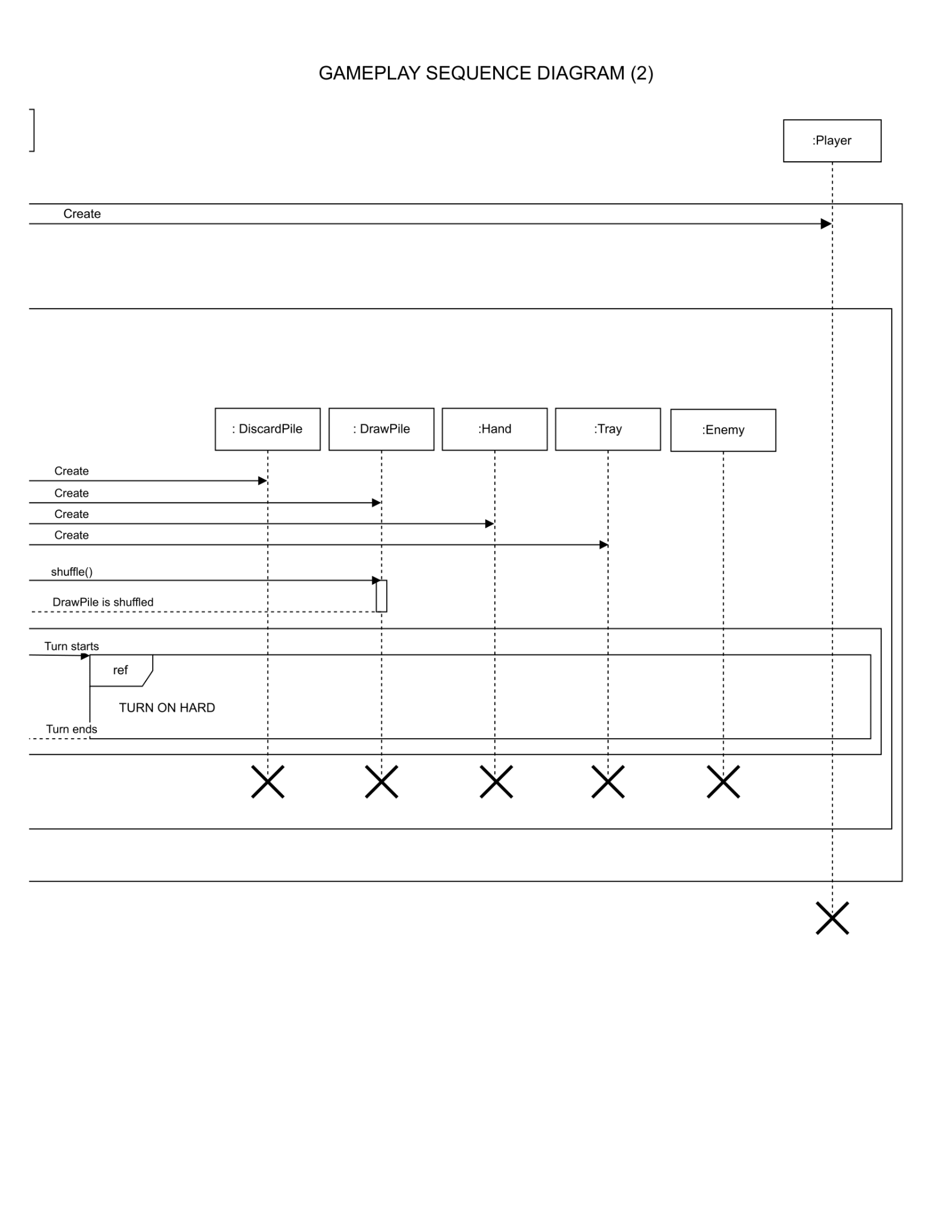
The following sequence diagrams give visual representations for some of the major scenarios of the system and show the interaction between objects in the game over time. The diagrams are too large to fit on one page so they have been broken up into subpages. Look at the numbers prefacing the diagrams and match them to the diagram title numbers to see how they are to be read. The number prefacing the diagram show the location of each page in relation to the others

For easier viewing visit the following link: [Sequence Diagrams](https://ltorrettor.github.io/GooseGauntlet/#heading-sequence-diagram).

* + 1. Gameplay Sequence Diagram

1 2

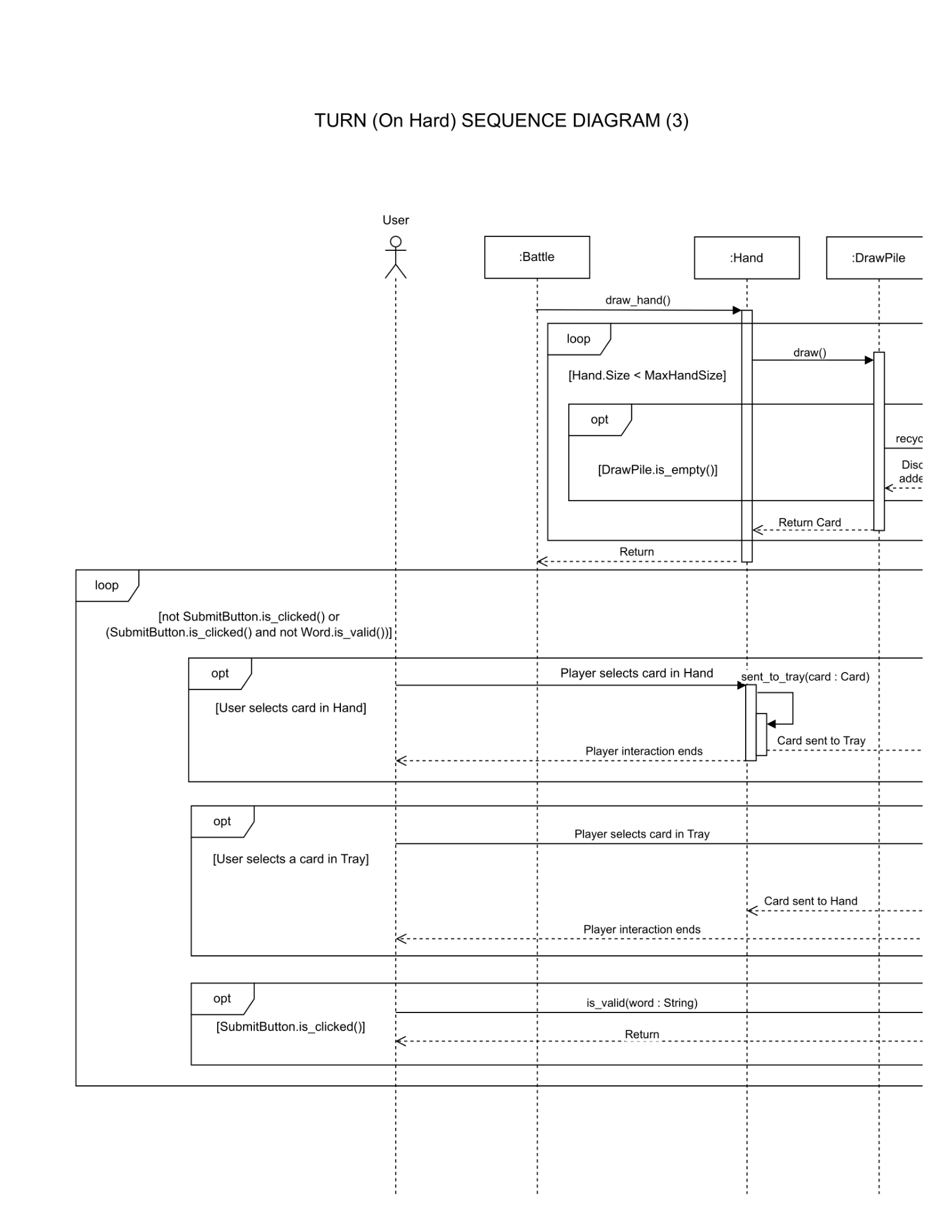


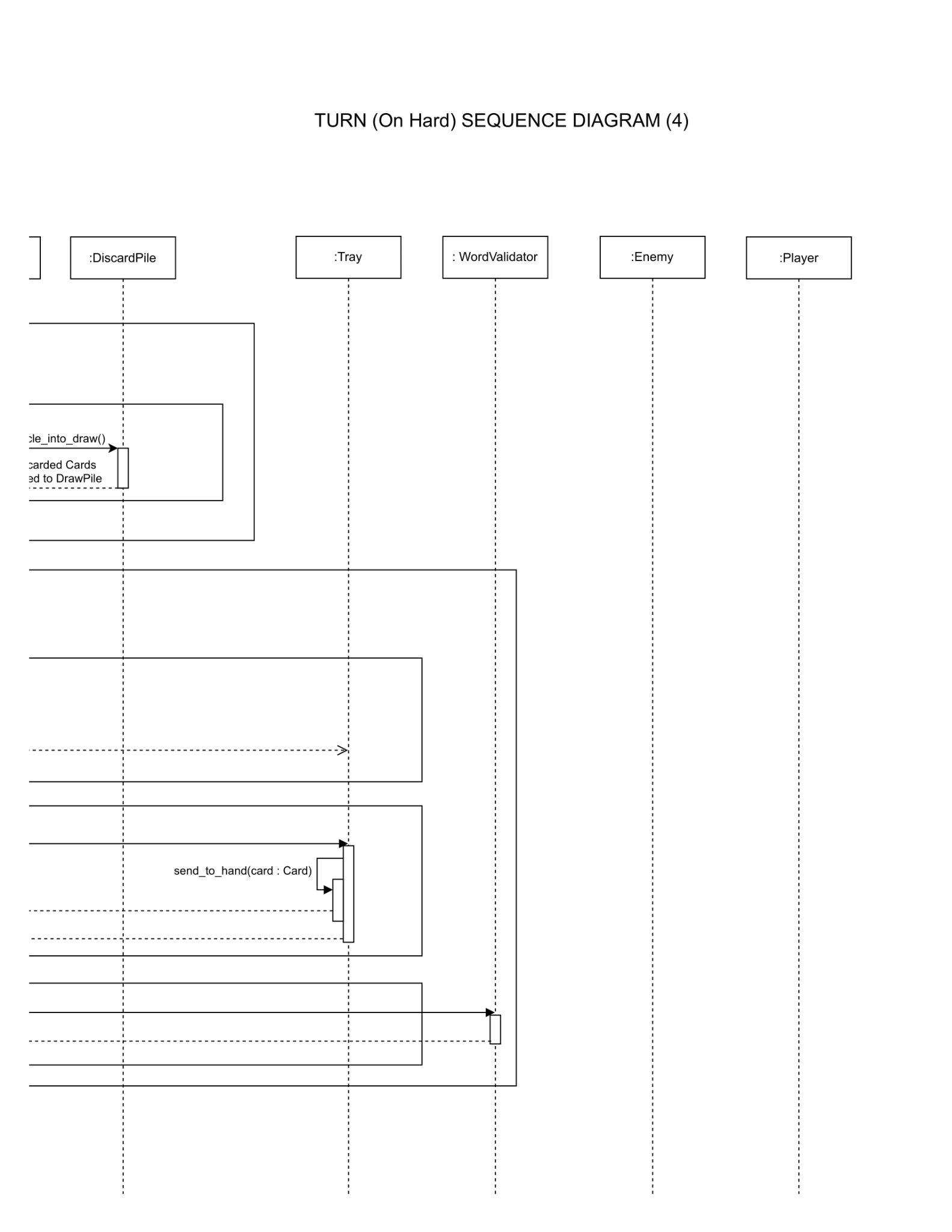


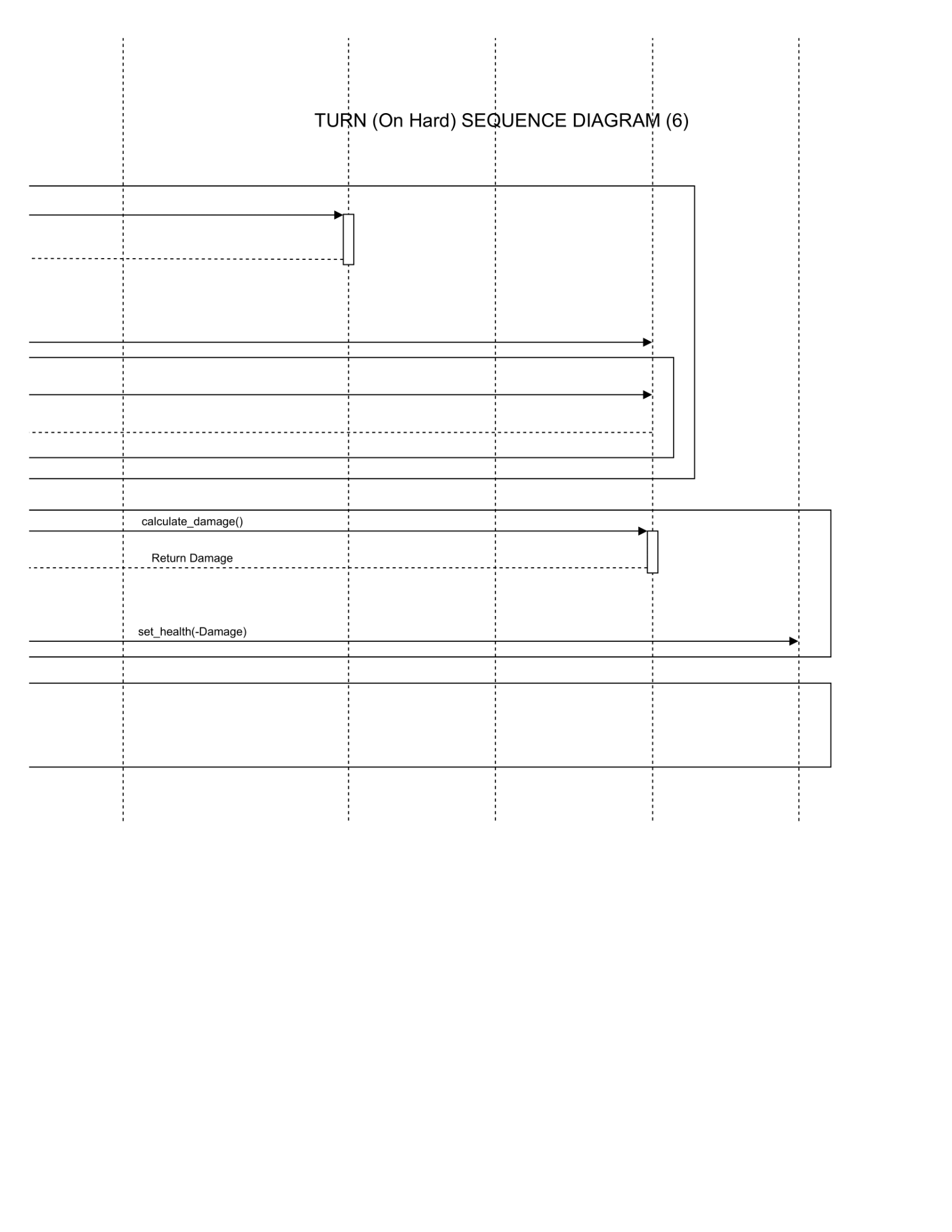
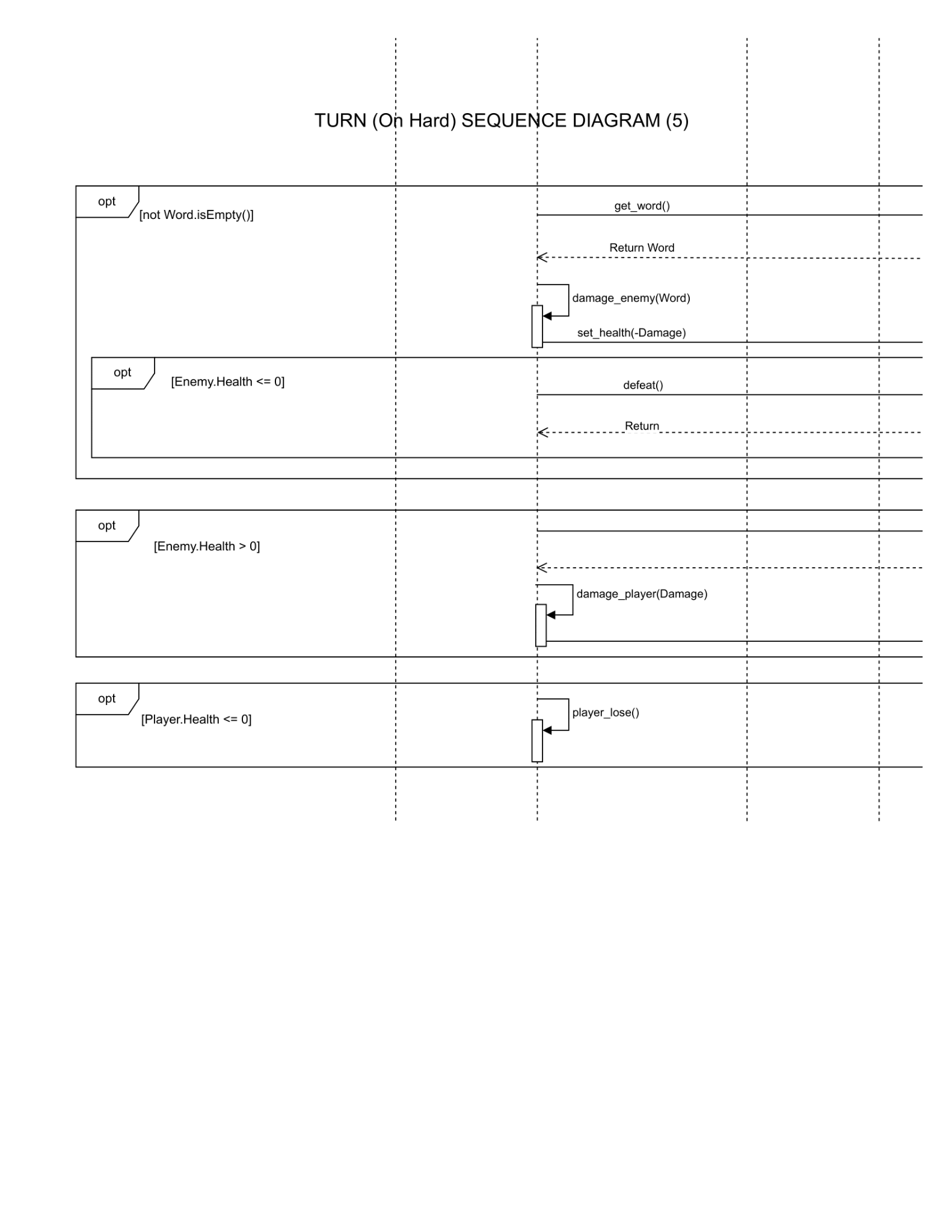
* + 1. Turn (On Hard) Sequence Diagram

3 4

5 6



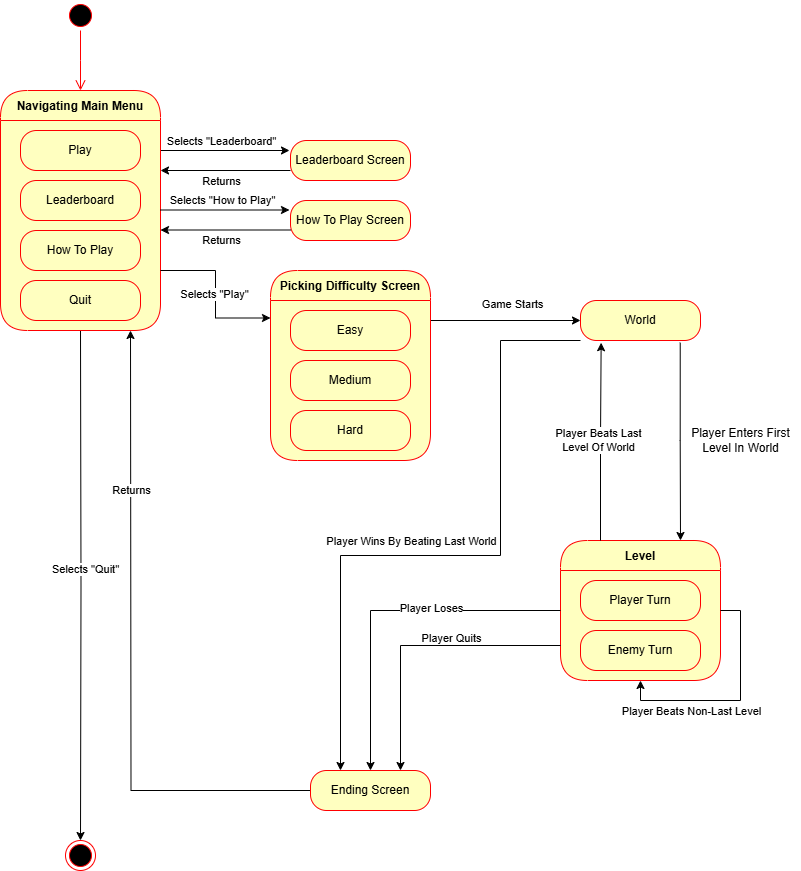




## 

## State Diagrams

The state diagram shows the game’s possible states, which dictate the game’s behavior, and how the game transitions between them.



# Prototype

The prototype will show the major UI elements of the final product. The user will be able to see the main menu and interact with the “Play”, “Leaderboard”, “How To Play”, and “Quit” buttons. The “Quit” button will exit the game. The “Leaderboard” and “How To Play” buttons will navigate to their respective screens and display their associated information. The “Play” button will navigate the user to a difficulty selection screen, where the user will choose their difficulty, which will then navigate them to the battle scene. The battle screen will display the UI features typically found in each battle, which includes the following:

* The player
* The enemy
* The draw pile
* The discard pile
* The hand
* The tray
* The cards in the hand or tray
* The submit button

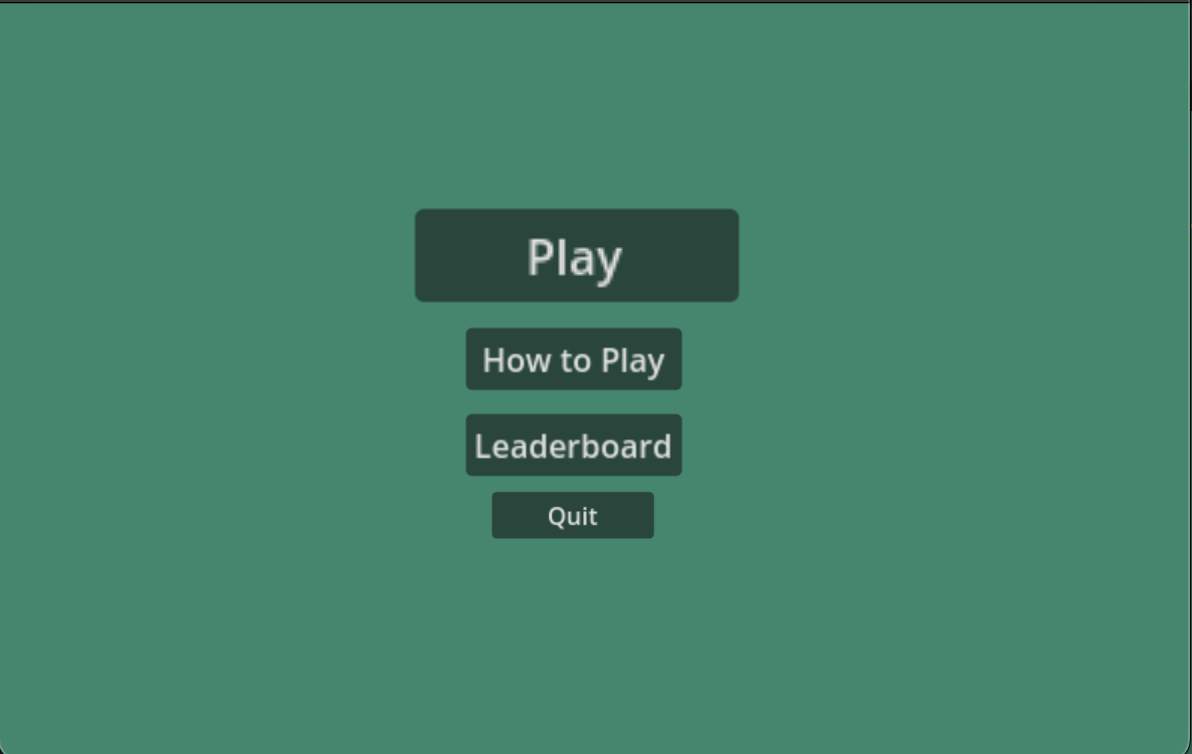
Each screen will also have a “Return” button that will navigate the user back to the main menu.

## How to Run Prototype

To run the prototype the user can either download as an executable or accessed via our website under the “Playable Prototype” tab at the following link: [Goose Gauntlet Website](https://ltorrettor.github.io/GooseGauntlet/). It is suggested the user be running Windows 10 and above and has a stable internet connection.

## Sample Scenarios

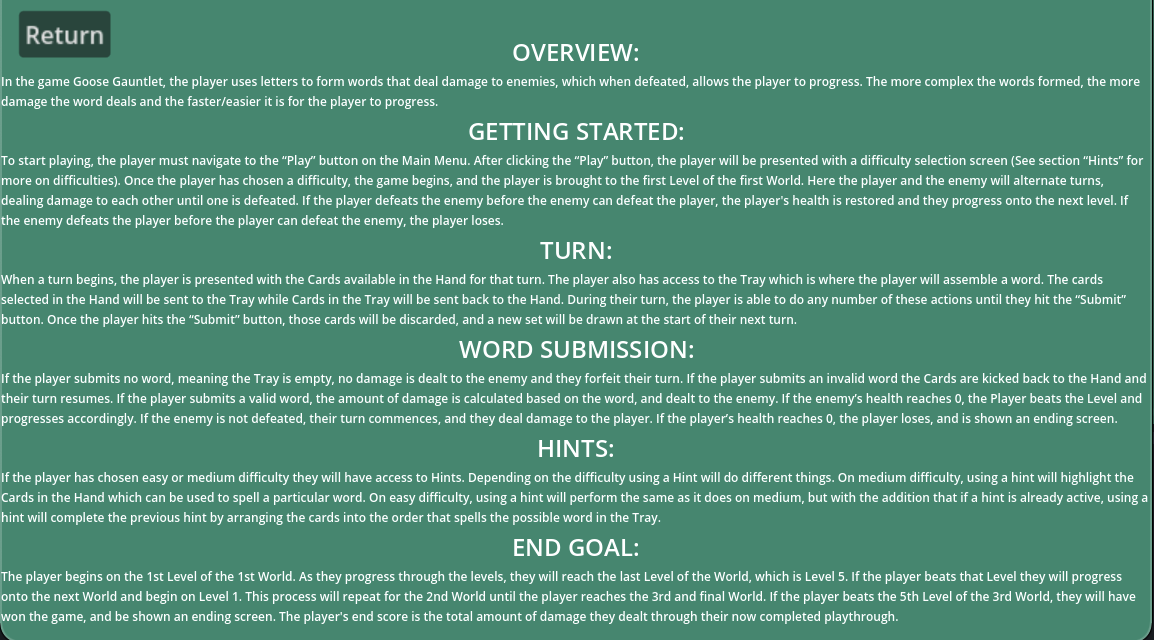
The user will first be presented with following main menu screen:



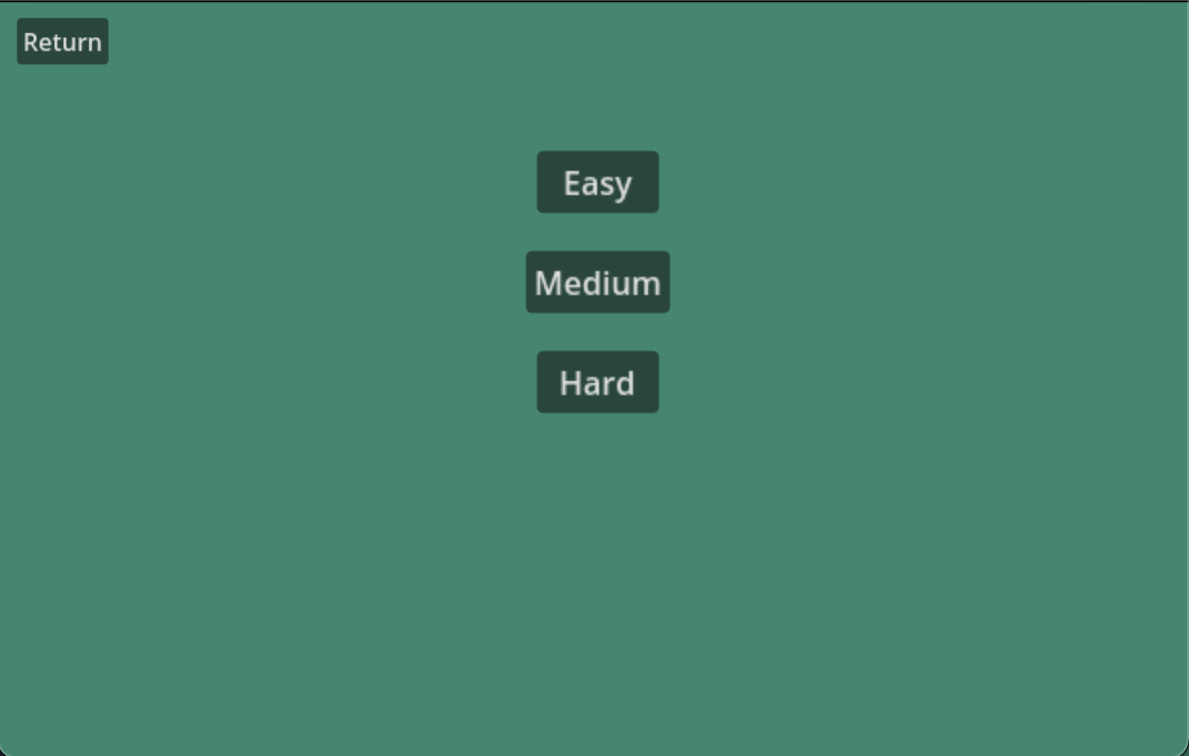
The user can navigate to the “Leaderboard” screen:



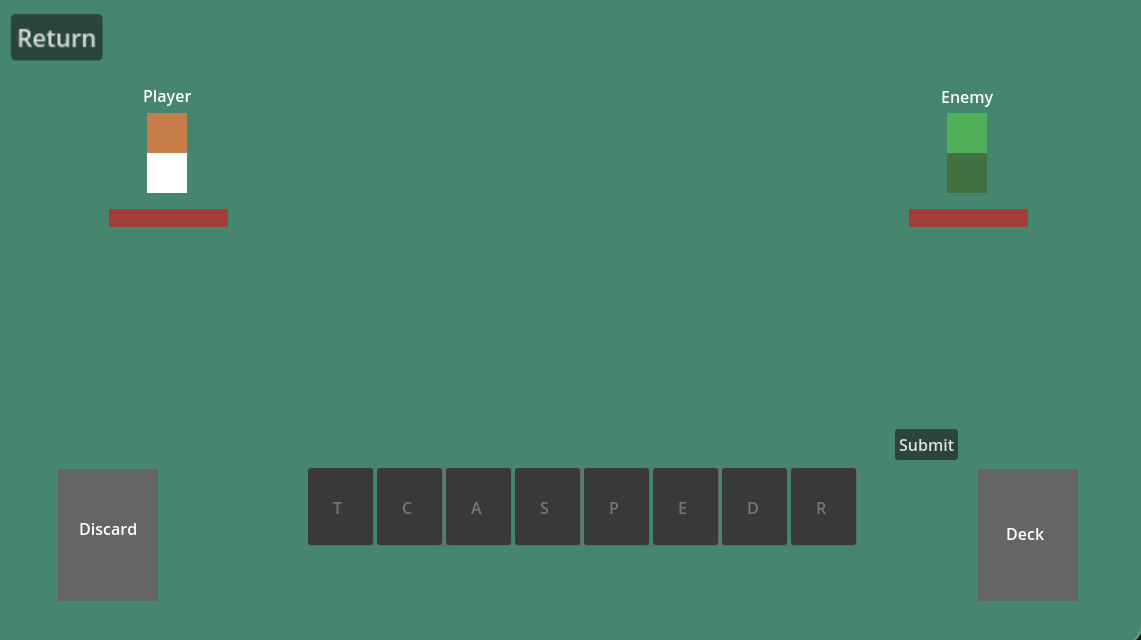
The user can navigate to the “How To Play” screen:



The user can click “Play" and then choose a difficulty:



In a typical battle the user will enter the scene and encounter an enemy:



The user can select cards to add them to the tray:

*\* Implemented in later prototype/final game\**

The user can use a hint if on easy or medium difficulty:

*\* Implemented in later prototype/final game\**

The enemy will take damage after the user submits a valid word:

*\* Implemented in later prototype/final game\**

The player will take damage if the enemy has not been defeated after the player’s turn:

*\* Implemented in later prototype/final game\**

The player will be shown an ending screen if they have lost or won:

*\* Implemented in later prototype/final game\**

# References

Website: <https://ltorrettor.github.io/GooseGauntlet/#heading-bibliography>

1. Draw.io - free flowchart maker and diagrams online. Flowchart Maker & Online Diagram Software. (n.d.). https://app.diagrams.net/
2. Godot Engine. “Free and Open Source 2D and 3D Game Engine.” Godot Engine, 13 Nov. 2024, godotengine.org/.
3. “Edutainment Game.” Goose Gauntlet, 2024 https://ltorrettor.github.io/GooseGauntlet/project-description.pdf
4. “English Language Arts and Literacy.” Massachusetts Curriculum Framework – 2017, 2017, www.doe.mass.edu/frameworks/ela/2017-06.pdf. Massachusetts Department of Elementary and Secondary Education. “Vocabulary and Morphology.” Skills for Early Reading: Vocabulary and Morphology - Evidence BasedEarlyLiteracy, www.doe.mass.edu/massliteracy/skilled-reading/language-comprehend/vocab-morphology.html
5. M. D. of E. and S. Education, “Skills for Early Reading: Vocabulary and Morphology - Evidence Based Early Literacy,” www.doe.mass.edu, Feb. 16, 2023. https://www.doe.mass.edu/massliteracy/skilled-reading/language-comprehend/vocab-morphology.html
6. “Slay the Spire on Steam.” Slay the Spire on Steam, store.steampowered.com/app/646570/Slay\_the\_Spire/

# Point of Contact

For further information regarding this document and project, please contact **Prof. Daly** at University of Massachusetts Lowell (james\_daly at uml.edu). All materials in this document have been sanitized for proprietary data. The students and the instructor gratefully acknowledge the participation of our industrial collaborators.