Primer Proyecto de Programación

Gwent-Pro

MATCOM-2024

Barbaro Yoel Martínez González C-122.

Introducción :

Esta guía está destinada a personas que quieran saber sobre la estructuración del proyecto,brindando una descripción general de la implemantación de la lógica del juego y la interfaz gráfica.

Gwent-Pro es un juego de cartas para dos jugadores inspirado en Las Cronicas de Narnia , desarrollado utilizando Unity como motor gráfico y C# como lenguage de programación.

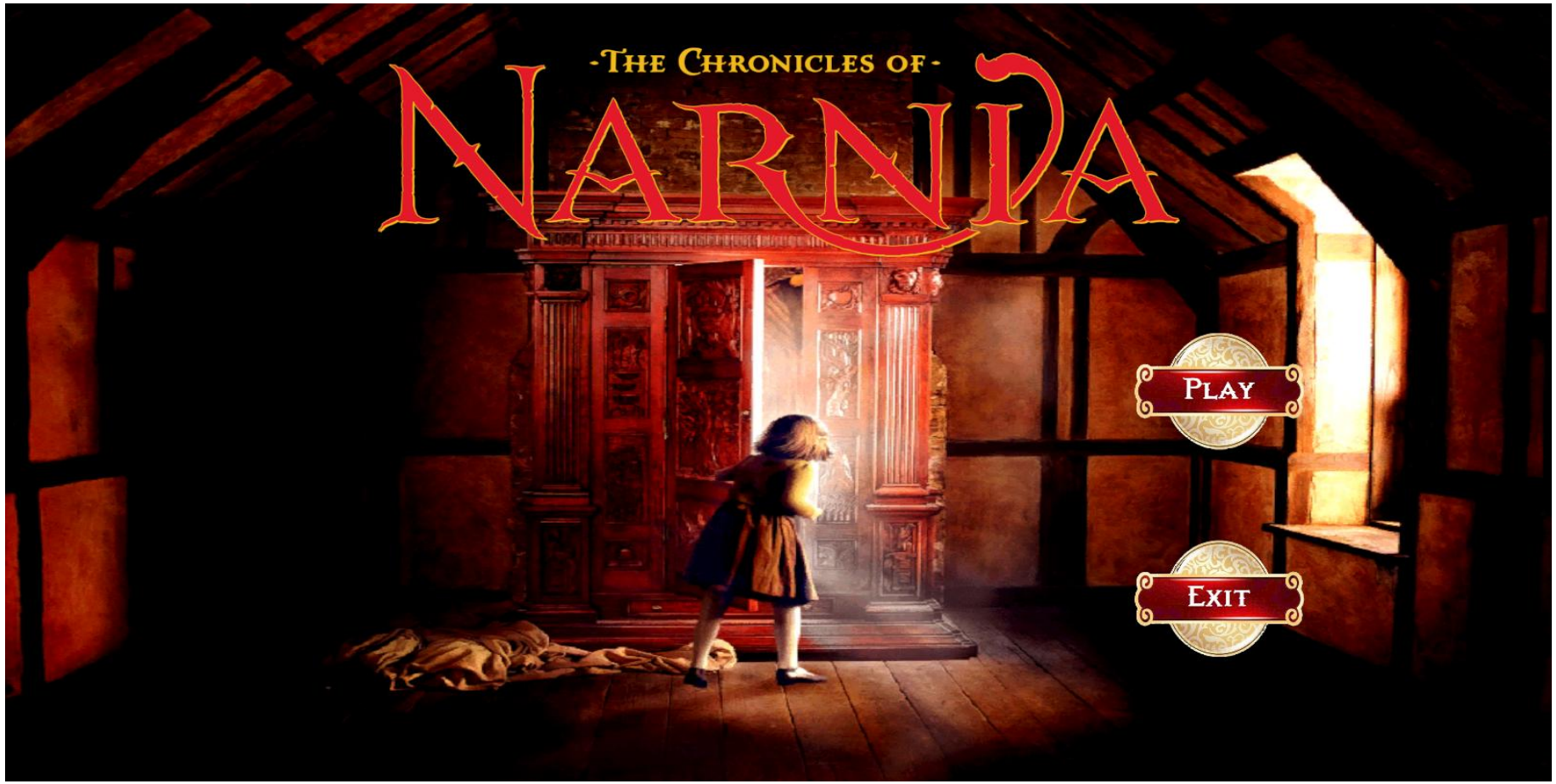
Descripción general del juego:

Gwent-Pro es un juego de cartas en 2D . Donde al empezar el juego los jugadores tienen que poner sus nombres y escoger la facción que los va a representar , luego tienen la oportunidad de cambiar hasta dos cartas de su mano y pasan a la escena principal ,el campo de batalla donde se lleva a cabo la partida . En cada turno , los jugadores pueden colocar una carta en el campo de batalla,activar la habilidad del lider o pasar la ronda . Al final de cada ronda , se suma el poder total de todas las cartas de unidad en el campo de batalla . El juagador con más poder gana la ronda .El primer jugador que gane dos rondas gana el juego.

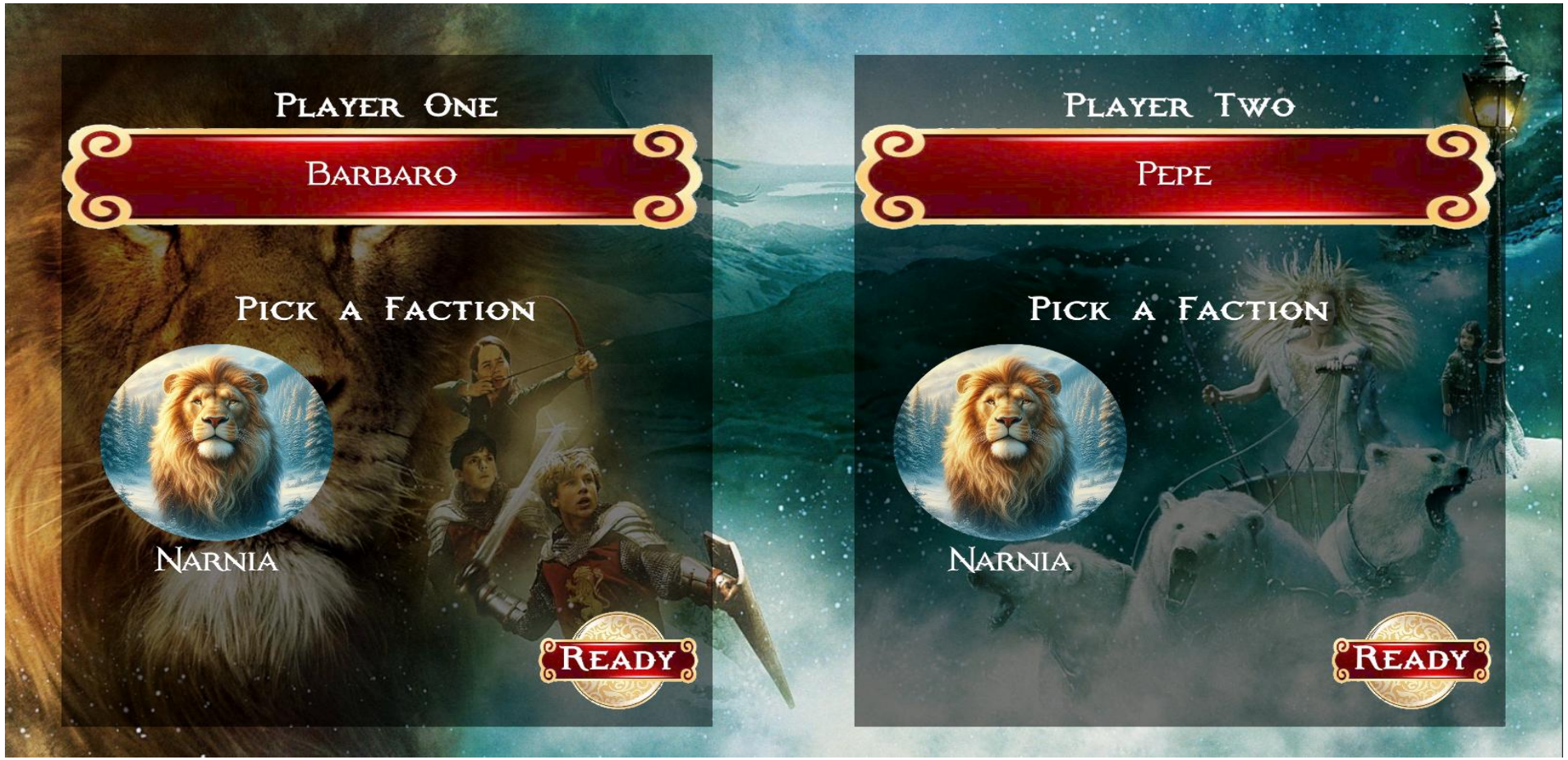
Desarrollo en Unity:

En esta parte del informe no nos detendremos mucho tiempo , no por restarle importancia , sino que no es el gran objetivo de este trabajo. Para el desarrollo del juego fue necesario investigar sobre esta tecnología y familiarizarce con conceptos como ScriptableObjects donde almaceno la información que contienen las cartas y los decks , GameObjects , y multiples herramientas de UI que nos binda este potente motor gráfico , donde gracias a estas herramientas fue que se logró el resultado que les mostraré a continuación.

Menú:

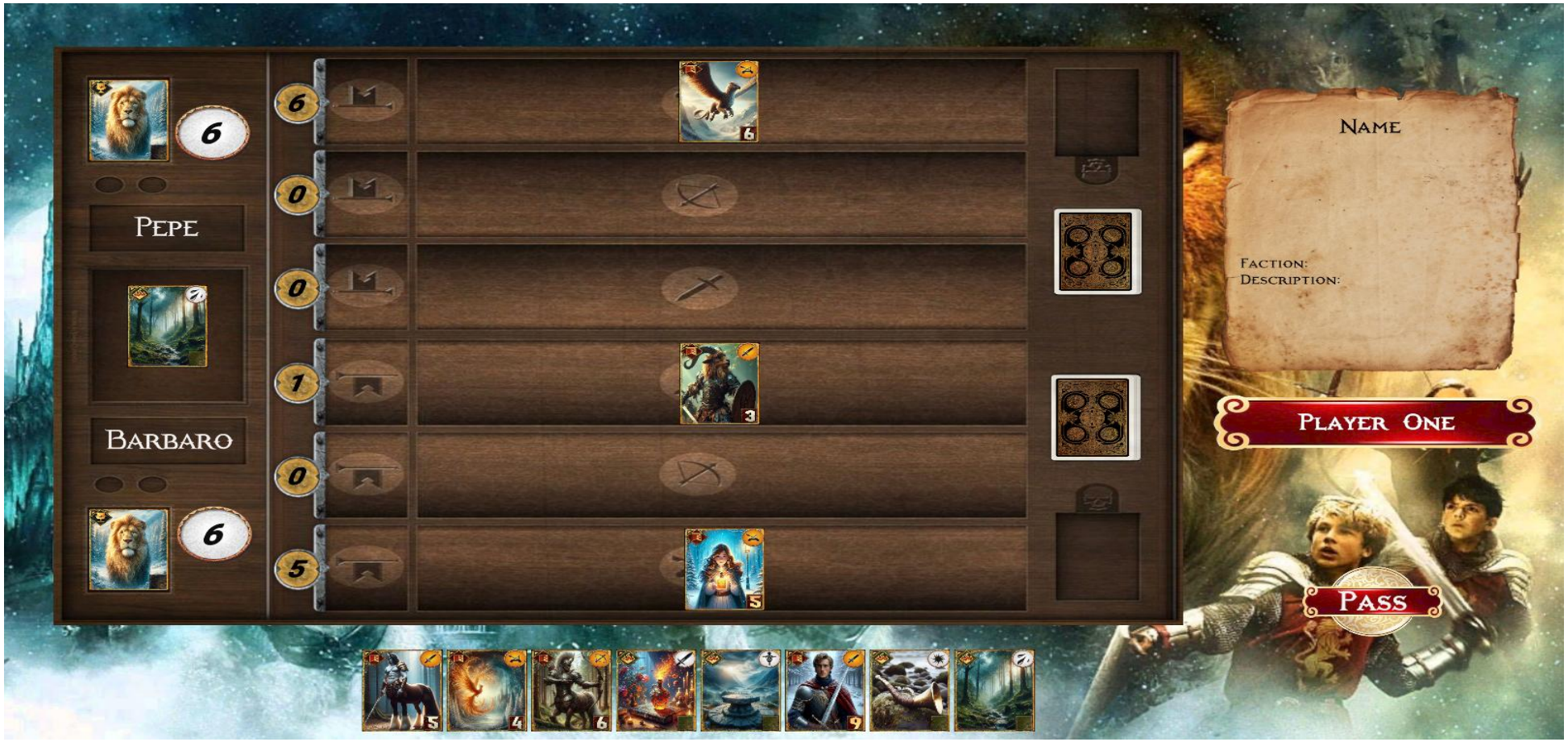


Seleccionar jugador:



Cambio de cartas:

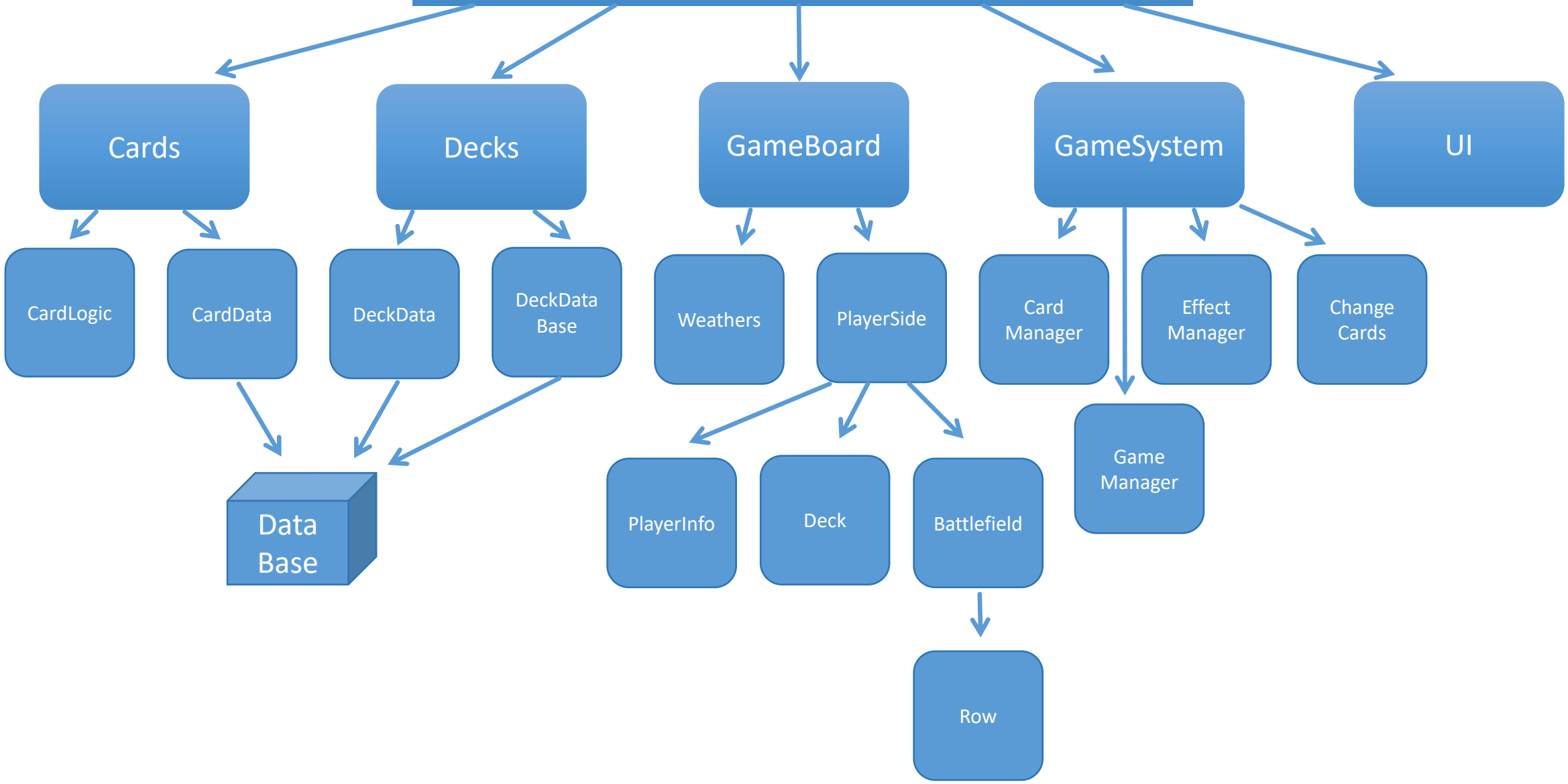
Juego:


6
Pepe
Barbaro
6
6
0
0
1
0
5
Name
Faction:
Description:
Player One
Pass

Implemanetación de la lógica del juego:

Para poder desarrollar el juego fue nesecario pensar que nesecitaba para tener un juego de cartas , como podía separarlo en pequeños componentes con funcionalidades distintas para terminar teniendolo todo . Como resultado de esto el juego se separa en cuatro grandes componentes que a su vez contienen otros más pequeños , además de una base de datos donde se almacena la información de las cartas y los decks ,y otro componente dedicado a la UI , aunque no contiene todo lo relacionado a este tema . A continuación les mostraré más a detalle esta información , sin llegar a temas muy profundos en general.Al final son los componentes del juego con sus funcionalidades ,para saber más sobre ellos pueden acceder directamente a los Scripts del juego.

Estructura del juego:



Cards:

Este componente se divide en dos elemenetos principales una CardData encargada de que tener la información y propiedades de cada una de las cartas donde se separan según su tipo heredando todas de CardData.

Y otro elemento CardLogic encargado de la lógica de las cartas en el juego , capaz de saber cual es su rol en cada momento y está muy vinculada con el componente CardManager del GameSystem.

Decks:

Este componente se divide igual en dos elementos uno DeckData encargado de almacenar cada una de las CardData y otro DecksData Base encargado de almacenar cada uno de los decks del juego .

DataBase:

No es uno de los grandes componentes del juego pero se encarga de contener los ScriptableObjects que contienen la CardData y los Deks y es muy útil para el correcto funcionamento del juego , además que permite un rápido acceso a esta información.

GameBoard:

Es el componente más grande en cuanto a cantidad de elementos pues contiene todos los elemtos del tablero . A grandes razgos contiene tres elementos PlayerOneSige , PlayerTwoSide ,Weathers.

Weathers:

Este componente se encarga de todo lo relacionado con los Weathers , controlando su funcionamiento en los Battlefields de cada jugador , y teniendo métodos para el control.

Algunos métodos de Weathers:

- ActivateClearing()
- ActivateRain()
- DeactivateRain()

Entre otros con la misma idea de activar y desactivar cada uno de los climas según sea necesario.

PlayerSige:

Contenido dentro de Board y que lo hace tan grande es el PlayeSige, el es uno de los componetes más importantes ya que conoce y controla la información del jugador , su deck y su battlefield.

Donde cada uno forma parte de un script con propiedades y métods diferentes .

PlayerInfo:

Este script contiene el Nombre del jugador , el lider de la facción , el poder total del Battlefield y la cantidad de victorias del jugador .

Deck:

A diferencia con el otro deck , el cual almacenaba la infromación de las cartas, este podrá manejar la lógica en el juego , siendo capaz de instanciar las cartas , y permitiendo robar una carta del deck o añadir una .

Battlefield:

Compuesto por tres filas cada una dedicada a un espacio en el tablero , y métodos los cuáles nos permiten acceder a la información que nesecitamos rápidamente .

Algunos métodos de Battlefild:

- BattlefielPower()
- GetRowWithLeastUnit()
- GetStrongesUnit()
- CardAppearances()
- NumberOfCardSOnTheBattlefiel()

Row:

Este script responde directamente a el Battlefied , capáz de brindar información tan valiosa como el poder de la fila , si están o no activados los climas o los aumentos , además que almacenan las cartas jugadas .

Algunos métodos de Row:

- AddUnitCard()
- RemoveUnitCard()
- TotalRowPower()
- ActiveWheather()
- ActiveIncrease()

GameSystem:

El corazón y el celebro del juego compuesto por cuatro scripts, encargado de manejar las cartas y sus efectos , además de controlar todo el flujo del juego .

CardManager:

Es uno de los scripts de GameSystem , encargado del funcionamineto de las cartas y sus moviminetos según el estado del juego.

Algunos métodos de CardManager:

- InvokeCard()
- SentToGraveyard()
- SentToHand()
- DestroyCard()
- InvokeDecoy()

ChangeCards:

En cargado de gestionar las cartas en el cambio de cartas el inicio del juego , que tiene con método más importante ChangeCard() , responsable de hacer el cambio de las cartas .

EffectManager:

Encargado de gestionar todos los efectos de las cartas ya sean lider o de unidad .

Algunos métodos de EffectManager:

- ActivateUnitEffect()
- ActiveteLeaderEffect()
- Draw()
- MultiplyPower()
- ClearStrongestUnit()

GameManager:

Encargado de manejar todo el fujo del juego , cuenta con un enum para determinar los diferentes estados y un Switch dentro de ChangeState() para controral el juego y cuenta con otros métodos que lo complementan.

Algunos métodos de GameManager:

- ChooseRandomPlayer()
- ChangeTurn()
- PassTurn()
- TurnEnd()

- RaundWinner()
- RaundEnd()
- ContinueGame()
- GameEnd()
- ChangeHand()

UI:

Este componente contiene scripts dedicados a la intefaz de usuario y la interacción entre esenas .

UI está compuesto por :

- AudioManger
- CardPreview
- GameData
- PlayButton
- QuitButton
- ReadyButton

Conclusiones :

El desarrollo de este juego ha sido una experiencia valiosa .El proceso ha desostrado la importancia de la planificación y que cuando tengamos un problema muy grande podemos dividirlo en problemas más pequeños los cuales sabemos resolver, para vencer este gran problema.