**Highland AI : Design Document**

The core principals of this project is to create a character driven card game with a heavy emphasis on deck building and character identity.

A current issue with card games is the sense of disposability and delicate balance attributed to board control. For many collectable card games the balance of fun can quickly be thrown off by the lack of balanced power on the board, along with “**ramp-up**” mechanics (**ramp-up**: units/player having better access to higher power options the longer the game progresses). This makes some card games feel more like a race than a strategic battle, with a loss of momentum creating a sense of hopelessness for the losing party.

Summoning is usually a core aspect that plays with board control; trading units back and forth in a effort to keep control of the field. So if there is a card that a player particularly likes that may not be very good, it becomes a constant trade off of playing the cards they like and always losing, or winning more frequently with the cards that don’t quite interest them.

**Lingo**:

Actions -> Cards

Stack -> Deck

Unit -> Character

Utility -> Unit Resource for using Actions

Exhaust -> A Unit Status that leaves them vulnerable

**Current Game:**

The cards work through the UI, using buttons to process the actions. I’m still debating if a drag and drop system should be used. Being that these are Actions and not Unit cards, drag and drop may not really make sense. But the action trigger is pretty rough right now, and could be cleaned up.

Right now there is no turn switch, so that still has to be implemented, which can probably be done with a simple bool. But if we want to have it so you can have more than two players at a time, then we might need something a little more elaborate.

If we are doing online multiplayer, then we will need to have that in mind from the start and structure the code around that. We could use the Unity Asset Photon?

But our priority is to have the game in a playable state so we can play test the hell out of it. We don’t need much to get to that state, as we are already close to it. Maybe when we get together we should paper prototype a bunch to save us a lot of time down the road.

**Mechanics:**

The game is player vs player: 1v1. Each player has 4 units, and each unit has their own deck (currently referred to as “**Stack**” in the project). Each deck can be fully customized, drawing from it to have a max hand of 3 per turn (and maybe a 4th character card that is always accessible that will help further distinguish one character from another).

There are 4 main values: Health, Attack, Defense, and Utility.

**Health:** The total health of a unit. When it reaches 0, that unit is eliminated. All units on an opposing team must be destroyed to win the round.

**Attack:** Each unit will have their own attack value, with cards (also known as **Actions**) relying on that value for effects.

**Defense:** This mitigates damage. Example:

Damage Done = Damage – Defense.

Unit 1 Health = 20

Defense = 6

Unit 2 delivers 10 Damage.

10 – 6 = 4 Damage done to Unit 1, leaving them at 16 health.

Currently defense is removed by damage, and is recovered on the next turn. So if Unit 3 follows up and attacks Unit 1 for 6 damage, that full 6 damage will go through, as Unit 1’s defense is now 0.

**Utility:** This is your card resource. To use a card, you must spend Utility. To gain utility, you must discard a card. Typlically, cards give about half their spending value if discarded. So if it costs 6 Utility to use an action, you will gain 3 utility if you discard it instead. Because of this, your power does not grow passively, but through conscience decision making each turn.

**Exhaust:**

Still debating how punishing Exhaust should be, but when a unit spends all their cards in deck and hand, they go into exhaust phase, leaving them inactive for one turn. After that one turn, the discard pile is shuffled back into the deck and the cycle continues. Their defense might drop to 0 as well, but we’ll playtest that. This also opens a more interesting trade-off for deck size. Having a larger deck means more flexibility in your discards and less frequent exhaust phases, but less likely to draw the cards you need for that turn. A small deck allows for more consistency, but you become vulnerable more frequently.

Ideally, you would have a team with all different sizes of decks, so a defensive unit would have a larger deck that improves the defense of their offensive units, who in turn can take more risks.

**Current Phases:**

Game Start

* Turn Start
  + Check Units Exhaust Status of Units
  + Turn Start (Trigger Turn Start Effects)
  + Draw Actions(Cards)
  + Execute Actions (Use or discard Action cards)
  + End Turn (Trigger End Turn Effects)
  + Check Units to set to Exhaust
* End Turn (Turn Switch)

If(All Units in Team Eliminated){

Game ends, Player with units standing wins;

}

**Challenges:**

By removing some of the core properties that commonly exist in other card games, we open more design challenges in the process, such as pacing and power levels.

**Action Editor:**

The Action Editor allows for quicker card creation so we can quickly playtest ideas.

It currently works by having a base PreFab called **\_ActionCardBase**, which it then clones and modifies depending on what values you selected in the editor. Currently, every card is assumed to have a **Action\_Immediate** script connected to it. Originally the idea was that a Action would have one of three Classes (Action\_Immidiate, Action\_TurnStart, or Action\_TurnEnd) depending on the effect you wanted. But I may have Action\_Immediate required on all Actions, with Action\_TurnStart or TurnEnd being created through Action\_Immediate, since playing the action has an immediate effect, whether that be doing damage, or creating a passive Object that has an effect every turn.

