Glass Joe's Punch Out!

Genre: Sports/ Fighting / Arcade

Mechanics:

Attacking :

Attacking is your main way to win. You can attack with your left and right arms mapped to the Q and E keys respectively. This is because of your opponent having the ability to dodge to the left and right as well. So when your opponent tries to dodge to the left and you hit him fast enough with your left arm you can interrupt his dodge and give him a bit of extra damage.

Dodging and countering:

Knowing how which way to dodge and being able to react fast enough after a dodge is the main tactic for the player to win the game. Dodging to the right will be done with a press of the D key and to the left with a press of the A key. After a successful dodge you can counter your opponent and stun him by attacking him with either arm while his still in his attack animation.

Interrupting:

When your opponent triggers an attack animation you have a really short chance to interrupt him with your own attack. Since your character hits a little faster this has the chance to land before he lands his hit on you and can result in stunning the opponent for a little bit of time.

K.O. recovery:

In the original punch out you have to tap the buttons really fast to increase the HP you have when you recover. But since my version will have you go down in one hit you will instead have to tap the A and D keys to recover with more stamina. If you got a full stamina when you get up you will enter a special mode (let's call it being in the zone) where for a little bit of time you can dodge without your stamina decreasing. Of course every time you go down (limit being 3 times) it becomes harder to recover your stamina this way.

Design Strategy:

For this game I'm using Unity Engine 2018.2.6f1.

I'm gonna start with the GameManager to make sure all the ui works and get the scene transitions working. Then I'll move on to the player and finally the opponent(enemy)

The GameManager is going to take care of scene management , menu's ,pausing and the games timer. Therefore I think the Singleton Fits here.

For the Enemy I think the State (Machine) pattern will fit best because of it being a AI like system that need to be able to react on its own to be able to beat the player.

Game UML: