SGD Design Document

# High Level Information

## Director Name: Simon Whittle

## Game Name: Lazer Cut Ocillion Arena

## Game Summary:

*A paragraph or two describing your game. Think of it as a marketing blurb. Sell me.*

Top-down arcade-style arena fighter with shoot-slash-and-dash mechanics similar to Hyperlight Drifter and a neon aesthetic (plus some Hotline Miami vibes).

## Core Aesthetics:

*Describe the game’s core aesthetics. What kind of experience are you trying to create for your players, and how do you plan on delivering it? If you haven’t taken CS 4730, you may wish to watch this first:* [*https://www.youtube.com/watch?v=uepAJ-rqJKA*](https://www.youtube.com/watch?v=uepAJ-rqJKA)

I’m looking to present a relatively fast-paced combat experience where one fights waves of enemies with various, modular behaviors. The neon aesthetic should allow for high visual distinction between enemy, projectile, and hazard types by playing with bright, highly-contrastive colors against a largely black background. The intensity of the combat should be accented by Hotline Miami-esque edm-style music.

## Original Concept:

*Where did this game idea originate? I.E. Portal + dance dance revolution, or “A game where you are your own enemy,” etc.*

It originated from the idea that the last game I worked on, The Great Man Theory, could have been an equally fun project with smaller, more manageable scope if it had been an arena fighter. This idea that an arena fighter is a good, small-scale project developed primarily based on ease of implementation, hence the hope for highly modular enemy behavior design such that newer programmers can have a small, manageable task to complete in the form of a single module, rather than a whole, large, enemy behavior script.

# Gameplay

## Objectives:

*What are the high-level goals of each section of the game (reach the flag, collect all coins, etc…)?*

The goal is always to survive until the next wave by destroying enemies. Hopefully the difference in enemy types and the occasional boss battle (scope allowing) should provide enough differences in kind to maintain engaging gameplay.

## 

## Rules:

*What are the rules of your environment?*

* Everything always has a small amount of health that is visually rather than numerically communicated to the player.
  + Most enemies have one or two hit points and the player’s gun does 1 damage but the sword does 2 damage
  + Enemies visually change when damaged by changing color or dimming in intensity of color, etc.
  + The player can take one hit before death. After being hit, if the player remains safe for long enough, they will regenerate their hitpoint. Having lost the hitpoint is communicated by the screen tinting red.
* You always regain full ammo after some wave / round interval

## Resources:

*What resources do the players use (coins, time, points, etc…)?*

The primary resource the player uses will be ammo. I haven’t decided exactly what system I want for the player to regain spent ammo, but I think a limited ammo amount is vital to keeping the shoot-and-slash from becoming just shoot.

## Win Conditions/Outcomes:

*What are the specific goals of the player during gameplay? If you ca win, what defines “winning” (last level, kill all of the other players, etc…)? What determines doing well vs. doing poorly (score, victory/defeat record, level, etc…)?*

The main goal is always to blast without being blasted. I expect a quick iteration time for retrying waves with an expectation sometimes having frequent deaths. I am open to the possibility of a points system where perhaps one gets more points for getting through waves with fewer deaths or more sword kills, kills on bonus bots that appear and disappear Space Invaders mothership-style.

## Game Mechanics:

*Outline and describe any specific, major mechanics in your game.*

* Shooting
  + Shooting is going to be restricted by an ammo limit such that one cannot rely on it entirely. I haven’t determined whether there should be enemies that can be only destroyed by shooting, as one may not always have ammo. I expect there to be a cooldown on shooting as well, but the ammo limit allows it to be rather short.
* Slashing
  + Sword strikes should be more powerful than shooting, but more difficult to pull off. Potentially, sword strikes will help replenish ammo. I expect it would be reasonable to have some enemies only be destroyed by slashes. There will be a short cooldown on slashes but otherwise they are unlimited.
* Dodging
  + Enemy projectiles and abilities should be such that they can be dodged or countered somehow.
* Dying
  + Taking just a few hits (probably two) should kill the player. Thus there is a quick iteration time for waves and they shouldn’t be terribly long. Furthermore enemies should also die quickly unless they are bosses, in which case there will probably be modules or weak points that do die quickly such that the boss only dies after all or enough modules are destroyed.

## Controls:

*What controls types will the game accept? What are the basic control mechanics?*

I expect to work primarily with keyboard controls, though controller is not impossible.

Hotline Miami-style, wasd to move towards, left, right, and away from the mouse i.e. mouse determines facing and wasd moves relative to mouse.

Left click fires the gun, right click slashes the sword.

Space bar dashes in the direction pressed on wasd or towards the mouse if none are pressed.

## Extension:

*What modules of the game can be designed later, time permitting (levels, powerups, characters, etc…)?*

The game can always have more enemies and / or bosses designed. Furthermore special player powers and abilities could be added time permitting.

# Implementation Details

**Engine/Language:**

*What engine/programming language will you be using to build your game?*

Unity 2018 (assuming it’s not still kinda slow) and C#

**Team Requirements:**

*Briefly describe what type of human resources you will need to make the game.*

I’ll need programmers primarily, but I can do with a small and / or less experienced team as I expect there to be plenty of coding to do, but relatively low intensity coding. No complex AI or networking here.

Designers will also be useful for designing enemies, and player abilities or bosses time permitting. I also want to get stuff in place to allow designers to easily put together waves and rounds of enemies early in the project so they can also create those sorts of scenarios.

Sound designers would be useful, as the game will only be made more satisfying with good, home-made sound effects. Plus, neon aesthetic should allow for some nice, cool, sounds.

Musicians would be fantastic, but I know where to find some if none sign up. As I said, I want some good, intense bops to be blasting bots to.

The one category I can’t see much work for is writers. I can imagine some kind of narrative being added to the game in the form of perhaps an arena overlord that is speaking to you during breaks in the fighting and some sense that you are perhaps ascending through levels towards some surface goal or vice-versa. Definitely not necessary for the game, though, and perhaps even a distraction.

**Team Organization:**

*An overview of how you will structure and manage your team.*

Team structure depends highly on the size of the team and who is in it. If I have too many of any given portion of the team to deal with myself, I’ll delegate more experienced members to help manage. I hope to have a general scrum, though, where people can feel free to bleed across boundaries between roles and pitch in where able. In the optimal case, I’ll be able to do a bunch of work early on to have solid task lists to keep everything organized. Likely a google sheet deal.

**Version Control System:**

*What will you use for version control? We strongly recommend Git.*

Git, git, and more git. I will probably get new folks working with Github Desktop as it has a nice, simple interface. More experienced folks can use what they want.

**Scope & Timeframe:**

*Give a timeframe for your game’s completion, and briefly describe why you believe your project is of a reasonable scope to be finished in that time.*

So, let’s say I have 3 hours’ worth of meeting time during each week. There should be 12 or so weeks to work on the game, and thus 36 hours of working time (not including crunch nights). This is about the amount of good working time one can expect for a game jam (people sleep and eat and stuff), and I think this is a game that could be made effectively in a game jam’s worth of time. Furthermore, I have a nice, functioning prototype of the basic mechanics that I almost certainly spent less than even 12 hours on making, so 24 extra hours of one person working on this game should be enough to make a fully playable game.

**Art:**

*Is the game text based, side-scrolling/top-down 2D, 3D, or something else entirely? What type of assets will you need? What about music?*

2d, top-down sprite-based stuff. We can get away with making things just solid colors, so very simple sprite making or even just throwing together cool shapes with prefabs made of unity’s default sprites could be sufficient for making the sprites for this game. Some things will likely need some particle work like projectiles or beam effects. I’m comfortable doing those myself, or teaching the particle system to a newbie wouldn’t be too hard.

**Programming:**

*What types of programming resources will you need? Are you working from an existing codebase? What type of onboarding (if any) will be required for incoming programmers?*

Between now and the pitch night as I polish my demo up a bit to present it, I’ll decide whether I’d like to work off of the demo code or start from more-or-less scratch. I expect I’ll bring over a few things and concepts from the demo even in the second case

**Challenges:**

*Will anything in your game require overcoming significant challenges? If so, how might you and your team approach them?*

I think one of the main challenges is designing and putting together a wave-delivery system that is easy enough for the designers to work with and versatile enough to provide them the tools to make interesting wave structures. This ultimately isn’t too difficult, but I’ve not worked up a good solution yet.

In a more broad view, keeping up with the designers should be a challenge as it is for any game. I expect it to be an especially big challenge for this game as it seems to me a rather attractive project for designers, due to the promise of lots of enemies to design. Thus, if we have a lot more designers than programmers, there’ll definitely need to be something done. I hope to have enough editor work to keep designers busy, or perhaps even get them enough coding experience to do some of the lowest hanging fruit for the simpler behaviors.