



Design Document + Development Log

Mo Abdulaziz

A Search for A Star 2023 submission

Greetings, I'm Mo, and this is my submission for the A Search for A star project. Here I will present the full development cycle of my project called "Disorderly Combat". This will include the core design of the Unique Mechanic, further design of other underlying mechanics, a weekly progress report on the prototype as well as a testing period and polishing period. The last segment will detail the publishing the showcase video as well as some final thoughts.

Contents

Rough Project Plan	2
Week 1 - Initial Conceptualisation	3
Week 2 – Further Design	10
Week 3 – Extended Design & Prototype Planning	24
Week 4 – Level Design and UI Design	30
Week 5 – MVP and Design issues	45
Week 6 – Further Development.....	56
Week 7 – Further Development & Design Issues #2	61
Week 8 – Polishing & Publishing.....	66
Asset References	72

Rough Project Plan

Week 1 - Planning + Initial Design

Week 2 - Futher Design

Week 3 - Other System Design

Week 4 - Level & UI Design

Week 5 - MVP

Week 6 - Futher Development

Week 7 - Futher Development

Week 8 - Polishing + Publishing

Week 1 - Initial Conceptualisation

Conceptualisation refers to the **pre-production** of the game. Here, I will detail my initial thought process. I will also detail the project goal, provide an executive summary, design the unique mechanic's core loop, and design gameplay pillars that will influence the design of this mechanic.

Goal

Analysing the brief, what stands out is the importance of a core mechanic. Hence the goal of this project will be to design a satisfying core mechanic that is creative, unique and has a lot of depth.

The goal of this project is to answer the design question, "Can I make an engaging combat system within a first-person shooter?". I love designing gameplay systems, specifically combat systems. Systems like Bayonetta's combo system and Devil May Cry's engaging style system have me in awe to this day. Of course, I'm also taking part in this jam to develop my design skills further. This program is an extremely useful chance to show off my design skills to industry professionals. During the next 8 weeks, I want to show that I can:

- Conceptualise and design an interesting core mechanic.
- Let the core mechanic influence the further design of the game to create a cohesive experience.
- Develop a functional prototype in the Unreal Engine.
- Document the process in a design document to show my thought process.

Core Concept

It all begins with an idea. Before delving into detail of a unique mechanic, some initial concepts should be drafted as a foundation. Where I would like to start is to imagine what genre this mechanic could feature in.

The genre I would like to focus on is the **Movement First-Person Shooter**. Movement FPS are shooting games where you have a high degree of control and mobility. They also feature a wide array of foes to challenge the player. These games include advanced movements such as sliding, wall running and double jumping. Examples of games like these include, ULTRAKILL (2020), DOOM (2016), Neon White (2022).



I have decided centre this mechanic on the Movement Shooter genre because I enjoy the sense of thrill and adrenaline these games provide. These types of shooters also ooze with style and a sense of cool. Also, these games usually revolve around only having one central mechanic. Neon white is an example of having the design philosophy that tailors to the brief, as the game revolves around a single core mechanic where you can discard cards to use as weapons or movement. Cards can be found in the level and are dropped from enemies. Levels are designed around the use of this single mechanic. Hopefully, this document will showcase how I can develop my own game using the movement shooter as a basis.

Executive Summary

An executive summary is a short paragraph or two that will describe the core features of a game. These summaries are a useful way to inform developers and players on what your games about and what makes it different. For Disorderly Combat, I'm writing this summary now so it can serve as a basis for the rest of my design decisions. An example of a summary of this game is written below. This summary contains the rough idea that I have for this brief:

Disorderly Combat is a fast-paced 3D movement shooter where you can have the power to manipulate energy to your advantage. In this game, the player's energy doubles as their health, with energy decaying each time they move. The player will have access to the Entropy System, a mechanic that allows them to drain energy from their surroundings to refill their own. This system also provides the unique ability to fire condensed energy as a damaging output. These take the form as "Entropy Weapons".

In Disorderly Combat, the player is tasked with eradicating foes in a level. By playing aggressive and stylishly, players can build "Discord", build enough and they can enter the "Discord State", this state providing bonuses to the player. To survive players must keep their energy topped up while using "Entropy Weapons" to manipulate the terrain and take out enemies to reach the end.

Unique Mechanic Concept

With the basis set, the next thing to target is the core mechanic. In most Movement Shooters, the player is provided with a wide arsenal of powerful firearms. For this project, I will pack the offensive power of an arsenal of weaponry into a single mechanic, which I will dub, the "Entropy System".

Entropy System – In Detail

The core theme behind the entropy system is growth and decay. I want to design a core mechanic that's easy to use and understand. This core mechanic will also influence the design of other underlining mechanics such as the "Discord State". I want this mechanic to create a risk and reward relationship. This system can also be seen as a system that forces to player to focus on switching between playing on the offensive and defensive.

The premise of this mechanic is that the player will be able to **absorb** objects and **release** a damaging output. These functions will be tied to the mouse with [Right Mouse Click] executing absorption and [Left Mouse Click] executing releasing. The player has a resource meter that governs this system. By absorbing objects, the fill amount of the resource meter will grow. If the player wants to release an output, then this will cause the fill amount of the resource meter to decay. But the twist on this mechanic, what if this resource meter, was also the players health.

By absorbing objects, the player can regain health/energy, the player will also be able to release this energy/health as a projectile. However, releasing a projectile will cost the player an amount of energy. Allowing this resource pool to function as health will create a loop where the player can't abuse releasing since it may result in a game over, meaning they will need to pay attention to finding energy sources during a fight. If doesn't sound frantic enough, what if we also add the rule that by moving, health/energy also decays. This will create friction as the player needs to decide to move or keep attacking. It forces the player to constantly be on the lookout for sources of energy during a run. Its already starting to sound like a gameplay system, so let's create some design pillars to help supplement the design of the core mechanic.

Design Pillars

Design pillars help supplement the core design of the mechanic. Below are the design pillars for the core mechanic of “Disorderly Combat”.

Aggression

- During combat, the player should prioritise attacking. Playing it safe will often result in being stuck on the defensive. The Entropy System is the only way the player can be defeated. Enemies will be aggressive with multiple enemies attacking the player at a time, providing the player opportunities to gain more energy.

Frantic

- This pillar describes the pace and feel of combat. Combat encounters should feel fast and hectic. The player will be bombarded with many different obstacles during a level. To add to the chaos, the constant draining of energy will provide players with little time to think.

Mobility

- This pillar describes the feel of controlling the player character. Movement should feel snappy and twitchy. The player will have a bevy of different movement options such as double jumping, sliding and wall-running.

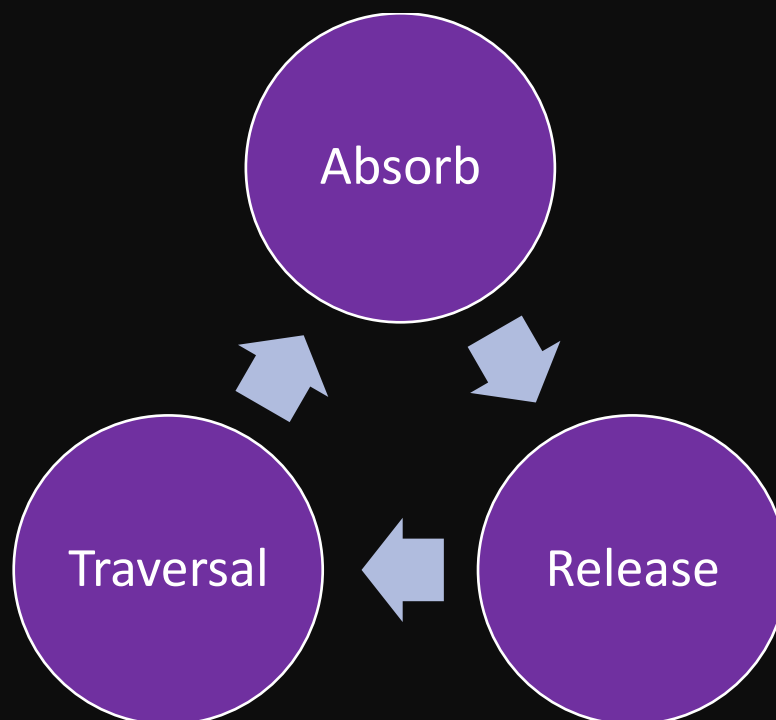
Experimentation

- The entropy system will contain entropy weapons with that each behave differently. Levels will be designed in a way where they can be completed in different ways using the different weapons the player can obtain

With the design pillars created, lets design a core loop to visualise the flow of the “Entropy System”.

Core Loop

A core loop helps people visualise the mechanical flow the unique mechanic. This loop will focus on the entropy system mechanic, showing the general flow of the core mechanic. Below is the core loop for Disorderly Combat:



Core Loop – In Detail

Traversal – This is starting point of the loop. Traversal refers to the player navigating way through the level. Unless movement is a core aspect of a game, “movement” or “traversal” would rarely be a substantial mechanic to detail in a loop, however due to the importance and reliance placed on movement in the Movement Shooter genre, traversal has been placed here. Moving will cause the player’s energy to decay over time. To regain energy, the player will need to look for objects to absorb to regain lost energy. Leading to our next segment.

Absorb – Absorb refers to the player actively absorbing energy from objects. Players can absorb energy by holding [Right Mouse Button]. Absorbing energy will refill the energy meter allowing for the player to continue traversal. If they so wish, they can release this energy. Objects will have different types of energy. Absorbing different energy will provide the player with an “Entropy Weapon” that corresponding to that energy type.

Release – Release refers to releasing stored energy. Players can release energy by pressing [Left Mouse Button]. Released energy will take the form of a “Entropy Weapon”. The properties and behaviour will change depending on the type of energy the player has absorbed. Releasing an entropy weapon will also remove a set amount of energy. This loop shows that gameplay will consist of a balancing act between releasing and absorbing. This is a good start, but when developing this mechanic, there are some design questions/problems. It’s necessary to solve these design issues to both balance this mechanic so it’s fair but also instills a sense of challenge onto the player. It will also help with developing the rest of the game. Below are some relevant questions placed into a chart.

Traversal	Absorb	Release
• What are the movement options in this game? • How much energy does the player lose per second of movement? • Do all movement options cause energy to decay?	• What can the player absorb energy from? • What consequence does absorbing energy from those objects cause? • Is all energy the same or will there be different energy to absorb? • Does this different energy have an effect on what you release? • How much energy gets refilled per absorption?	• What is the player able to release? • What is the cost of releasing energy? • Does the player release an particle or can it be something more thematically interesting? • Do particles have a tangeble effect on the level.

Traversal Options

In this section I will go into detail about movement design. This will include their movement options that they will be able to execute during the game. Below is a list of movement options that will/could be featured in this game and if they cause decay to energy. It’s important to note that during this design document’s creation, moves can be tweaked to not cost energy, due to balancing.

- Basic Move
 - Description: Player will move depending on their input. This will serve as the core movement option for the player.
 - Control: [W][A][S][D]
 - Energy Decay: Yes, moving will cause energy to decay over time. I have decided this because it will increase tension in the player.

- Jump
 - Description: Launches the player vertically into the air.
 - Max number of Jumps: 2, this will function as a double jump.
 - Control: [SPACE]
 - Energy Decay: Executing 1 jump won't cost energy, but for now it is planned that if the player jumps a second time, that jump will cost a set amount of energy.
- Dash
 - Description: Launches the player horizontally depending on the direction they're trying to go. Can be used as an evasive option. Can be used in the air. Player can dash again after a cooldown.
 - Max number of Dashes: 1
 - Control: [SHIFT]
 - Energy Decay: No.
- Slide
 - Description: Player slide across the ground in the direction they are facing. Sliding will be faster than basic movement.
 - Control: [CTRL]
 - Energy Decay: Yes, sliding should decay enemy by a small amount over time. Sliding should decay energy faster than basic movement since the slide is faster.
- Stomp
 - Description: Player will quickly launch themselves towards the ground.
 - Control: (While in air) [CTRL]
 - Energy Decay: No.
- Wall Run
 - Description: Player will move horizontally across a wall.
 - Control: Collide with wall.
 - Energy Decay: Yes, energy will decay over time.

Entropy System – Actions

In this section I will go into detail about what the player will be able to absorb/release.

Absorption

Absorption is the first half of the Entropy System. As mentioned in the loops, this section will serve to answer the questions proposed. By the end of it, we will have a detailed list of objects you can absorb energy from. Objects fit into two categories, static objects, and dynamic objects. Static objects are objects that do not move while dynamic objects do move. Examples below:

Examples of Static Objects:

- Obstacles
 - Description: Objects in the level that may hinder the players progression in the level. These can include walls, slopes, and platforms. These objects aren't deadly to the player.
 - Absorption Effect: If the player absorbs energy from obstacles, then that object will lose energy, and the player will gain energy.
- Static Hazards
 - Description: Objects in the level that are designed to harm the player. These can include spikes, disappearing platforms and trip mines.

- Absorption Effect: If the player absorbs energy from hazards, then that object will lose energy, and the player will gain energy. In rare cases, provide the player with a new entropy weapon.

Example of Dynamic Objects

- Hazard Projectiles
 - Description: Objects that are fired from static hazards. An example of hazard projectiles are bullets from an automatic turret.
 - Absorption Effect: If the player absorbs energy from hazards projectiles, then that projectile will dissipate, and the player will gain energy. Absorbing this energy will provide the player with a new entropy weapon.
- Enemy Projectiles
 - Description: Objects that are fired from enemies. An example of enemy projectiles are bullets from the gun of an enemy.
 - Absorption Effect: If the player absorbs energy from enemy projectiles, then that projectile will disappear, and the player will gain energy. Absorbing this energy will provide the player with a new entropy weapon.

The amount of energy gained from absorbing these objects will depend on the difficulty of absorbing it. The harder something is to absorb, the greater the energy gain. Below are all these options ordered by difficulty, danger, and energy gain.



Obstacles should give the player the least amount of energy since they're the easiest object to absorb energy from. Enemy Projectiles should give the most energy since they would be the most difficult to absorb due to particle movement also present the most harm to the player.

So, we know that the player can absorb energy from objects to refill their energy, but what I haven't touched on is the types of energy that will exist within the game. When the player absorbs energy from an object, I want each type of object to have a different type of energy. For example, if the player absorbs energy from a static hazard, that energy will change the properties and behaviour of the projectiles the player can release. We can classify these changes as "Entropy Weapons".

Releasing

In this section, I will detail the process of releasing energy in this game. When the player presses [Left Mouse Click], the player will fire a “Entropy Weapon”. For this game, I want each projectile the player fires to take the form of a weapon. Depending on the type of energy stored prior to releasing, the weapon will take a different form and will also have a different properties and behaviour. The way I will be designing the player’s projectiles is by detailing their spectral form, the cost of energy to release and the effects those projectiles will have in the world. Below is the first “Entropy Weapon” I want the player to start with.

Name	Entropy Dagger
Description	The starting weapon
Damaging	Yes, will deal the least amount of damage
Cost	The projectile will deplete a set amount energy when fired but will cost the least amount of energy to fire
Use	Can be used to damage enemies. Can also be used to fill up objects with energy. An example is if the player spots a platform that lacks energy, they can fire their daggers to give energy to the platform. The platform can then be jumped on.

Week 1 Retrospective

With the first week complete, I’m looking hopeful towards the project. I’ve proposed the goal for the project which is going to guide me along. I’ve identified which genre I would want the “Entropy System” to fit in. An idea for the general game idea as well as an executive summary has been documented. Most importantly, the unique mechanic for the game has been developed. Looking at feedback from friends I feel like the mechanic will be interesting. The brief has asked for a core mechanic is creative and has depth. The entropy system which involves absorbing and releasing energy has potential for depth. I believe this system counts a single mechanic because both actions rely on each other. I’ve tried to go for this sense of offensive/defensive play or growth/decay systems as concepts like this sound appealing to me. To me, an engaging combat system should have this distinct rhythm of attacking and defending. This system like this is that concept given mechanical form.

The question I had at the start was *“Can I make an engaging combat system within a first-person shooter?”*. I believe that the Entropy System is that first step in the answer. The Entropy system is engaging because it requires the player to interact with their surroundings to find situations that allow them to attack. Once they have attacked, their energy will be low so they will go back to interacting with the surroundings to find energy.

In the next section I will go even deeper, designing some obstacles that the player can manipulate as well as some enemies that can be featured in levels. I will also design more types of “Entropy Weapons” the player can release as well as their special effects. I will then start doing some numerical design so we can balance the power of “Entropy System”.

Week 2 – Further Design

This week marks the second week of **pre-production** for the game. We will be designing some more objects such as obstacles and hazards. I will also start designing some enemies as well as some more entropy weapons and how to gain them. The purpose of this week is to design some opportunities that will allow the player to use the Entropy System in an advanced way. Let's start off with some objects.

Objects – Obstacles

Obstacles are things that the player will interact with in the level. All objects will contain energy that the player can absorb. You can also provide energy to objects by firing "Entropy Weapons" at them. Here are some obstacles and hazards I have designed that could make it to the final prototype.

Name	Energy Crystal
Type	Obstacle
Description	Main source of energy in the Realm of Order
Stats	Energy Stored - High
Absorb	Players can absorb energy from this object. When absorbed, crystal will lose all energy and player will gain the energy stored from crystal
Release	Energy Crystals cannot be refilled with energy released from entropy weapons.
Energy Type	Entropy Dagger – Energy Crystals will provide players with Dagger energy. This energy will change the player's current entropy weapons for to the dagger.

Name	Pristine Platform
Type	Obstacle
Description	A floating platform that can either be hollow or filled with energy. When Hollow – Platform will have ghostly appearance. If player's attempt to jump onto it, they will pass through. When Filled – Platform will have a solid appearance. Players will be able to collide with the podium in this state
Stats	Energy Stored - Average
Absorb	Players can absorb energy from this object. When the podium loses all energy, it will become hollow.
Release	Players can release "Entropy Weapons" to fill the platform. Once it receives enough energy, it will become filled.
Energy Type	Entropy Dagger – Pristine Platform will provide players with Dagger energy. This energy will change the player's current entropy weapons for to the dagger.

Name	Pristine Wall
Type	Obstacle
Description	A floating wall that can either be hollow or filled with energy. Wall has same properties as the Ivory Podium. Difference is walls can be wall-ran on.
Stats	Energy Stored - Average
Absorb	Players can absorb energy from this object. When the podium loses all energy, it will become hollow.
Release	Players can release "Entropy Weapons" to fill the wall. Once it receives enough energy, it will become filled.
Energy Type	Entropy Dagger – Pristine Wall will provide players with Dagger energy. This energy will change the player's current entropy weapons for to the dagger.

Name	Pristine Launcher
Type	Obstacle
Description	A launcher that boosts the player jump height momentarily. Launcher has two states, hollow and filled. When Hollow – Launcher will have no effect. When Filled - Launcher will launch the player according to the amount of energy is inside it.
Stats	Energy Stored – Average
Absorb	Players can absorb energy from this object. When the launcher loses all energy, it will become hollow.
Release	Players can release “Entropy Weapons” to fill the launcher. Once it receives enough energy, it will become filled.
Energy Type	Entropy Fan – Pristine Launcher will provide players with Fan energy. This energy will change the player’s current entropy weapons for to the fan.

Name	Crystalline Orbs
Type	Obstacle
Description	Two orbs that are located next to each other. When both orbs are filled with energy, a crystalline chain will materialise and connect the two orbs, allowing players to surf on them. Chain will only materialise when both orbs are filled with energy.
Stats	Energy Stored - Average
Absorb	Players can absorb energy from this object. Orb will take a grey appearance. When absorbed players will gain a new type of weapon.
Release	Players can release “Entropy Weapons” to fill each orb. Once it receives enough energy, it will become filled. Once both orbs are filled, chain is formed.
Energy Type	Entropy Chain – Absorbing the energy from an orb will provide the player with then Entropy Chain weapon.

Name	Gravity Platform
Type	Obstacle
Description	Two orbs that are located next to each other. When both orbs are filled with energy, a crystalline chain will materialise and connect the two orbs, allowing players to surf on them. Chain will only materialise when both orbs are filled with energy.
Stats	Energy Stored - High
Absorb	Players can absorb energy from this object. Orb will take a grey appearance. When absorbed players will gain a new type of weapon.
Release	Players can release “Entropy Weapons” to fill each orb. Once it receives enough energy, it will become filled. Once both orbs are filled, chain is formed.
Energy Type	Entropy Chain – Absorbing the energy from an orb will provide the player with then Entropy Chain weapon.

Enemies

In Disorderly Combat, enemies will split into three types. Each type of enemy will serve a different purpose. I've decided to handle enemy design in this way so that enemies have enough variety to feel distinct. Also, each enemy could be seen as a gameplay test, with each enemy serving test the player. Each enemy will also have a unique entropy weapon that the player can gain. To obtain these weapons, the player will need to use the entropy system to their advantage. The harder Below are the three categories of enemies.

Smashers – Smashers are enemies that pose a low risk. They should be easily dealt with so the player can have some low-risk fun. For instance, these enemies can be taken down easily by using the base Entropy Dagger. Entropy Weapons can easily be obtained by these enemies.

Emphasisers – Emphasisers are enemies that emphasise the use of a specific tactic or weapon. They should encourage players to use specific weapons against them to defeat them. This isn't a necessity so they can still be defeated using the base dagger.

Enforcers – Enforcers are enemies which force the player to use a specific weapon or tactic to defeat them. This means the player can only defeat them using these weapons. With the other weapons having little or no effect.

Each enemy will have different stat attributes, each of these attributes go from very low to very high. These attributes serve as placeholder stats until I start doing some numerical design to give each enemy their unique stats.

Health – Total health points, once it hits zero, enemy is defeated.

Damage – How much damage one attack deals.

Fire Rate – How often enemy will attack within a given time.

Range – How far the enemy's attack reaches.

Speed – How fast the enemy can travel.

Name	The Trooper
Type	Smasher.
Description	A medium sized enemy that patrols the Realm of Order. Wields a magical staff.
Enemy Stat	Health – Average Damage – Average Rate of Fire – Average Range – Average Speed – Average
Enemy Behaviour	When the enemy spots the player. They will start to fire projectiles that home at the player and proceed to get closer to them.
Absorb	Players can absorb energy from projectiles this enemy fires. Absorbing the projectile refill, the player's energy gauge and will give the player an entropy weapon.
Release (Entropy Weapon weaknesses)	This enemy can be taken down using any entropy weapon
Projectile Energy Type	Entropy Dagger (<i>Edit: Now provides Entropy Dual Dagger</i>)
Strategy	Since this is a smasher enemy, there won't be a definitive way to take them down. Players can use any weapon at their disposal, including the Entropy Dagger.

Name	The Air Trooper
Type	Smasher.
Description	A medium sized enemy that patrols the Realm of Order. This version of the trooper hovers in the air. Wields a magical staff.
Enemy Stat	Health – Average Damage – Average Rate of Fire – Average Range – Average Speed – Average
Enemy Behaviour	When the enemy spots the player. They will start to fire projectiles at the player and proceed to get closer to them.
Absorb	Players can absorb energy from projectiles this enemy fires. Absorbing the projectile refill, the player's energy gauge and will give the player an entropy weapon.
Release (Entropy Weapon weaknesses)	This enemy can be taken down using any entropy weapon
Projectile Energy Type	Entropy Dagger (<i>Edit: Now provides Entropy Dual Dagger</i>)
Strategy	Since this is a smasher enemy, there won't be a definitive way to take them down. Players can use any weapon at their disposal, including the Entropy Dagger.

Name	The Chameleon
Type	Smasher
Description	A trooper that specialises in recon. Has the ability to turn somewhat invisible. Wields a short magical wand.
Enemy Stat	Health – Low Damage – Average Fire-Rate – Average Range – Very low Speed – Fast
Enemy Behaviour	When the Chameleon spots the player, they will turn invisible and move towards the player. While firing projectiles that shift in an irregular pattern. Every time they fire a projectile, they will be briefly visible again.
Absorb	Absorbing energy from the chameleons' projectiles will give the player energy as well as access to the Entropy Twin-Blade. Difficulty comes from trying to predict the projectiles trajectory before it shifts into another irregular pattern.
Release (Entropy Weapon weaknesses)	This enemy can be taken down using any entropy weapon
Projectile Energy Type	Entropy Twin-Blade
Strategy	The challenge from the chameleon comes from trying to spot them before they turn invisible. While avoiding the irregular projectiles.

Name	The Pursuer
Type	Emphasiser
Description	A nimble enemy known for its quick speed and high aggression. Wields a sword.
Enemy Stat	Health – Average Damage – High Rate of Fire – Very Fast Range – Very Low Speed – Fast
Enemy Behaviour	When the enemy spots the player. Enemy will launch at the enemy at full speed. This enemy is impervious to player attacks while charging at them.
Absorb	Player can absorb energy from the pursuer's strikes. Absorbing the strikes will give the player access to the Entropy Blade
Release (Entropy Weapon weaknesses)	While this enemy can be defeated using any weapon, due to the fast nature of the pursuer, they will be hard to hit while they are charging at the player.
Projectile Energy Type	Entropy Blade
Strategy	This enemy emphasises the use of absorbing at the right time. Pressing absorb right as they're about to strike allows the player to stun them. This also gives the player energy and a new weapon to use. Players can then retaliate, attacking the pursuer during their stun window.

Name	The Brute
Type	Emphasiser
Description	A slow but large enemy that can stop the player in their tracks with his defensive aura. Paladin also has a large suit of Armor that seemingly looks impervious to Entropy Weapons. Wields a boom-stick.
Enemy Stat	Health – High Damage – High Fire-Rate – Slow Range – Average Speed – Slow
Enemy Behaviour	When the brute spots the player, they will plant themselves into the ground and fire wide arc slow-moving projectiles. If the player gets into the range of the brute, they will knock the player back taking a chunk of their energy. If the armour is shattered, the brute can be defeated in a single hit of any weapon.
Absorb	Player can absorb energy from the brutes' slow-moving projectiles.
Release (Entropy Weapon weaknesses)	While this enemy has their armour on, they are impervious to entropy weapons. Only a select few weapons can dent the armour, while only one weapon can outright destroy the armour in a single hit.
Projectile Energy Type	Entropy Axe
Strategy	This enemy emphasises the use of absorbing specific energy to gain specific entropy weapons. The Brute wear's weapon-resistant armour. The Entropy Dagger will have no effect. The best strategy is to absorb the projectiles the brute fires to gain the entropy axe, then use that axe to destroy the brute's armour.

Name	The Scout
Type	Enforcer
Description	A long-ranged scout that has their eyes peeled waiting for susceptible foes to fall into their scope. The scout will always be in a location that is off the main course of a level. The scout wields an extremely large staff.
Enemy Stat	Health – Low Fire-Rate – Very Slow Damage – Very High Range – Very High Speed – Very Slow
Enemy Behaviour	When the scout spots the player, there will be a lens-flare effect, a laser sight and a sound cue that plays. Once the laser turns to a solid red the scout will fire their shot.
Absorb	Player can absorb energy the projectile of the scout. This gives the player access to the Entropy Lance.
Release (Entropy Weapon weaknesses)	The scout is susceptible to any entropy weapon, the problem comes from the distance between the player and scout. Most entropy weapons will fail to reach the scout before they fire their shot.
Projectile Energy Type	Entropy Lance
Strategy	This enemy enforces the use of absorbing energy at a specific time. Absorbing at the right time will absorb the energy from the rifle shot and the player will take no damage. They will then be able to use the entropy lance to dispatch the enemy from afar.

Name	The Jackhammer
Type	Emphasiser
Description	An enemy that wields a large hammer. The Smasher is a large enemy that is high-tempered and very aggressive. Many troops in the Realm of Order fear him.
Enemy Stat	Health – High Fire-Rate – Slow Damage – Very High Range – Very low Speed – Slow
Enemy Behaviour	When the smasher spots the player, the enemy will proceed to run at the player, they'll proceed to jump and smash the ground leaving a damaging ring that proceeds to get larger over time. The ring counts as AOE damage and can even damages enemies around them.
Absorb	By absorbing the rings that smasher produces, gives players energy as well as access to the Entropy Hammer.
Release (Entropy Weapon weaknesses)	This enemy can be taken down using any entropy weapon
Projectile Energy Type	Entropy Hammer – AOE rings
Strategy	The Smasher emphasises movement and well-timed absorbing. Players will need to jump when the smasher strikes and strike when the smasher jumps.

Name	The Splitter
Type	Emphasiser
Description	A slender enemy that wields one chakram in each hand. Can split into two copies of itself. The right copy has a purple appearance while the left one has an orange appearance.
Enemy Stat	Health – Low Fire-Rate – Average Damage – Average Range – High Speed – Average
Enemy Behaviour	When the splitter spots the player, the splitter will split into two parts. Each part will fire two different projectiles. Hitting each part within the same time will cause them to re-join.
Absorb	Absorbing either projectile will give the player that respective entropy weapon as well as some energy.
Release (Entropy Weapon weaknesses)	Each part is susceptible to any entropy weapon.
Projectile Energy Type	Entropy Light Chakram – Chain attacks – each chain refills energy Entropy Dark Chakram – Chain attacks – each chain does damage
Strategy	The Splitter emphasizes use of tactics. If the player hits both enemies at the same time, they will re-join. This will cause this enemy to become stunned for a moment giving the player a window to dish out the pain. Players can also use any entropy weapon that chains or is AOE to make them join too.

Name	The Support
Type	Enforcer
Description	A small enemy that floats around groups of enemies. This enemy wields a wand that shoots a healing aura.
Enemy Stat	Health – Low Fire-Rate – Fast Damage – None Range – Average Speed – Fast
Enemy Behaviour	The support will fly to any enemy that is low on health. When the support reaches them, they will proceed to heal the enemy over time. If the support manages to heal them completely then that enemy will get a buffed fire rate and damage making them harder to defeat.
Absorb	This enemy provides the player with nothing to absorb.
Release (Entropy Weapon weaknesses)	Is susceptible to any entropy weapon.
Projectile Energy Type	Does not fire projectiles.
Strategy	Though this enemy seems harmless, its supportive capabilities make it something that the player must immediately focus on. I've listed it as an Enforcer as it demands the players attention.

Hazards

In this section, we will be going into detail about hazards in Disorderly Combat. Hazards will also have stats like enemies, however not all enemies fire projectiles so for some hazards we will only be tracking Health/Energy and Damage. For static hazards that produce projectiles, it will closely mimic the structure of Enemy Design.

Name	Pristine Spikes
Type	Static Hazard
Description	Spikes that sap energy from the player. When the player touches sap spikes, a set amount of energy will be drained.
Stats	Energy/Health – High Damage – Low Rate of Damage - Low Range – Average
Absorb	Players can absorb energy from the sap spikes. Like some obstacles it also has two states. When Filled – On contact, the player will take a lose set amount of energy. When Hollow – Spikes will have no effect.
Release	Once the spikes are absorbed, they will become hollow and disappear. The player will gain a new entropy weapon.
Energy Type	Entropy Caltrops

Name	Leeching Zone
Type	Static Hazard
Description	A burning area. When the player enters the area, the player will lose energy at an increased rate.
Stats	Energy/Health – Average Damage – Very Low Rate of Damage – Very High Range – Average
Absorb	Players can absorb energy from the burning area. Below are the two states: When Filled – On contact, the player lose energy over time. When Hollow – Burning area will have no effect.
Release	Once the area is absorbed, they will become hollow and disappear. The player will gain a new entropy weapon
Energy Type	Entropy Sap

Name	Pristine Mine
Type	Static Hazard
Description	A condensed area of volatile ordered energy. When the player collides with its impact radius, it will detonate and instantly deplete all their energy
Stats	Energy/Health – Average Damage – Very High Range – Average
Absorb	Players can absorb energy from the mines. Like some obstacles it also has two states. When Filled – On contact, the player will lose all energy. When Hollow – Mine will have no effect.
Release	Once the mine is absorbed, they will become hollow and disappear. The player will gain a new entropy weapon
Energy Type	Entropy Mine

Name	Pristine Turret
Type	Static Hazard
Description	A stationary machina that analyses its surroundings for intruders.
Stats	Health – Low Damage – Average Fire-Rate – Average Range – High Speed – Average
Absorb	Players can absorb energy from projectiles this hazard fires. Absorbing the projectile refill, the player's energy gauge and will give the player an entropy weapon.
Release	Once the mine is absorbed, they will become hollow and disappear. The player will gain a new entropy weapon
Energy Type	Entropy Missile

Entropy Weapons

Considered the second half of the Entropy System, Entropy weapons are powerful tools the player will gain any time they absorb energy. The player can gain these weapons by using the Entropy System in interesting ways.

Most Entropy Weapons can be obtained by absorbing projectiles fired by enemies. You can't have multiple Entropy Weapons loaded at the same time. For example, if the player has the Entropy Twin-Blade, then gained the Entropy Fan, the Twin-Blade will be discarded. Entropy Weapons also have limited uses. Once you use up all the uses, the weapon will be discarded, and you will be re-equipped with the Entropy Dagger. Players will need to be wary as Entropy Weapons have an associated energy cost that's determined based on their power. The more powerful the weapon, the more energy it uses up, potentially putting the player at risk.

This section will detail every Entropy Weapon that has been designed. There will be 6 categories, the weapons name, the type, description, stats, behaviour and uses. I will finalise the stats and uses in a numerical design document. The key here with weapon design, is each weapon needs to have a distinct feel and use. No two weapons should feel the same. This will encourage players to use the full arsenal of "Entropy Weapons".

Name	Entropy Dagger
Targeting Type	Single Target
Description	<i>A quick and reliable throwing dagger. Furi's favourite weapon.</i>
Stats	Damage – Low Speed – Very Fast Range – High Rate of Fire – Very High
Behaviour	Can be used to damage enemies. Can also be used to fill up objects with energy. Dagger will move in straight line and dissipate once it goes out of range.
Uses	Infinite

Name	Entropy Mine
Targeting Type	Area of Effect
Description	<i>A powerful explosive force condensed into a volatile orange orb. A demonologist dream weapon.</i>
Stats	Damage – Average Speed – Fast Range – Average Rate of Fire – Average
Behaviour	Player will fire a bomb onto a ground, after a short amount of time, the player will get launched into the air based on their proximity to the bomb. Enemies around the bomb will also take damage.
Uses	Single

Name	Entropy Sap
Targeting Type	Area of Effect
Description	<i>A weapon that drains the life of those in the vicinity. Original was developed to serve as a healing item, but tests proved its lethal capabilities</i>
Stats	Damage – Very Low Speed – Very Fast Range – Average Rate of Fire – Low
Behaviour	Creates a small zone on the ground. Enemies that enter the zone, will have their energy drained. All energy drained will be provided to the player.
Uses	Single

Name	Entropy Caltrops
Targeting Type	Area of Effect
Description	<i>A bundle of sharp tac's loaded into an orb. Lucky for user's, the caltrops come pre-loaded.</i>
Stats	Damage – Very Low Speed – Fast Range – Average Rate of Fire – Low
Behaviour	Creates a small zone on the ground. Enemy's that collide with the zone will be slowed down and more susceptible to damage.
Uses	Single

Name	Entropy Fan
Targeting Type	Single Target
Description	<i>An elegant fan with sharp blades hidden within it. Leaves behind a field that boosts agility.</i>
Stats	Damage – Average Speed – Fast Range – High Rate of Fire – Average
Behaviour	Projectile - When fired it will launch a fan in a straight line. The fan stops once it hits a single target. Once the fan stops it will leave a short area, this area boosts the move speed and jump height of the player.
Uses	Limited – 2 uses

Name	Entropy Twin-Blade
Targeting Type	Single Target
Description	<i>Two curved swords attached together with a handle. Sometimes, creativity solves all problems.</i>
Stats	Damage – High Speed – Fast Range – Average Rate of Fire – Low
Behaviour	Projectile - When fired it will travel forward in a straight line before traveling back. Player can reabsorb the twin-blade once it travels back.
Uses	Limited – 1 use

Name	Entropy Blade
Targeting Type	Single Target
Description	<i>A long Blade infused with chaotic energy. Sometimes, efficiency is all that's needed.</i>
Stats	Damage – High Speed – Very Fast Range – Low Rate of Fire – Low
Behaviour	Melee – Player will travel a dash a short distance forward. Player is impervious to damage during this dash. Anything hit during the dash will take damage.
Uses	Limited – 1 use

Name	Entropy Axe
Targeting Type	Area of Effect
Description	<i>A large dominating axe strong enough to destroy objects. Despite its size, its surprisingly easy to launch.</i>
Stats	Damage – Very High Speed – Slow Range – Average Rate of Fire – Very Low
Behaviour	Projectile – When launched, axe will travel in an arc upwards before landing. Once it lands, will create a damaging AOE sphere. Can also blow hole through objects.
Uses	Limited– 2 uses

Name	Entropy Lance
Targeting Type	Area of Effect - Line
Description	<i>A long pole with a sharp edge at the end. Sharp enough to pierce a hole through multiple things at once.</i>
Stats	Damage – High Speed – Fast Range – Very High Rate of Fire – Average
Behaviour	Projectile – When launched, lance will travel forward, once it hits an enemy, that enemy will take critical damage. Lance will punch through enemies.
Uses	Limited – 2 uses

Name	Entropy Hammer
Targeting Type	Area of Effect - Sphere
Description	<i>A gargantuan rock attached to a long pole. Written on the rock “property of Grom”</i>
Stats	Damage – Very High Speed – Very Slow Range – Low Rate of Fire – Low
Behaviour	Melee – When activated, weapon will charge up before colliding onto the ground. Once it hits the ground, all enemies will be launched into the air taking heavy damage and will be suspended there for 3 seconds.
Uses	Limited – 1 use

Name	Entropy Chakram (Light + Dark)
Targeting Type	Single Target
Description	<i>Two large disks that shift colour often. Its behaviour shifts depending on its appearance. Slaying meets style.</i>
Stats	Damage – Low (Light), Average (Dark) Speed – Very Fast (Light), Fast (Dark) Range – Very High (Light), High (Dark) Rate of Fire – High (Light), Average (Dark)
Behaviour	When fired, chakrams will home in on the nearest target before chaining to the next one. Can chain with a maximum of 4 enemies. Each chakram has a different effect. Light – Player gains energy for each enemy the chakram hits. Dark – Each enemy loses increased damage per enemy hit.
Uses	Limited – 1 use

Numeric Design

With everything we have designed, the next logical step will be to put it all into an excel spreadsheet and do some numeric calculations to generate some values. This process is often called Numerical Design. Numerical Design is an important step in designing gameplay systems because it puts everything we've designed into perspective. Helpful values such as the players max energy and damage for all enemies and Entropy Weapons can be calculated using Numeric Design. It's also a helpful balancing tool, because it can help us set appropriate values for attributes such as energy lost by moving. If I want the game to be difficult, I will use Numerical Design to find an average value for energy lost and bump it up. Attached to this document will be the Disorderly Combat Numeric Design Spreadsheet. This document will be annotated and clear to read so you can understand my thought process behind the numbers.

Name	Type	Individual Damage	Individual Speed	Individual Range	Individual Rate of Fire
Entropy Dagger	Single-Target	2	8	10	10
Entropy Mine	AOE	2	3	7	6
Entropy Sap	AOE	1	10	7	4
Entropy Caltrops	AOE	1	6	7	4
Entropy Fan	Single-Target	3	8	12	5
Entropy Twin-Blade	Single-Target	3	9	8	4
Entropy Blade	Single-Target	5	12	5	4
Entropy Axe	AOE	8	2	7	3
Entropy Lance	Single-Target	4	7	15	6
Entropy Hammer	AOE	10	1	6	3
Entropy Chakram (Light)	Single-Target	3	8	14	8
Entropy Chakram (Dark)	Single-Target	3	6	12	5
Entropy Dual Dagger	Single-Target	3	8	10	10

Name	Type	Final Damage	Final Speed (M/S)	Final Range(M)	Final Rate of Fire (S)
Entropy Dagger	Single-Target	100	40	80	0.5
Entropy Mine	AOE	75	12	42	0.9
Entropy Sap	AOE	12.5	50	42	0.8
Entropy Caltrops	AOE	12.5	24	42	0.8
Entropy Fan	Single-Target	225	32	96	0.75
Entropy Twin-Blade	Single-Target	300	36	48	0.8
Entropy Blade	Single-Target	500	60	20	0.8
Entropy Axe	AOE	500	4	42	0.75
Entropy Lance	Single-Target	400	28	150	0.9
Entropy Hammer	AOE	625	1	24	0.6
Entropy Chakram (Light)	Single-Target	150	40	140	0.8
Entropy Chakram (Dark)	Single-Target	225	24	96	0.75
Entropy Dual Dagger	Single-Target	225	32	80	0.5

Figure 1: Snapshot of numeric design

Week 2 – Retrospective

So, there's a lot we've established this week. The purpose of this week was to design scenarios where the player can use the Entropy System in an advanced way. We've gone into more detail about the obstacles you will find in the game. We've come up with some hazards and enemies. The design for the enemies falls into a design structure that I use when designing enemies for intense hack and slash games. Enemies should be designed in a way where it tests the players offensive and defensive skills. Interesting enemy design can also encourage player behaviour such as playing tactfully, prioritising problem enemies.

Another important aspect of this week is the numbers (see the numeric design sheet). This is because it has helped me attach numbers to basic concepts to our game. Numbers allow use to infer the rest of the game design, based on the player's values and stats. It also shows if any design so far is flawed. Catching flawed design now is very useful as it saves time down the road. We also have the most interesting part for me when it comes to combat systems and that's weapons or "Entropy Weapons". I chose melee weapons for projectiles because I want the appearance of a melee combat system but the mechanical feel of a movement shooter.

Question Time! *"Can I make an engaging combat system within a first-person shooter?"*.

Having gone deeper into the design of the Entropy System, I can answer this question with a yes. Having a fun combat system isn't enough if you don't have objects and enemies for players to interact with. Good enemy design can push a combat system to its limit, and that's what I want with Disorderly Combat. Another aspect of engaging combat systems is providing the player with an arsenal of distinct weapons that each serve a different gameplay purpose. With numerical design, I've put across how each enemy and weapon will have distinctly different stats and behavioural patterns.

For next week, we will be going into the "Discord" mechanic, which will serve as the underlying mechanic to hit the 3rd bullet point of the brief. We will also go over Level Design, detailing the structure, goals, and pitfalls of levels in Disorderly Combat. The last thing I will be detailing is the project backlog, this will be useful to create structure once we start prototyping.

Week 3 – Extended Design & Prototype Planning

Welcome to week 3 and this week we will be starting strong with the further design of the 'Discord' mechanic. From there we will take everything and start to develop our level for the showcase. At the end we will create a project backlog that will serve as the main plan for the prototype.

Discord

When I design games, I like to create underlying systems that would supplement main-mechanic use. These systems should inherently rate the players performance with the mechanic. Playing well should be something we should reward the player on. What we reward the player with changes from game to game, but the general idea is, reward the player for playing well. Discord uses this design philosophy as a basis.

The Discord mechanic will be an underlying mechanic that is based on the players performance using the "Entropy System". The general idea is that by using the "Entropy System" you will begin to build up "Discord". By building enough discord this will put the player into different "Discord States". Each state will provide the player with benefits. These benefits will serve to boost gameplay enjoyment and make the game more hectic. Let's discuss how to build discord.

Building Discord

Discord will have its own meter like the player's energy meter, except this meter only fills when using the "Entropy System". Players will be able to build discord meter by doing certain actions related to the "Entropy Systems. Actions include:

- Absorbing energy from enemy projectiles.
- Dealing damage to enemies using entropy weapons.

The discord meter will also drain over time, but by completing enough actions above, the player will be able to balance the discord meter, however not all actions provide the same discord build-up. To incentivise aggressive play and prevent passive play, players will build more discord by releasing entropy weapons. There will also be different modifying attributes such as building up increased discord when firing weapons in the air. Building discord will be challenging, this is because using entropy weapons drains energy fast, so if players aren't careful, this will lead to easy game overs. Due to this we want the rewards to be strong enough, so players aim to build discord while not being overpowered. This leads us to "Discord States".

Discord States

Once the discord meter is filled, this will put the player into a state of discord, providing the player with gameplay boosts for a short time. States will be split into 3 different levels of states, "minimal", "Optimal!" and "MAXIMUM!!!", each providing better rewards than the last. Attributes that will be affected once if a discord state include:

- Max Energy: Entering the discord state will provide the player with a temporary boost to max energy, this will allow the player to fire more projectiles before needing to search for energy.
- Player Speed: Entering the discord state will speed up the players movement speed.

By playing aggressively, we will reward the player by, allowing them to be more aggressive. This will provide the player with a significant boost to gameplay speed. Since these states are temporary it will ensure that players continue to play aggressively to reach an optimal state. It will also entice players to try get a rhythm or flow going during gameplay.

Mentioned prior there will be three states, "minimal", "Optimal!" and "MAXIMUM!!!". Each of these states require the player to fill a percentage of the discord meter. "Minimal" will require at least 20% fill rate, "Optimal!" will require 50% fill rate, while "MAXIMUM!!!" will require 100%. These words will appear above the discord meter once you fill up to the segment needed. Having words appear with satisfying sound cues will provide the player with visual feedback that they're doing well. There will also be a special VFX effect that will

cover the edges of the screen to signify discord, with this effect intensifying per threshold hit. Once the player hits a segment of the discord meter, they will not lose that state once it begins to decay. An example is once the player reaches “MAXIMUM!!!”, they will stay in that state until discord hits 0%.

Discord States Breakdown

Below is a short breakdown of the different values effected by the different states placed into a neat table.

State Name:	Minimal	Optimal!	MAXIMUM
Requirements:	Discord Meter: 20%	Discord Meter: 50%	Discord Meter: 100%
Effects	Max Energy: 115% Player Speed: 120%	Max Energy: 125% Player Speed: 140%	Max Energy: 150% Player Speed: 160%
Discord Meter Decay:	100%	120%	150%

Level Design

The next core discussion point for Disorderly Combat is the design for its levels. When designing levels, I like to define 4 aspects which are the most important. These being, **Goals**, **Structure**, **Scale** and **Speed**. **Goals** are the most important aspect; it entails what the main objectives of the level are. Levels could be as simple reaching the end of the level or you would need to defeat every single enemy in the level before the end opens. Goals regulate play. **Structure** describes the layout of the level. We could go for a linear structure with one defined path, or levels could open in a non-linear way, allowing players to experiment with how they want to navigate the level. **Structure** also describes the way that levels have arranged obstacles and enemies. **Scale** describes the size of the level. With **scale**, the bigger the level is, the longer the level would take. One thing that we can also discuss with scale is verticality. Levels that have verticality, are generally more appealing since they help add some variety to levels. Finally, there’s **speed**. **Speed** describes the pace at which the level flows. For games that have an emphasis on speed-running, the pace is break-neck.

For this brief I will need to make a single level to showcase the “Entropy System”, so for the next sections I will go over goals, structure, scale, and speed.

Level Goal

In Disorderly Combat, the player will be tasked clearing out enemy arenas in a level. Once the player clears out each arena in a level, a special portal will materialise at the end of the level. Entering this portal will count as a MISSION CLEAR and that level will end. When the level ends the player will be assessed on their performance in that level. The three assessment criteria will be:

- Points – Defeating enemies will provide the player with points. Taking damage will cause the player to lose points. Higher points = Higher ranking.
- Time – The time it takes for the player to reach to get a MISSION CLEAR. Lower time = Higher ranking.
- Weapons Used – This is a count for different entropy weapons used. Using many weapons = Higher ranking.
- Discord Rate – Measures the average fill rate of the Discord meter. Staying in a state of discord will give the player a higher discord rate. Higher rate = Higher ranking.

These assessment criteria can be seen as level sub-goals or optional goals as getting low ranks do not impede player progression. With these criteria in place, it will give levels a sense of replayability as players would want to try repeat levels to get better ranks.

Level Structure

Levels will be formed of **paths** and **arenas**. Arenas and paths will challenge the player in different ways.

Arenas are the focal points of each map. Arenas are locations in a level that are filled with enemies that the player needs to clear. When the player enters the arena, a field will appear, and block all exits out of the arena. Once the player clears the arena, they will be able to leave. Once all arenas have been cleared then they can complete the level. Arenas will primarily challenge the player through combat as enemies will pose the highest threat. Arenas will also feature some obstacles to provide the player some cover if they need it.

Paths can be seen as routes to the arena. There will be multiple different routes to one arena, it will be the player's choice if they want to follow the paths or instead make their own path. Paths will serve to challenge the player through traversal. Players will need to use their movement abilities and entropy weapons to overcome these movement challenges. There will be enemies will be sprinkled liberally along paths to offer a variety of challenge.

Level Scale & Speed

Defining the scale of levels allows us to determine how big and how long maps should take in Disorderly Combat. Levels should be mid-sized and should roughly be completed in 6 minutes. Below are breakdowns for the amount and the time taken to complete arenas and paths.

Name	Arenas
Amount per Level	2 to 3
Time taken	1 minute to 2 minutes

Name	Paths
Amount per Level	5 to 6
Time taken	20 to 30 seconds

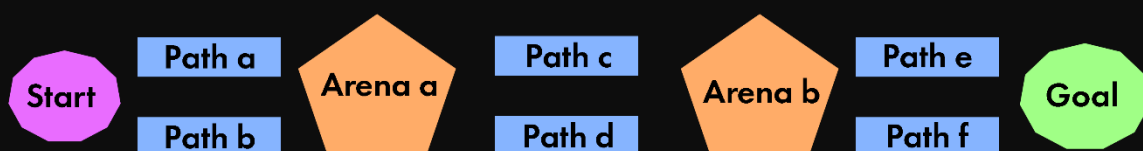
Maximum Time to complete level with 3 arenas and 5 paths = 8 Minutes 30 Seconds

Minimum Time to complete level with 3 arenas and 5 paths = 4 Minutes 30 Seconds

Time taken to complete a level will be a factor that will be assessed during the level. Players who manage to complete the levels in times faster than what's considered the average time for that level, will get higher ranks. The way we will find out the average time to complete a level, is by playtesting the level. This will be something that will happen during the production of the game.

Level Flow

Below is a crude mock-up of how a level would flow.



A potential limitation is the structure for levels. Once we go into each level specifically, then we can experiment with enemy and obstacle placement, level verticality and path/arena structure. So, the next logical step is to decide what obstacles and enemies we will be prototyping for this brief. For that, we will need to create a project backlog.

Project Backlog

A project backlog is a list of all items you are going to develop for a project. For games, having a project backlog is essential because it helps you keep track of everything that needs to be implemented. It helps keep productivity high and lets you manage your time for the project. For this project, this backlog will refer to the prototype I will be showcasing for this brief. We will need to decide what enemies and obstacles will make it to the prototyping stage. Below are the ones I have chosen.

Mechanics

- Entropy System
 - Relevance: Core Mechanic needed for brief
- Core Movement
 - Relevance: Core Functionality

Other Mechanics

- Discord System
 - Relevance: Underlying mechanic needed for brief

Obstacles

- Energy Crystal
 - Relevance: Core obstacle that's needed to regain energy
- Ivory Wall
 - Relevance: Core obstacle that interacts with entropy system
- Ivory Platform
 - Relevance: Core obstacle that interacts with entropy system
- Pristine Launcher
 - Relevance: Core obstacle that interacts with entropy system

Hazards

- Pristine Spikes
- Leeching Zone

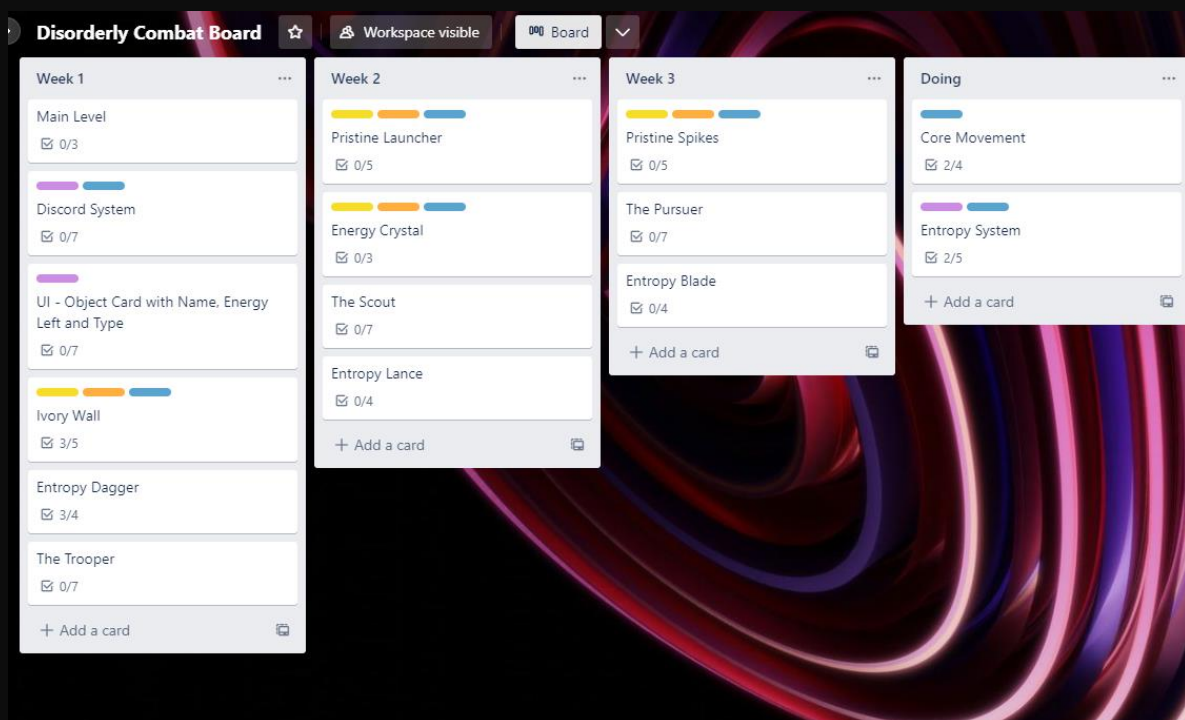
Enemies

- The trooper
 - Relevance: Average enemy
- The scout
 - Relevance: Long ranged Enemy
- The pursuer
 - Relevance: Melee Enemy
- The brute
 - Relevance: Large Enemy

Entropy Weapons

- Entropy Dagger
 - Relevance: Starting Entropy Weapon
- Entropy Lance
 - Relevance: Long-range Entropy Weapon
- Entropy Blade
 - Relevance: Short range Single Target Weapon
- Entropy Axe
 - Relevance: Mid-ranged AOE Weapon

Trello



To track tasks and keep productivity, I will be using Trello. From there I have put taken all the elements I've chosen to develop into a Trello board. From this point onward I will have four weeks to complete this project. Within this four-week time frame, I've placed all the tasks into their own respective week. The week structure follows:

1. Week 1
 - a. Main Level Design
 - b. Core Movement development
 - c. Entropy System development
 - d. Ivory Wall & Ivory Platform development
 - e. Entropy Dagger
 - f. The Trooper development
2. Week 2
 - a. Discord System development
 - b. Energy Crystal development
 - c. Pristine Launcher development
 - d. The Scout development
 - e. Entropy Lance development
3. Week 3
 - a. Pristine Spikes development
 - b. The Pursuer development
 - c. Entropy Blade development
4. Week 4
 - a. Project polishing
 - b. Itch page creation
 - c. Showcase video creation.
5. Stretch Goals
 - a. Leeching Zone development
 - b. The Brute development
 - c. Entropy Axe development

Week 3 Retrospective

With week 3 over, the pre-production & main design of the game is finished. I've focused on designing an underlying system that will hit the 3rd assessment criteria for the brief. We still need to design the actual level for the showcase but the design for the core mechanic is finalised. This is good because it means we can focus on prototyping the mechanic. It also stops us from over-designing the mechanic. As a designer, I know we can get stuck on designing mechanic it, and for a project like this where we have a set-in stone time limit, it's better to move on from the core design, prototype it and see what we end up with. If a feature feels off such as the rate of energy drain, then we can rebalance it in the numeric document. As my lecturer says, "Don't have a head transplant".

Once again, I would like to bring up the question:

"Can I make an engaging combat system within a first-person shooter?"

With the design of the Discord underlying mechanic, I feel like I can answer this question with a "YES!". The discord mechanic encourages skilled play and rewards the player with more opportunities for skilled play. This creates a positive loop where the player will always be rewarded for playing well, thus creating an engaging system and positive feedback loops are engaging.

With the next coming weeks, this document will morph into a more retrospective type of document where I detail the trials and tribulations of developing Disorderly Combat in the Unreal Engine. It will certainly be a trial as I have zero experience with the Unreal Engine, but there's always a first time for everything.

Week 4 – Level Design and UI Design

Welcome to WEEK 4 OF DISORDERLY COMBAT WOOP! WOOP! This week will see us designing the level for the showcase as well as developing the UI for the project. MVP stands for minimal viable product and serves as a prototype for video games before they go into full stage development. MVPs usually don't have any fancy effects or models and are usually just grey box environments. MVPs should contain elements that are core to your game. Before beginning the MVP, let us delve into the further level design for the showcase.

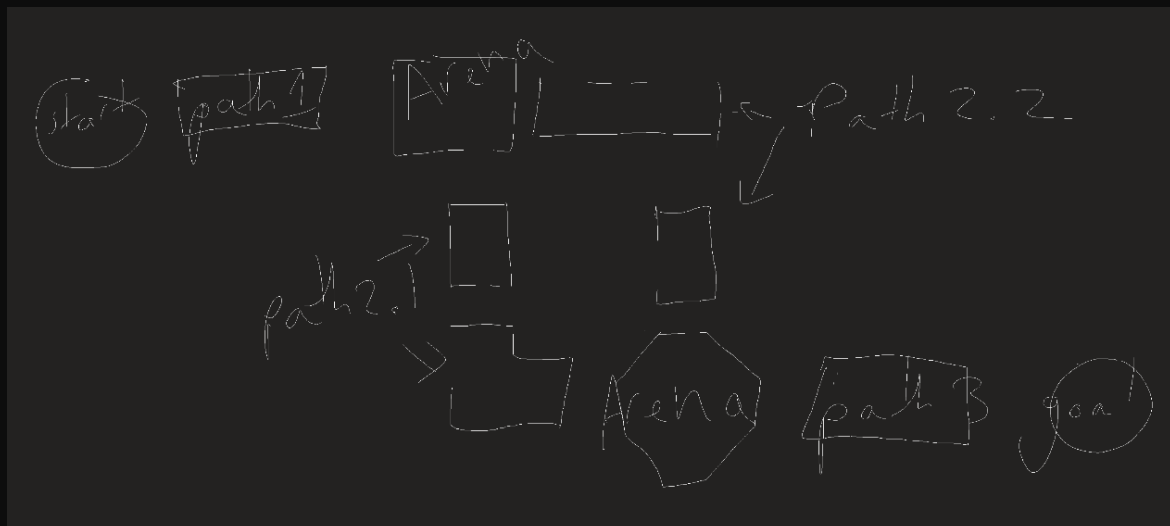
Level Design (Again?)

Now, this section of level design, will see us developing some sketches and tangible plans for the level. We will be using the goals, structure, and scale & speed we have assigned for game during week 3. To recap:

- Goal: Players need to clear all arenas in a map and proceed to the end goal.
 - Sub-goals: Gain a high rank by racking up points, finishing the level quickly, using a variety of entropy weapons and by staying in a discord state.
- Structure: Levels will be comprised arenas and paths. Arenas will be filled with enemies and will challenge the player through combat. Paths connect arenas and will challenge the player through traversal.
- Scale & Speed: Levels should contain 2 to 3 arenas and 5 to 6 paths. Levels will take 4 to 8 minutes depending on the players skills.

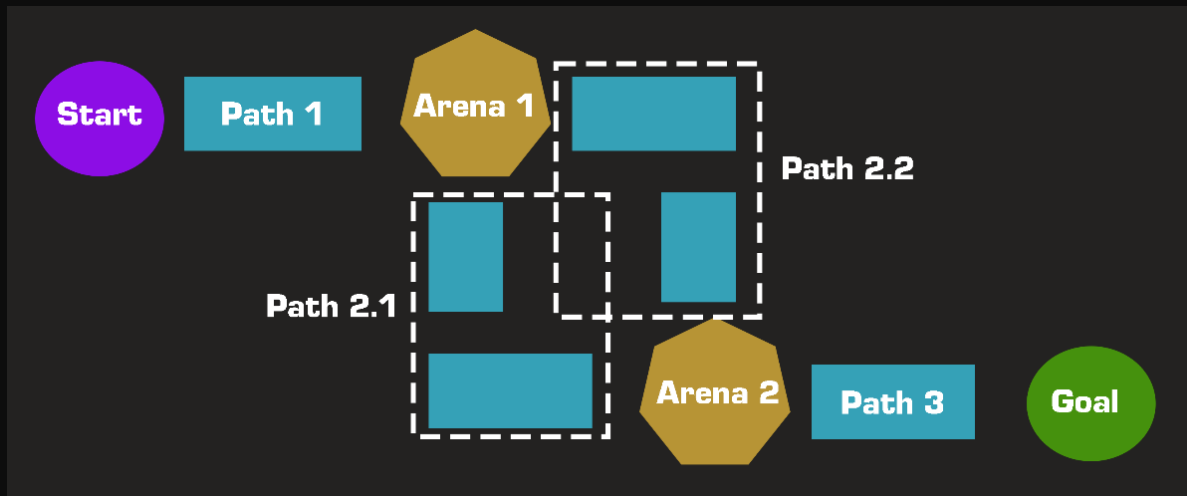
Level Rough Sketch

Below is a very rough sketch for what the showcase level will look like. The design of the level itself is always subject to change as the process continues.



Level Sketch

Taking the level sketch above, below is a clearer layout of the initial design for the showcase level.



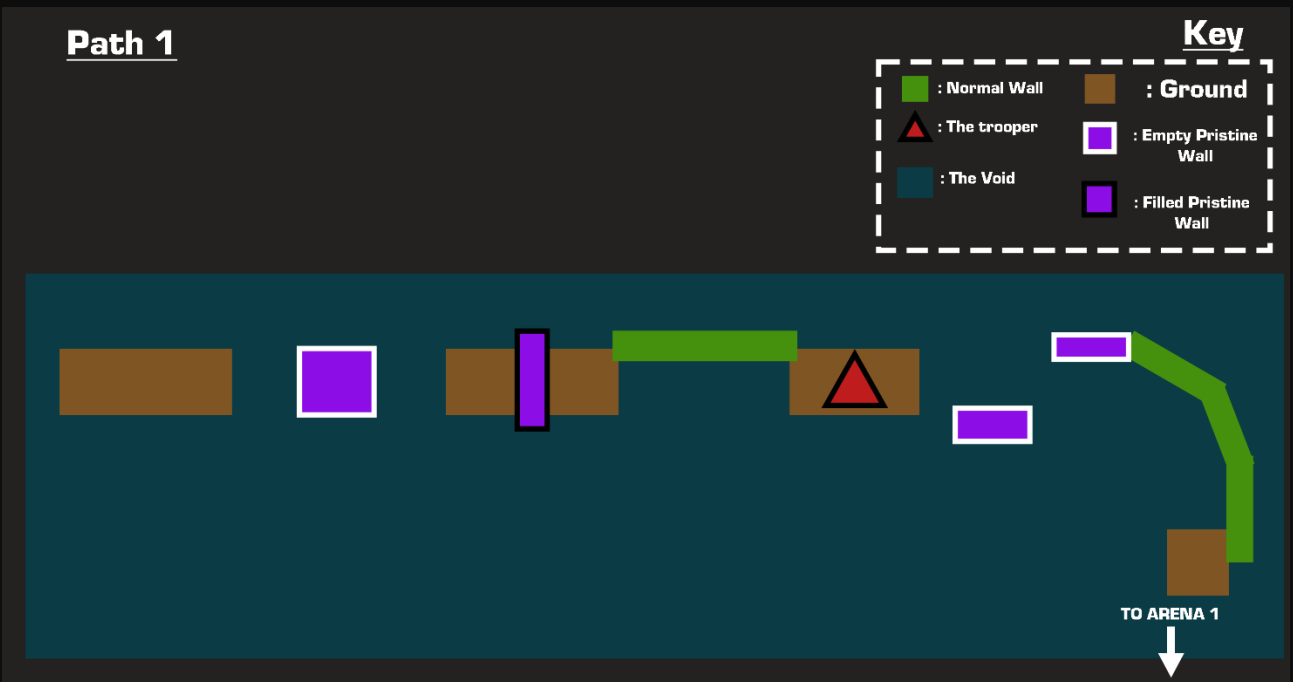
Level Structure Plan

The next step for designing this level is deciding what elements (obstacles, hazards, and enemies) I want to feature. Firstly, I need key obstacles such as Pristine Platforms/Walls/crystals that serve as the core backbone for gameplay. When it comes to enemies and hazards, I want an even variety. This is most important with enemies as I want to show the different use cases of the Entropy System when dealing with these enemies. Below is the rough structure of each area.

- Path 1
 - Obstacles: Energy Crystal, Ivory Wall, Ivory Platform
 - Enemies: The Trooper
- Arena 1
 - Obstacles: Ivory Walls
 - Enemies: The Trooper
- Path 2.1
 - Obstacles: Energy Crystal, Pristine Wall, Pristine Platform, Pristine Launcher
 - Enemies: The Scout, The Trooper
- Path 2.2
 - Obstacles: Energy Crystal, Pristine Wall, Pristine Platform, Pristine Spikes
 - Enemies: The Pursuer, The Trooper
- Arena 2
 - Obstacles: Pristine Spikes
 - Enemies: The Scout, The Pursuer, The Trooper
- Path 3
 - Obstacles: Energy Crystal, Pristine Wall, Pristine Platform, Pristine Launcher, Pristine Launcher
 - Enemies: The Pursuer

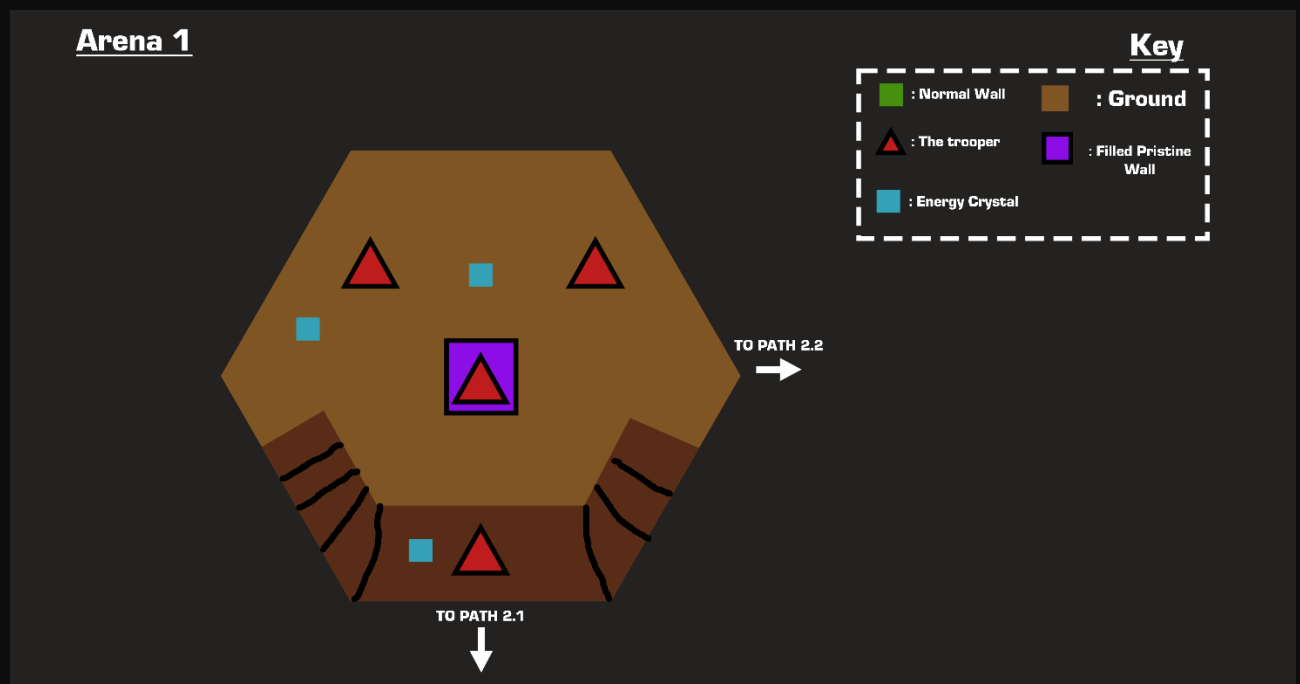
Path 1 Structure

Below is a draft of path 1. Path 1 should teach the basics of movement and the entropy system. It should be the easiest path of this level. The path will teach the player how to jump and wall run. It will also teach the player that they can absorb Pristine walls and shoot entropy weapons at them.



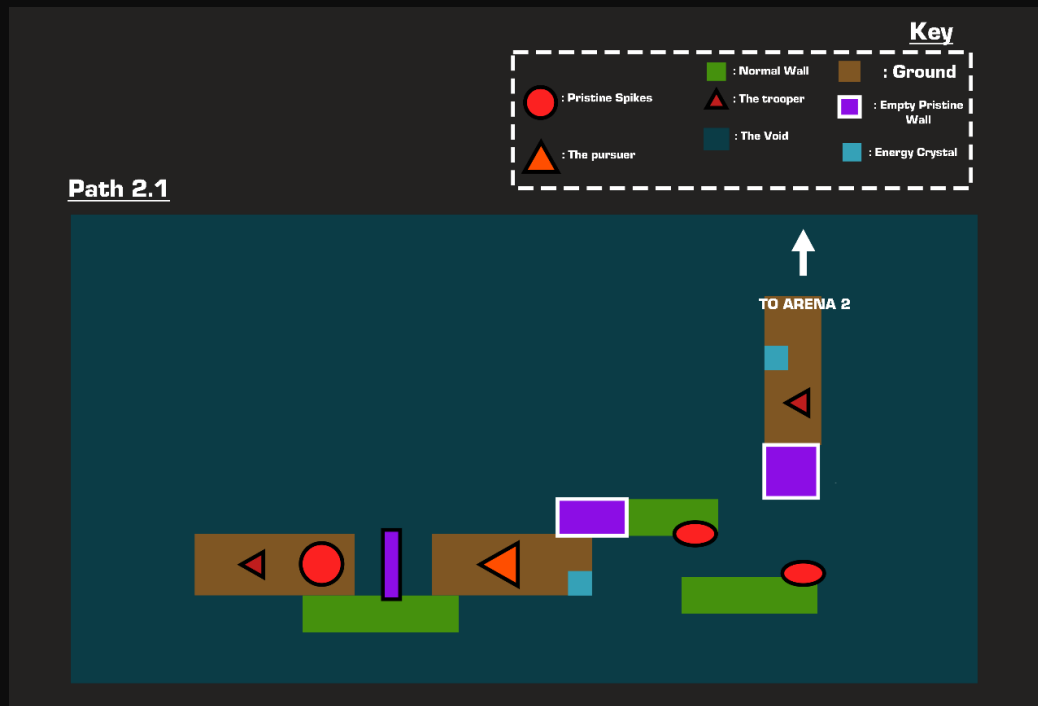
Arena 1 Structure

Below is a draft of arena 1. Arena 1 should test the player on the combat skills they have learnt on path 1. There will be three troopers on the ground and 1 trooper on an elevated staircase. This arena should serve to solidify the player's ability to absorb enemy projectiles.



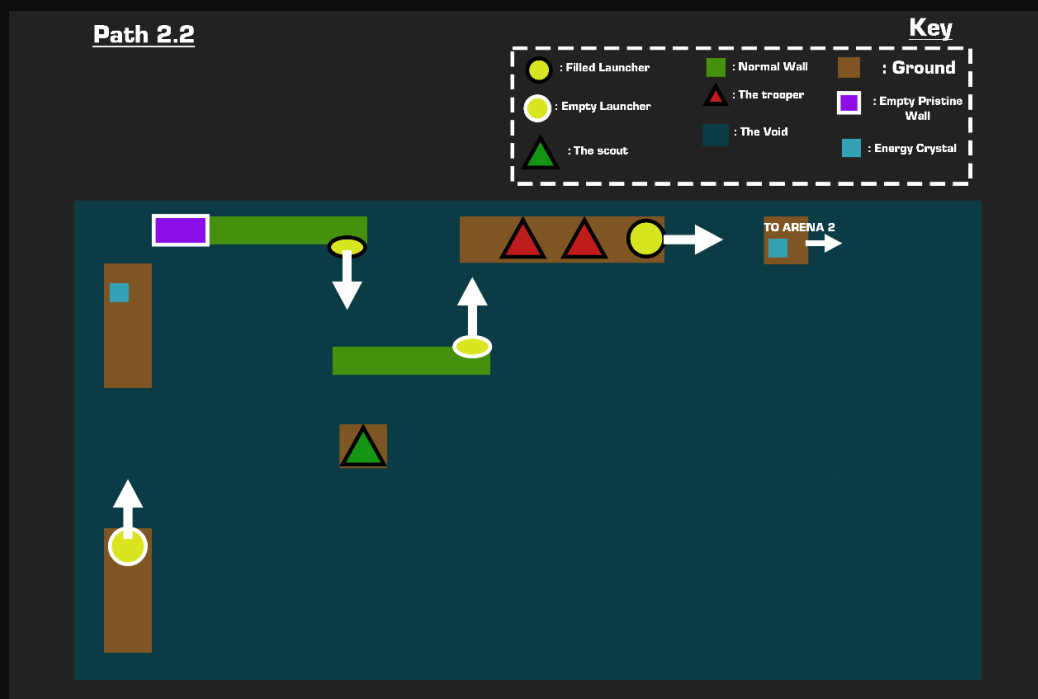
Path 2.1 Structure

Path 2.1 will introduce the Pristine Spikes and will also introduce a new enemy type, the Pursuer. This path will teach the player that spikes can be placed on the ground or on walls.



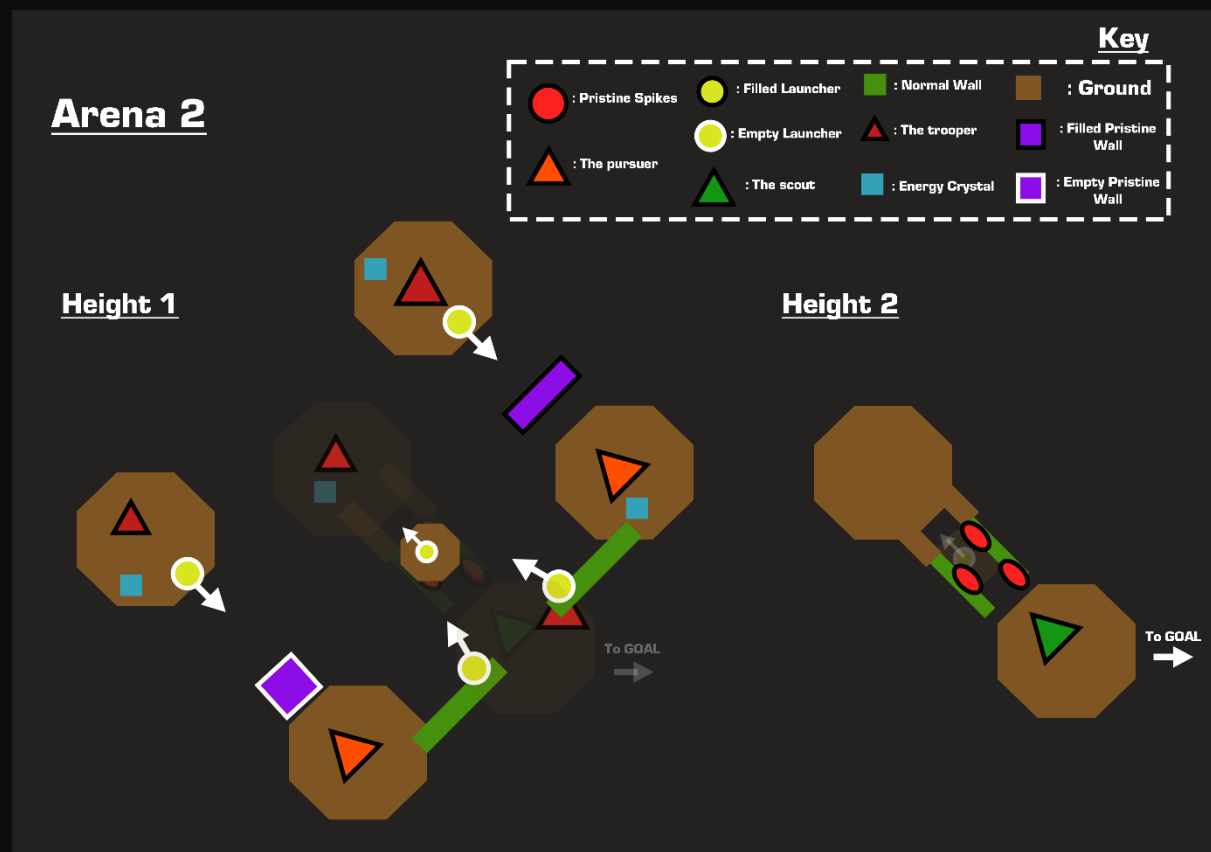
Path 2.2 Structure

Path 2.2 will introduce the Pristine Launcher and will also introduce a new enemy type, the scout. This level has the scout placed far away to the player is forced to absorb their attack.



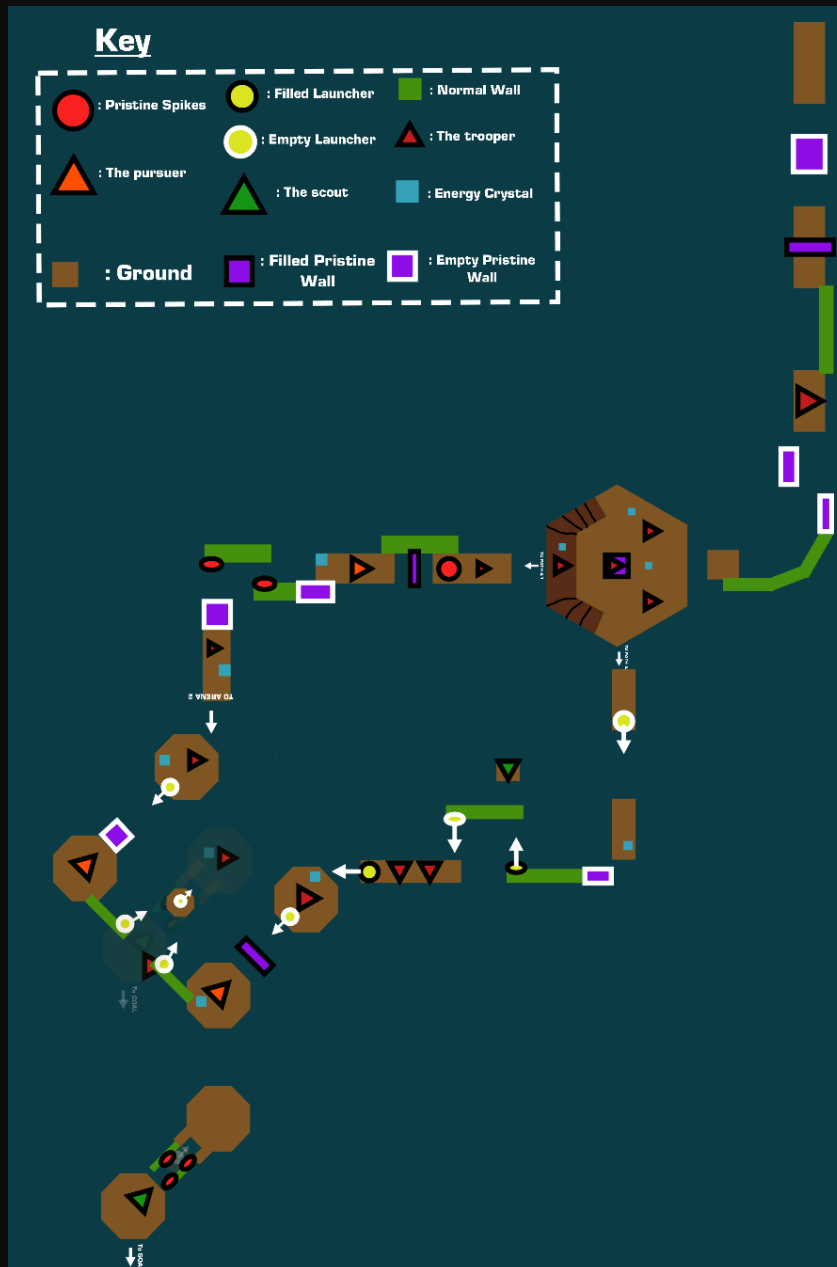
Arena 2 Structure

Arena 2 is the **focal point** of the showcase level. Arena 2 will be focused on verticality. I enjoy games that have the player ascending and descending to reach different locations. Arena 2 will have 3 levels. In the first level, the player will deal with troopers. In the second level, the player will face the pursuers and in the final level, the player will need to fight a scout. The arena itself will be separated into different islands that can be navigated to by using launchers. Since Arena 2 is the final test, this area should test everything that the player has learned so far. This area is what I call the **focal point** of the level. The focal point is the standout area of a level and usually is the most memorable aspect of a level.



Full Level Composition

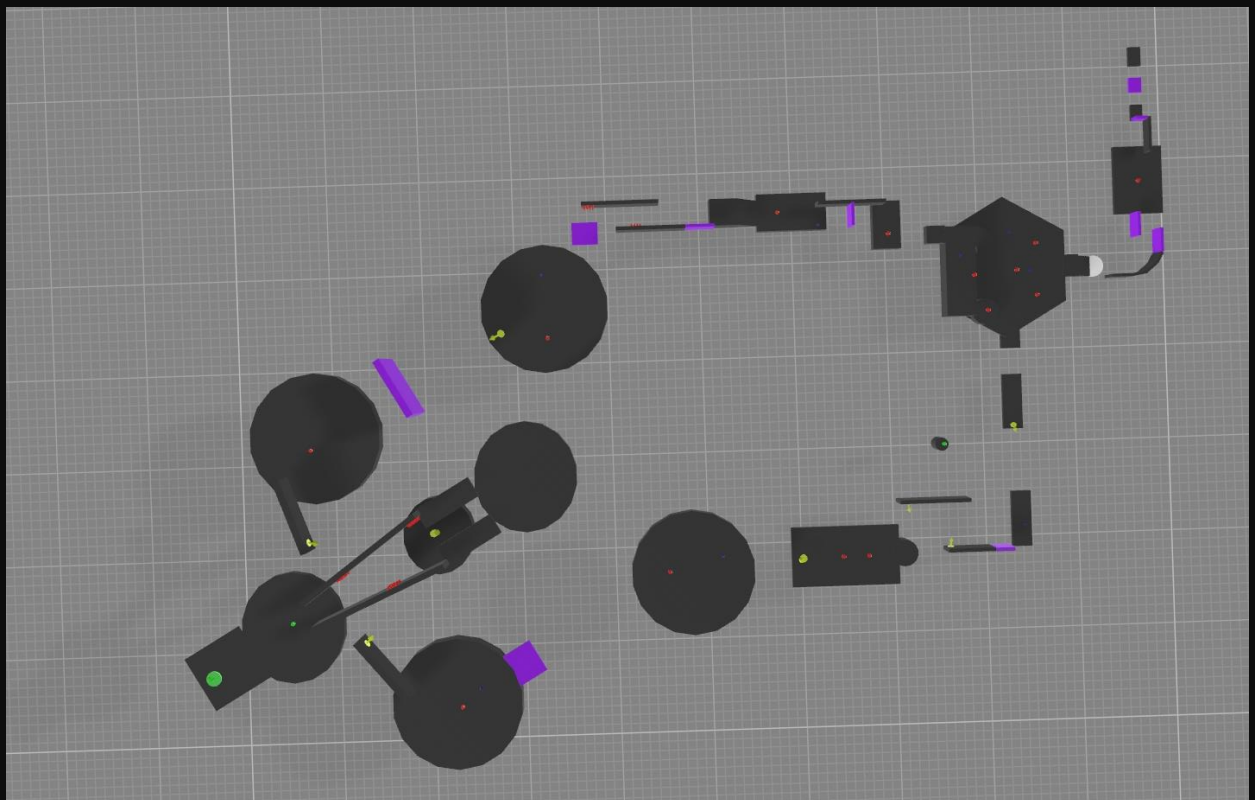
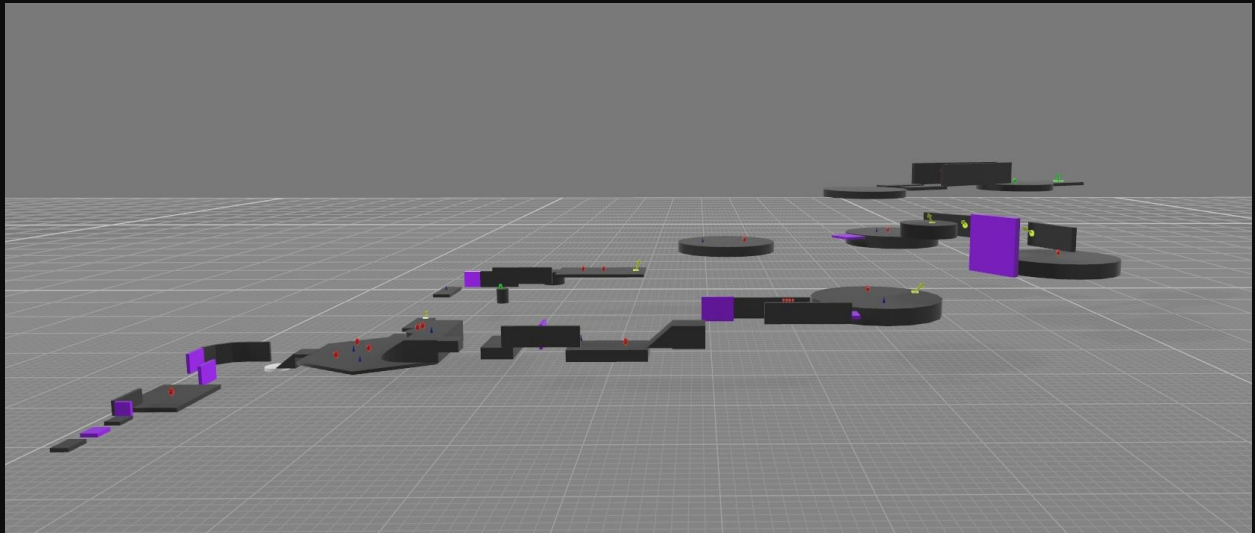
Below is the final composition of the level. I've combined the level mock-ups from each area into a single image. The overall design goal for this level was to create an engaging level that also serves as a tutorial. One of the issues that this game will have been how to teach the nuances of the game to the player. It will always be more enjoyable when a player finds out something about the game by themselves, but at the same time, some information does need to be spelled out to the player. I will touch on the idea of communication when I discuss the UI design for Disorderly Combat.



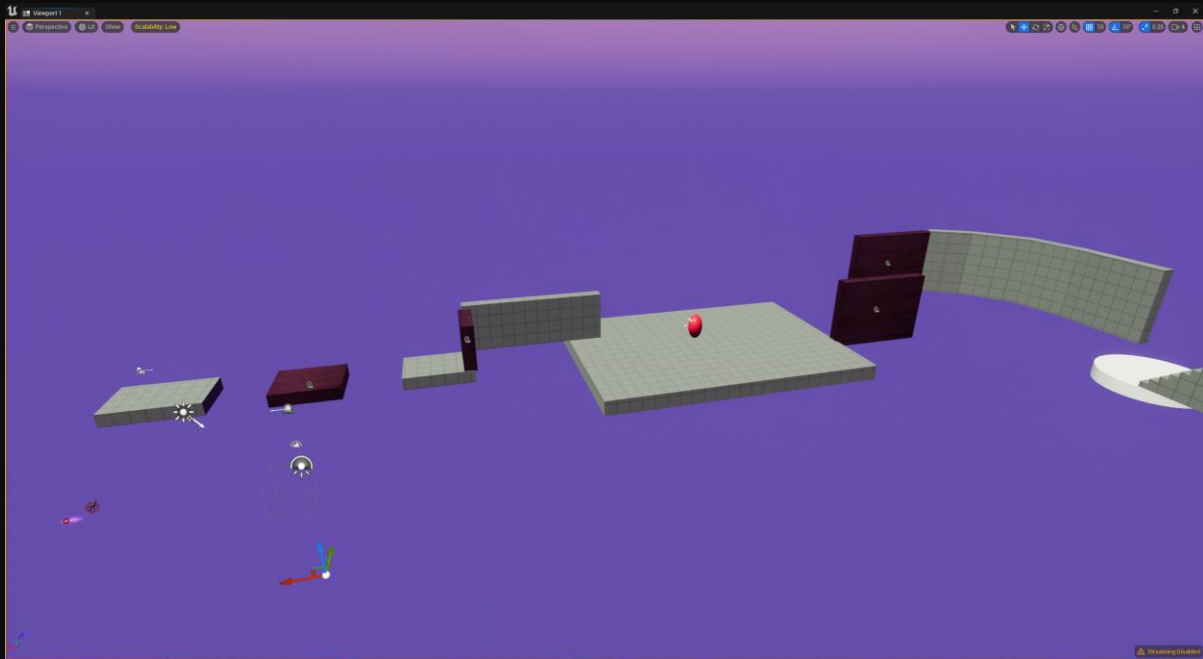
The next step regarding level design will be to grey box each area. Grey-boxing a vital tool in any level designer's toolset. It allows for rapid testing of any levels that have been designed so far. It also gives structure and helps the other areas of the development team to contribute to the game. Below I will showcase off the areas in a grey-box format.

Level Grey-boxing

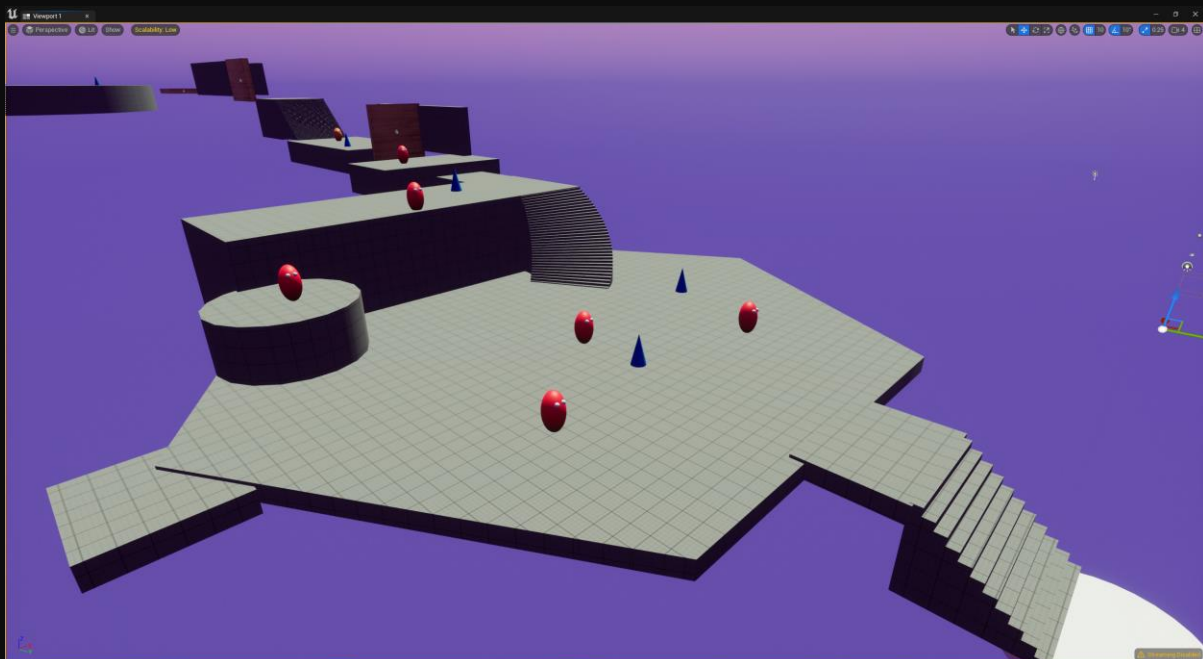
One of the immediate problems I ran into was not being able to choose what software to grey-box the level. Though I have experience with Pro-Builder in Unity, this project restricts us to Unreal Engine 5, an engine I've had no experience with. Regardless, picking up Unreal's modelling tool was not as hard as I imagined it would be, since it feels like Pro-Builder. What I did was model the entire level in Unreal 5, then proceeded to save the level as a static mesh. Then I exported the level as an FBX. This is useful as it allows me to view the entire level grey-box in Windows built-in 3D viewer. It's also incredibly useful for the reader for this document. This model will be in the main folder that contains the GDD and the Numeric Design Document.



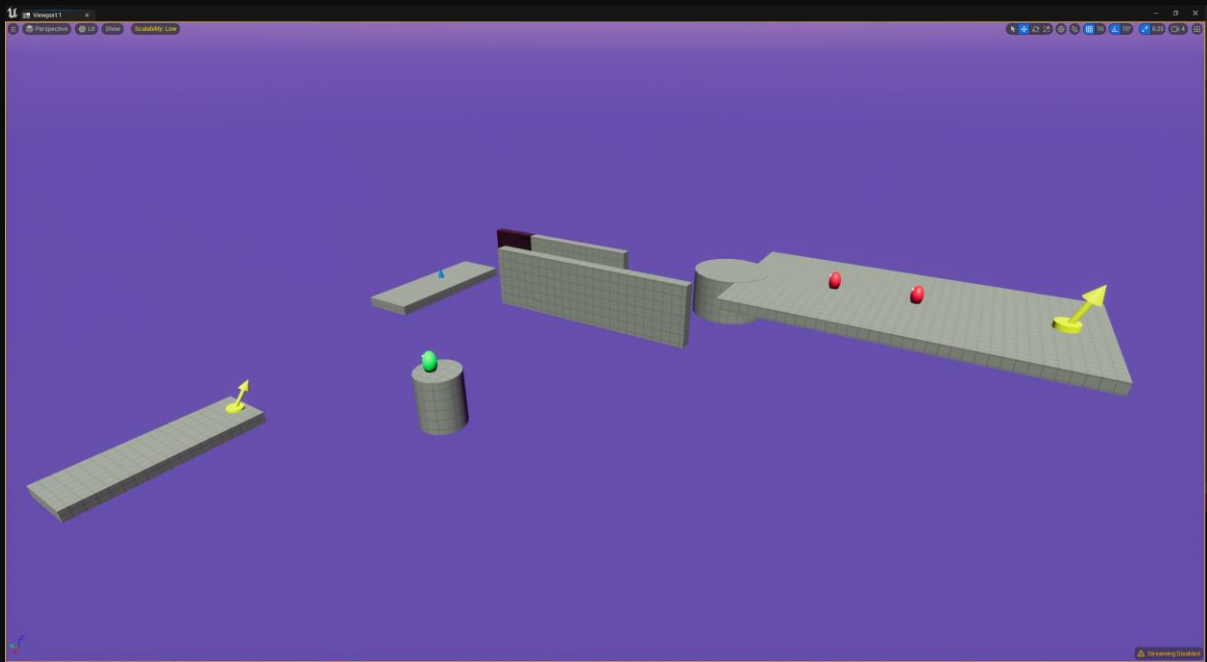
Path 1 Grey-Box



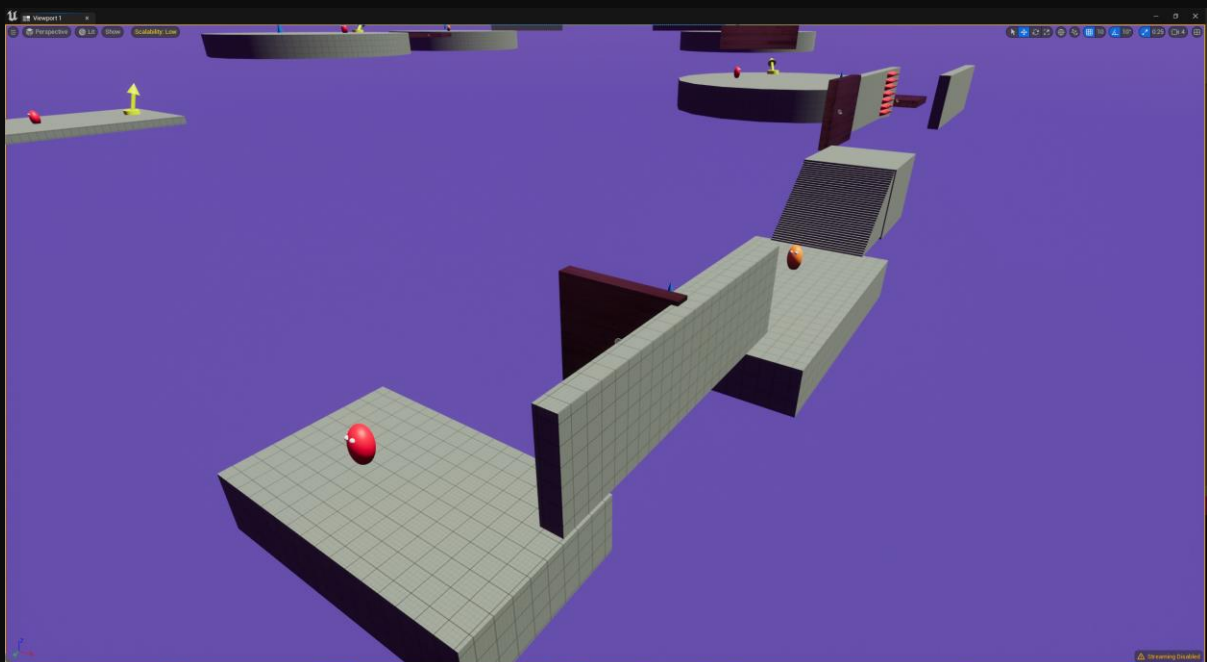
Arena 1 Grey-Box



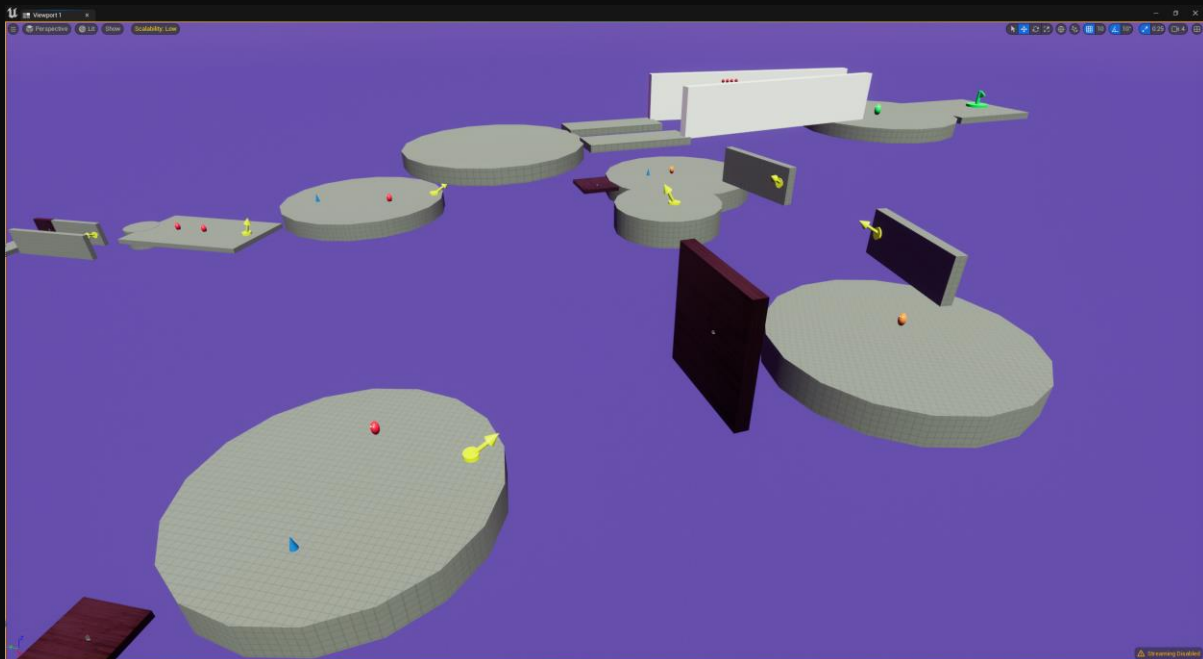
Path 2.2 Grey-Box



Path 2.1 Grey-Box



Arena 2 Grey-Box



Having completed the grey box for the first level, I'm satisfied with how well the level has translated into the 3D space. Of course, the level will still be tinkered with when adding the hi-fi models for each component. What this grey box map is supposed to achieve is an estimate on where we should place enemies, obstacles, and hazards.

The next area I will be designing and developing will be the UI for Disorderly Combat.

UI Design

UI design is an area that should not be overlooked as its important to visualise important information for the Entropy System. For shooters in particular, UI should hold all useful information that the player will need for the entirety of the game. There are lots of aspects to consider when I design UI, but there are three main aspects I like to focus on when designing UI. These are: **Purpose, Clarity & Cohesiveness**, and **Style**

- **Purpose:** This describes the purpose of each UI element you put on the screen. Each UI item placed on screen takes away vital screen space. For a fast-paced twitchy action shooter such as Disorderly Combat, only the most important of information should be shown so the UI remains clear.
- **Structure:** This describes the layout for each UI element. For any game, UI should be clear and understandable, so the player doesn't waste time.
- **Style:** This describes the visual aesthetic and art style the UI is going for. In games it's important that UI fit the theme of your game so it can bolster the appeal of the game. How you style the UI is up to the designer/artist.

Let's get down to business and describe how the UI should be designed regarding each of the aspects of UI design.

UI Purpose

This area will focus on the elements of the game that should be displayed on UI. For Disorderly Combat, only important information should be displayed. Below are the elements that should be displayed in order of importance.

1. Energy Bar (Shows value of energy)
2. Entropy Weapon Type (Shows what entropy weapon is equipped)
3. Entropy Weapon Usage Left.
4. Entropy Weapon Cost
5. Discord Values (Shows overall discord state achieved)

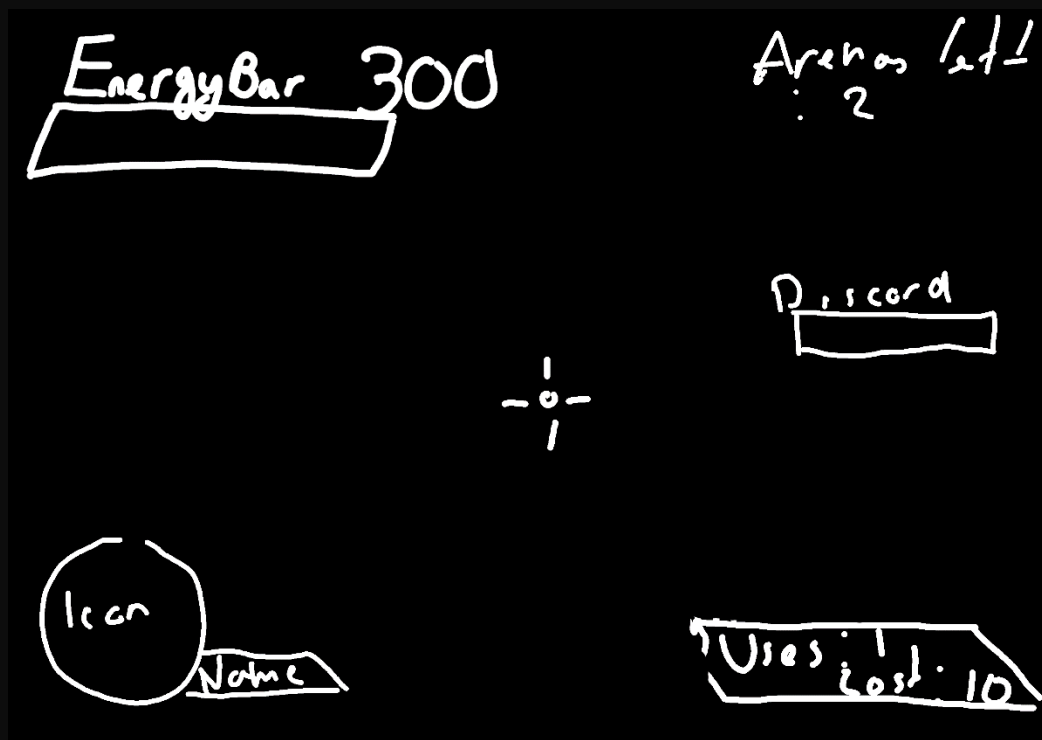
The elements above are aspects that are important to the core mechanic of the “Entropy System”. Energy values must be displayed. The player will always be checking values such as energy left, and the entropy weapon equipped so having them take the most importance makes sense. An Entropy Weapons uses left will also be another key element to display. The cost of each weapon is less important since the cost of a weapon doesn’t change, players will be able to memorise the cost, if they don’t then they can always look at the UI. Discord Values is another aspect that should be displayed as it’s an active way to rate performance. While all of these are important, we also need to imagine other information that could be useful to the player for the sake of creating a comfortable user experience. Examples can be:

- A UI element that displays time
- A UI element that displays arena’s left
- A UI element that displays the energy value of the object you are absorbing/suppling energy to.

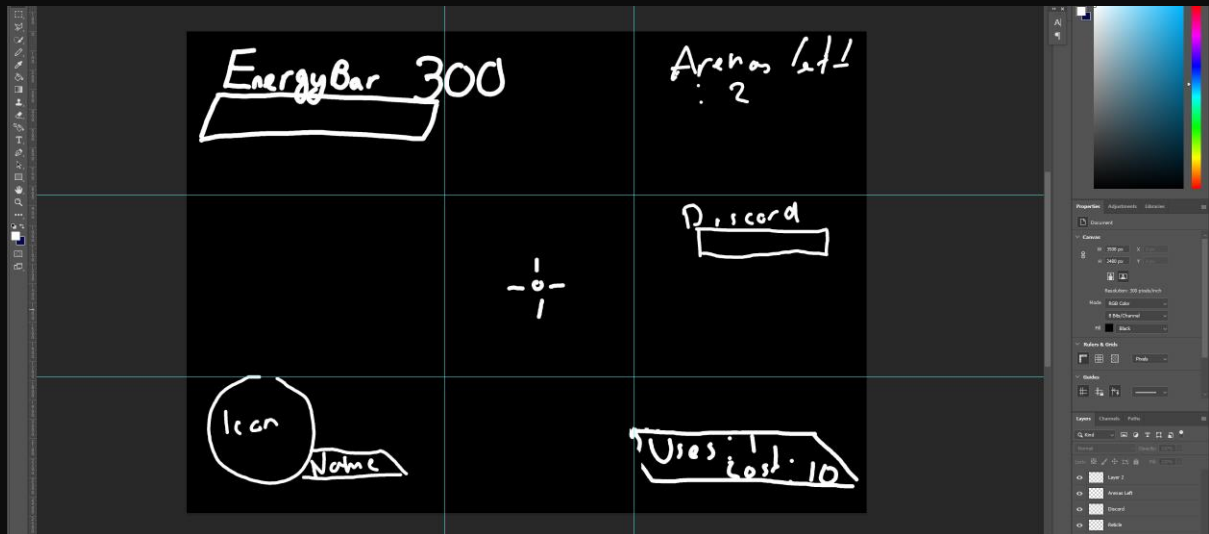
With all the potential use cases above, the next step would draft a rough layout scheme for the UI.

UI Structure

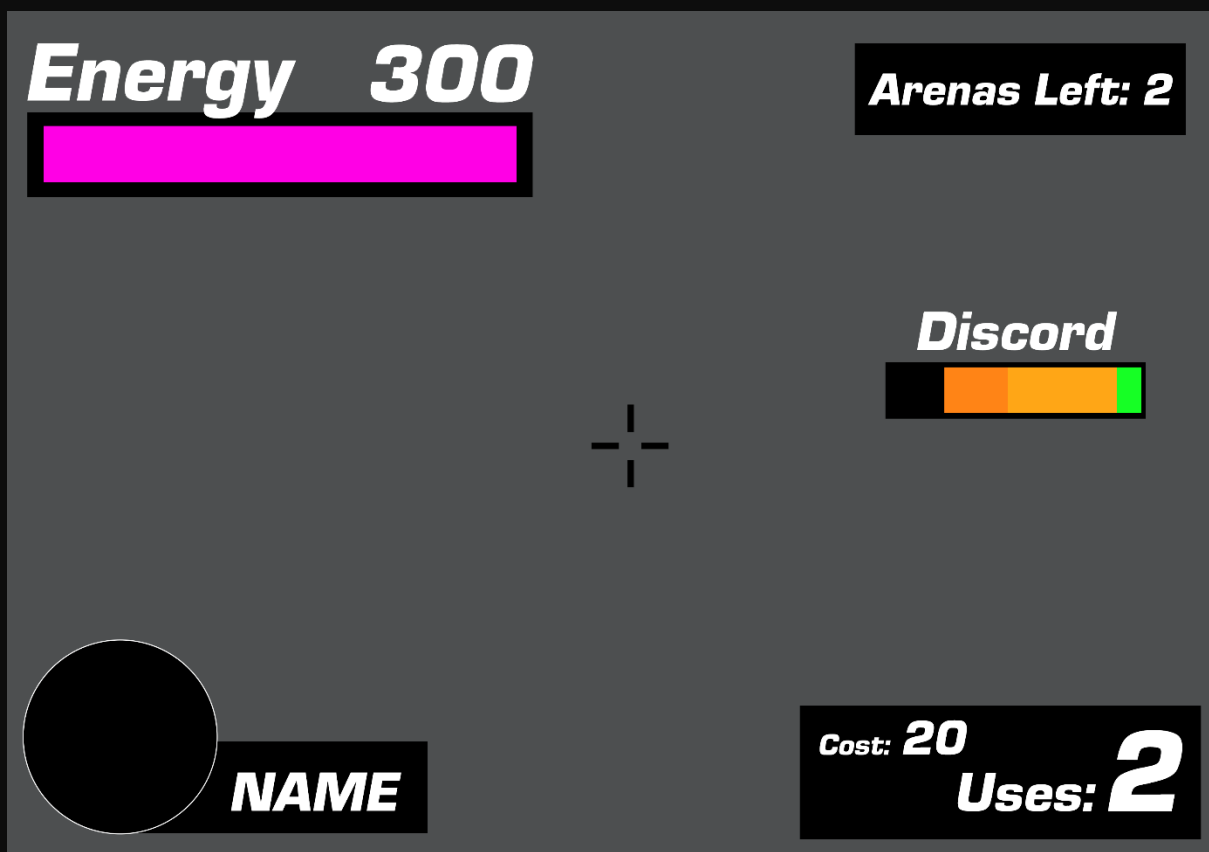
This section will detail the design process regarding the layout and structure of the UI. When it comes to designing UI, I always start out with sketching. Here is a sketch of what I would like UI to look like.



The sketch features all five of the important information that needs to be displayed. Using photoshop I then laid out gridlines to make sure that each element has equal screen space.



Next step I will take is to do a mock layout with primitive shapes.



Here is the final structure of the heads-up display for Disorderly Combat. It draws similarities to many different HUD layouts in other first-person shooters. Let's take this structure and apply some style to it.

UI Style

Style refers the visual appeal of the UI elements. Regarding style, aspects such as colours and patterns can show off the style of a UI. Style is always a preference and one can select a style based on their personal tastes. The style of the game can also affect the style of the UI, having stylish UI that closely resembles the style of the overall game, can elevate the user experience for the player. An example of a UI menu that resembles the style of the game is the menu system in Persona 5.



Figure 2: Main menu in Persona 5

For Disorderly Combat, I have the UI to replicate the style and aesthetic of games from the 2000s. Games you could boot up on a PlayStation 2. During this era, UI often features bold and blocky colours, polka dot patterns, lots of gradients, glow, and other cool effects. One game that can be pointed too is the UI style in Sonic Riders.



Sonic Riders is a racing spin off game released in 2006 for the PlayStation 2. If we look at the example image above, the style of the UI is filled with bright blue colours. Any text is in italics and is equipped with a faint orange glow. The style of this UI appeals to me because it screams out “VIDEO GAME”. Taking the UI structure into consideration, here is my attempt at applying this style towards my own UI.



The bullet points below summarise my thought patterns and creative process:

1. I took the structure described above and split the process into two parts, designing the style for the **shapes** and the **text**.
2. With the **shapes**, I wanted the colour scheme to consist of blues. I decided to go with a turquoise gradient effect for all shapes.
3. I also wanted all shapes to have a pattern background that is slightly visible. I created a polka dot pattern in photoshop, saved it as a pattern and applied it to the blue gradient shapes.
4. For the final touch, I needed something to make the shapes pop against the environment. I decided to add a dark blue stroke effect. I also added a white drop shadow effect.
5. With **text**, I wanted all text to be white, as white is the most common colour associated with text you are meant to read. I also decided to italicise the text to give it some edge.
6. Finally, I added a purple glow effect to the text to match the text found in Sonic Riders.
7. An outlier to point out, Discord and its fill gauge have their own colour scheme being a mix of pinks and purples.

With the style complete, we have conceptualised, designed, and created the UI for Disorderly Combat. Of course, we can always come back to the UI, either removing or adding elements based on feedback, but for the sake of time, let's move on to recap this week.

Week 4 Retrospective

It's been a month working on this project and reality is starting to set in that this project is well and truly underway. This week has been very busy, but I've managed to knock out level and UI design which are both very important areas in game design. I decided to give spotlight light to level design as it allows me to showcase the use cases of the "Entropy System" in the showcase video. I also gave importance to UI design as it was one of the areas we we're getting assessed in the brief. The page count for this week has ballooned due to the amount of imagery.

I feel confident with the design of the main level and the UI, but time will tell when we implement these features into the first build of Disorderly Combat. If we remember the question that I bring up for every retrospective, at this point of the project, I should retailor the question to fit the current development stage:

Have I designed engaging combat system within a first-person shooter?

The answer I would give is yes, but there are some design issues that could potentially break the flow of gameplay. We don't want this because with a game like Disorderly Combat where health is always draining, we want to reduce player frustration as much as possible. So, I believe it is time to address those issues, next week!

Week 5 – MVP and Design issues

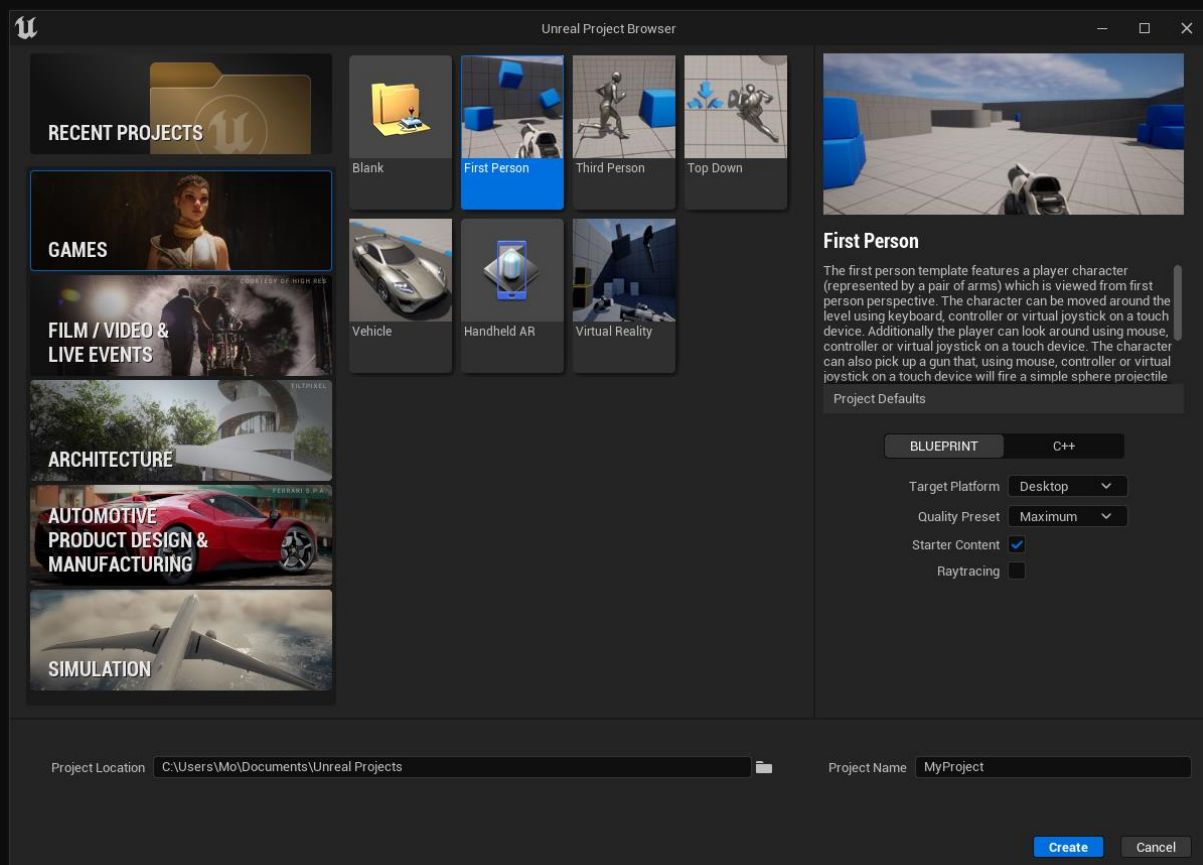
MVP stands for minimal viable product and serves as a playable prototype for video games. MVPs usually don't have any fancy effects or models and are usually just grey box environments. MVPs should contain elements that are core to your game. In case, we will be creating an MVP to showcase the basics of the Entropy System. Looking back at the Trello board for the project, we can consider all items in "Week 1" to be vital to have in the MVP. The items are:

- Core Movement
- Entropy System
- Entropy Dagger
- Ivory Wall
- The Trooper

Once all these features are implemented into our Unreal 5 project, we can consider our MVP built. Let's go into a short summary discussing the development of each element.

Core Movement

Core movement is the basic movement abilities the player will have access to during the game. These include **moving, jumping & wall running**. For moving & jumping, I decided it would be easier to start with Unreal's first person shooter starter project as it will save time. With the first-person template, Unreal already provides you with a basic movement system and a working gun that fires projectiles.



The next feature to implement was **wall running**. Using the first-person blueprint as a basis, I implemented functionality that would allow the player to grab onto a wall by pressing [W] & the key that corresponds with

the wall you wish to run on. While running on the wall, the camera will tilt at an angle to signify that the player is wall running. With wall running implemented, the next feature to implement is the entropy system.

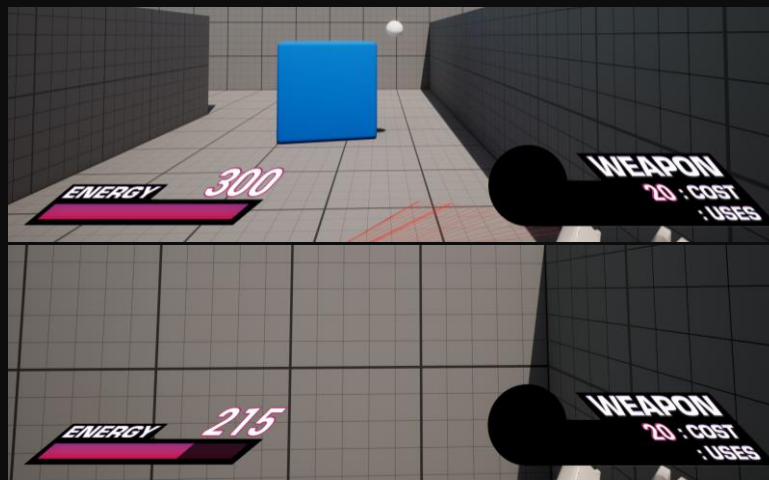


Entropy System

The entropy system is the core mechanic of Disorderly Combat. To recap the entropy system is a mechanic that allows the player to manipulate energy. The player has an energy gauge that acts as their health. Once energy runs out, the player loses. The player can release energy as “Entropy Weapons” and can absorb energy from their surroundings. The key features needed for the MVP is the energy gauge, the ability to absorb energy from objects and the ability to release energy as “Entropy Weapons”.

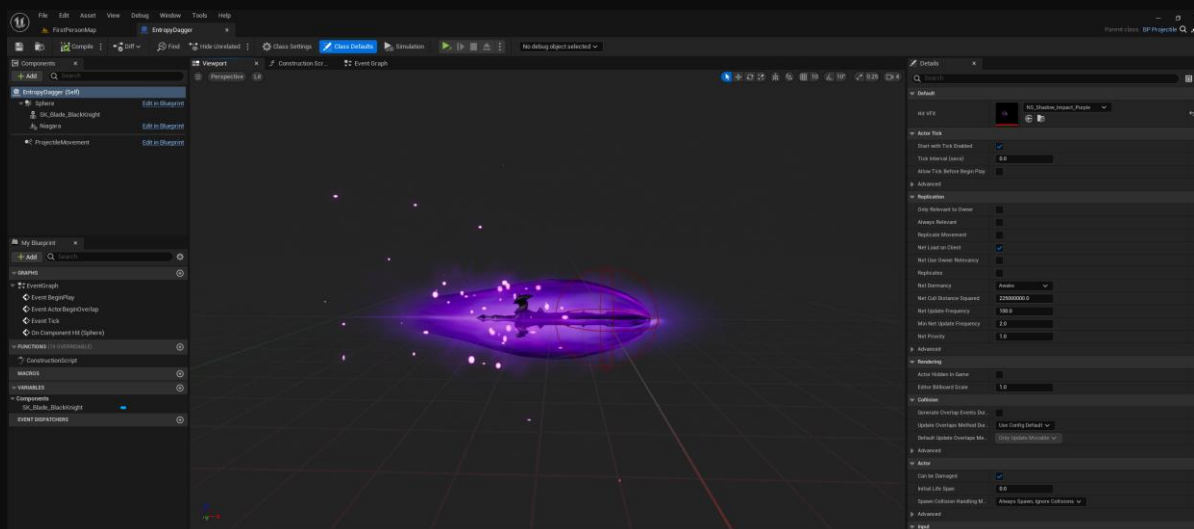


To create the energy gauge, I needed to create two float values *energy* & *maxEnergy*. *Energy* acts as the energy that gets changed during runtime while *maxEnergy* acts as your maximum energy. Next would be to display these values in-game for this I created a UI widget that displays an energy bar. The next step was to implement energy loss functionality. Whenever the player moves in game, energy begins to drain. To implement this, I located where movement inputs were tracked, there I hooked up a branch statement that would deduct energy based on movement.



Entropy Dagger

Once this was implemented, I then began working on the core entropy weapon, the entropy dagger. For the entropy dagger, the plan was to allow the player to fire an entropy dagger by pressing [LEFT MOUSE BUTTON]. This would instantiate an actor with a particle system that contained all functionality for the entropy dagger.



By switching the actor that gets spawned in the first-person blueprint to the entropy dagger actor, this allowed the player to fire entropy daggers. I also added an event that would deduct a set amount of energy based on the cost of the entropy dagger. At that moment, entropy daggers had no core functionality as there wasn't anything to fire energy at. This moves me on to working on the Pristine Wall/Platform

Pristine Wall/Platform

The Pristine wall/platform are the first objects that the player will be able to absorb energy from. Players can initiate absorbing by holding [RIGHT MOUSE BUTTON]. The first way to implement this is by creating a line-cast event once the player holds the absorb button. I added a print string event that would detect if the line-cast is hitting an actor with the tag "PristineObject". For the pristine wall/platform I created a single actor that contained a box with a wooded material. This actor had an energy value & once that energy value reaches a certain point, the actor will turn on/off its collision. I added an effect that allowed the transparency of the material to change depending on that energy value.

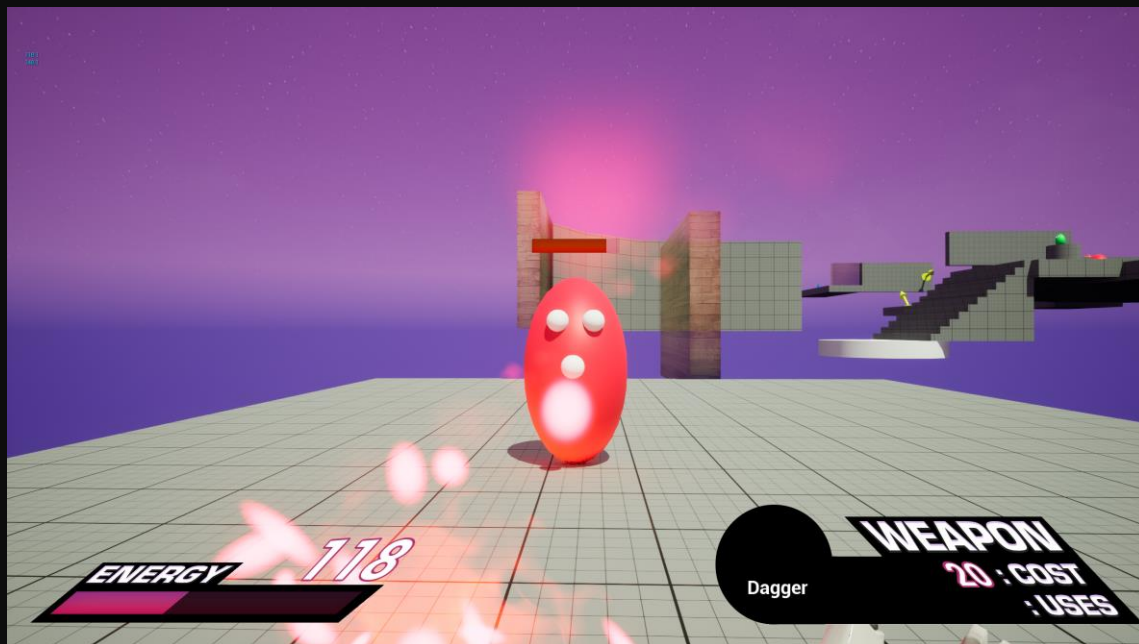


This same actor also has a `oncollisionhit` function that would detect if an entropy dagger has hit it. If it has, then the ivory wall/platform gains energy equal to the energy value of the dagger. With all the core systems built I then dropped in the level I designed for the project.



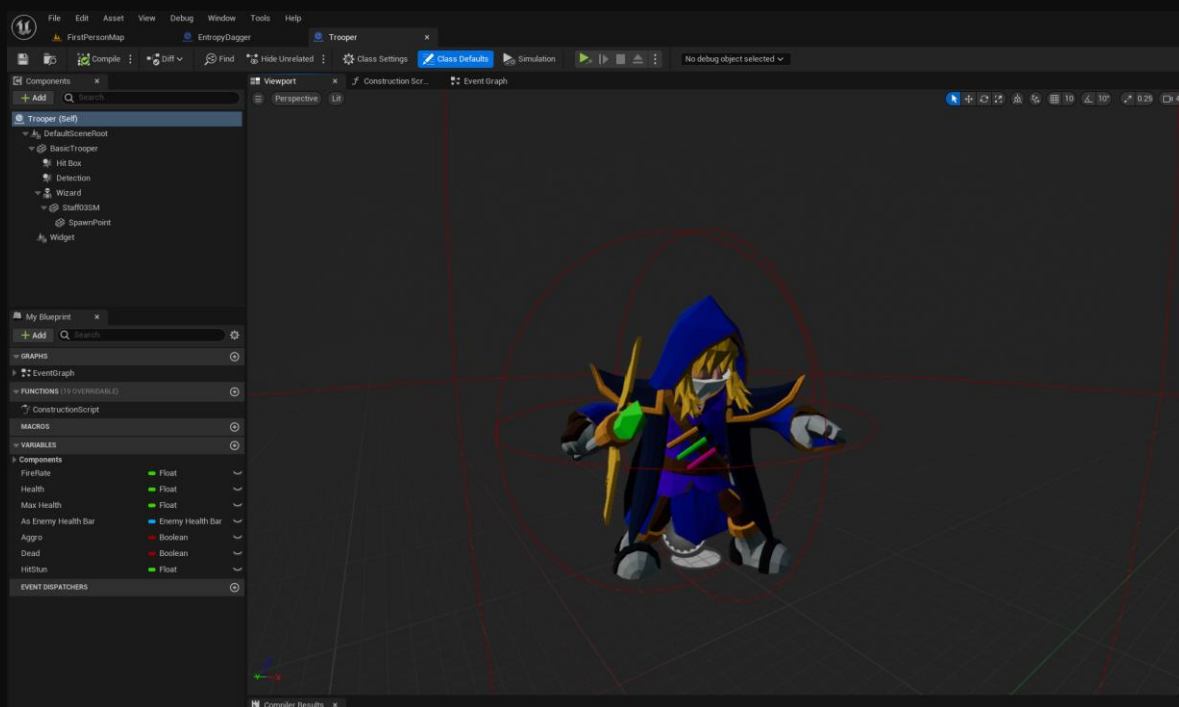
The Trooper

The next step was to develop the first enemy, "The Trooper". Using a capsule as a base model, I created an actor that would fire projectiles once the player enters their detection zone. Once the player gets close; they will begin to fire projectiles and a health bar will appear over their heads. Within the trooper blueprint is a `OnCollisionHit()` function that detects if an entropy dagger hits them. If it does, it lowers its health by the amount of energy the dagger contains. I also updated the functionality within first-person blueprint to allow the absorb line cast to detect enemy projectiles. Once it hits the line cast, the projectile will be destroyed, and the player will gain energy.

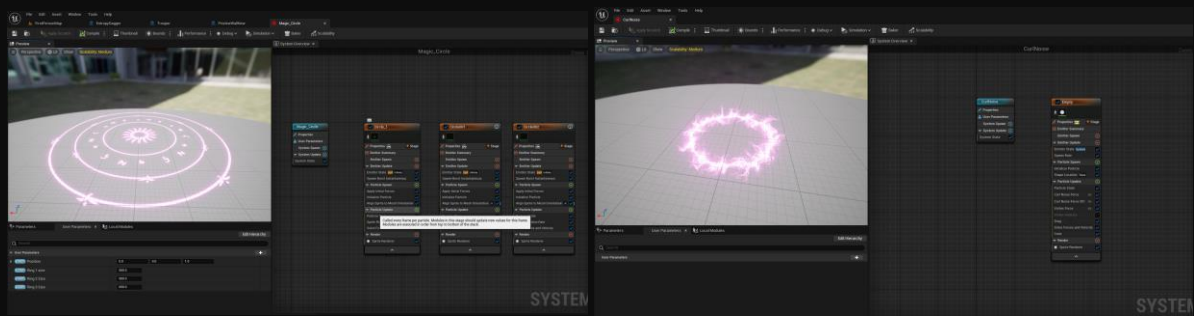


Fluff

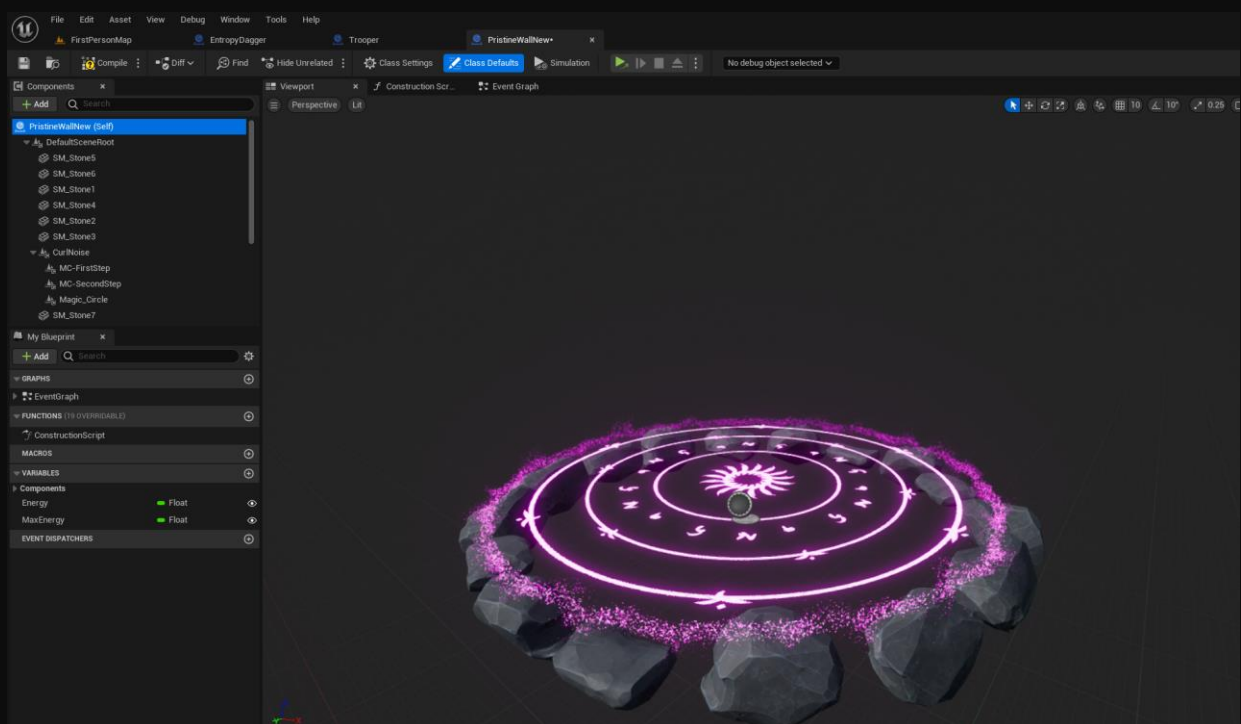
Since I had some spare time, I gave the trooper a model so that it wouldn't feel so drab. This model I found on the epic games store for free. When the enemy fires a projectile, it plays an animation and then fires a projectile out of a transform point that is attached to the trooper's staff.



Another design I updated while I was working on the MVP was the look of the pristine wall/platform. The reason I updated the design at this stage was to save time down the line. When creating the design and style of the platforms, I was inspired by magicians and magic circles. The idea then hit me to design the walls/platforms as magic circles that needed energy to become active. Using Unreal's Niagara particle system, I created two effects. One would be a magic circle that spins around in a circle, the other effect would be a particle ring that shows the player where they should aim their weapons at.



To make the new walls/platforms feel less empty, I added a ring of rocks below to give it a more arcane feel. I then updated the functionality to allow the rings to appear one by one depending on the energy value. I did this because it will tell the player clearly that firing entropy daggers will have an effect on these objects.



At this stage I was confident enough to say that the MVP for this project was completed. The first thing I did was throw this project at my friends to test and break. This leads me to my design flaws portion of the document.

MVP playtesting & Design flaws

This section will detail some design issues that either I have found during playtesting, or my friends have found during playing the MVP. Issues and flaws include:

- The Trooper feels too spongy due to high health.
- Trooper projectiles are too easy to avoid.
- Players don't gain a unique entropy weapon when absorbing trooper projectiles.
- Players don't know when they absorb something.
- Wall-Running is difficult to initiate.
- Energy drain via movement feels too strong.
- Player is hard to control while in air.

We can split these issues into categories:

- Player Movement
 - Player is stiff to control while in air.
 - Type: Functionality
 - Wall-Running is too hard to initiate.
 - Type: Functionality
- Entropy System
 - Energy drain via movement feels too strong.
 - Type: Design
 - Players don't gain a unique entropy weapon when absorbing trooper projectiles.
 - Type: Design
 - Players don't know when they absorb something.
 - Type: Design
- Trooper
 - The Trooper feels too spongy due to high health.
 - Type: Design
 - Trooper projectiles are too easy to avoid. (Fixed)
 - Type: Functionality + Design

Redesigning: Player Movement

Redesigning the movement for Disorderly Combat won't be too hard since we just need to edit some values in the editor. The culprit in question for both problems was for the lack of "Air Control". Players found the fact that you couldn't move in the air frustrating as they felt like they had to "commit to the jump". This also make wall-running more difficult as the player couldn't move closer to the wall.

Character Movement: Jumping / Falling		
Jump Z Velocity	500.0 cm/s	↔
Braking Deceleration Falling	0.0	
Air Control	1.0	↔
Air Control Boost Multiplier	2.0	
Air Control Boost Velocity Threshold	15.0	↔
Falling Lateral Friction	0.0	
Impart Base Velocity X	☒	
Impart Base Velocity Y	☒	
Impart Base Velocity Z	☒	
Impart Base Angular Velocity	☒	
Notify Apex	☐	

How I solved this issue was by simply enabling air control by setting it to 1 in the character movement component. Enabling air controlled allowed gameplay to feel more natural. It also birthed the ability to wall jump on walls for players who are skilled enough.

Redesigning: Trooper

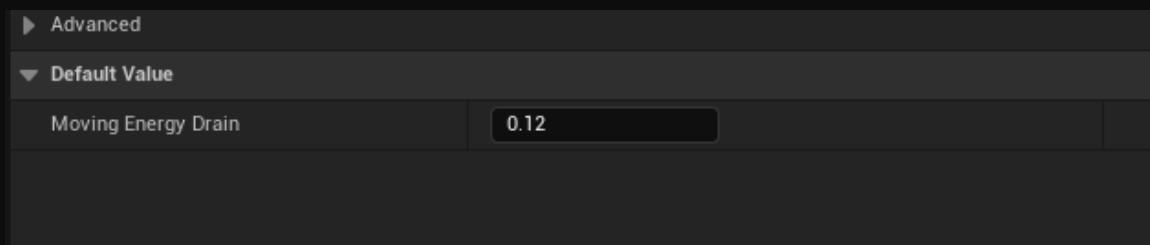
Similarly, to the player movement section, redesigning the trooper will be an in-engine task, at the same time there will be some values we need to tweak again in our narrative design document. The first value to tweak is the value of trooper health. Most players described the trooper as feeling “Spongy”, this was because the trooper required 6-7 hits to take down. This causes a problem where the player wastes almost half of their energy resource to take down an enemy. To remedy this problem, I simply changed some of the values related to trooper health calculation in the numeric design document.

Name	Type	Final Enemy Health
The Trooper	Smasher	280

This reduced the old value of 400 to a value of 280. Now the trooper requires 5 hits from an entropy dagger to take down. The next problem to solve is the fact that troopers failed to pose a threat to the player. The root of the problem could be found in the behaviour of the projectiles. Due to the fast pace of gameplay, the player rarely got hit by projectiles, most of the time the player would need to actively seek projectiles to absorb. To solve this problem, I decided to add homing functionality to the projectiles, so they posed a higher threat.

Entropy System

Unlike player movement which involved primarily functionality changes, issues with the entropy system are rooted in its core design. To fix core design issues, it's time for some redesigns. The first problem to tackle is energy drain via movement feels too strong. We can fix this by delving into the numeric design document and changing the value for energy drain. Currently the value for movement energy drain is 15, this gives the players 20 seconds of pure movement. The problem is that this doesn't account for firing entropy weapons. During playtests firing weapons while moving would only allow you to fire 4 daggers while moving for a few seconds. Let's change this value to 10, this would give the player 30 seconds of movement. When testing this in-game, the issue now arrives that energy drain feels too low.



The screenshot shows a dark-themed user interface for game settings. At the top, there is a tab labeled 'Advanced' with a right-pointing triangle icon. Below it, a section titled 'Default Value' is expanded, indicated by a downward-pointing triangle icon. Under this section, there is a row with the label 'Moving Energy Drain' on the left and a text input field containing the value '0.12' on the right. The input field has a dark border and a small clear button (an 'x' icon) on its right side.

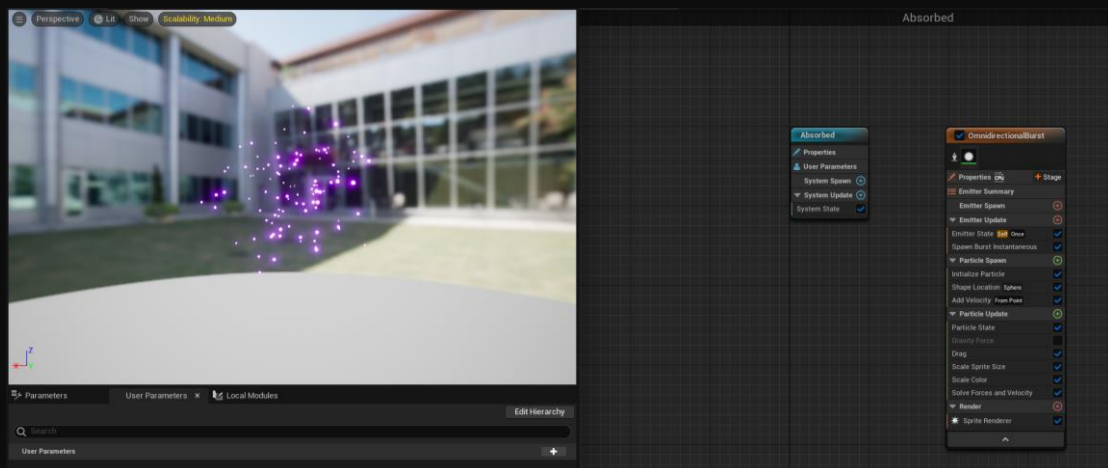
Let's change the value again to 12, this would give the player 25 seconds of movement. Testing this in game, this value feels more appropriate.

The next area to address is centred around the core design of the game and that's the entropy weapon selection. As of 18th January 2023, troopers don't provide the player an entropy weapon when you absorb it. The reason this is an issue is because players don't learn that you can gain new weapons until they fight more advanced enemies. To fix this, I would need to design a new entropy weapon. Here is the entropy weapon I have designed.

Name	Entropy Dual Dagger
Targeting Type	Single Target
Description	<i>A special variant of the Furi's favourite weapon. Double the amount for double the carnage</i>
Stats	Damage – Low Speed – Fast Range – High Rate of Fire – Very High
Behaviour	Can be used to damage enemies. Can also be used to fill up objects with energy. Dagger will home into the closest target and dissipate once it goes out of range.
Uses	Uses – 2

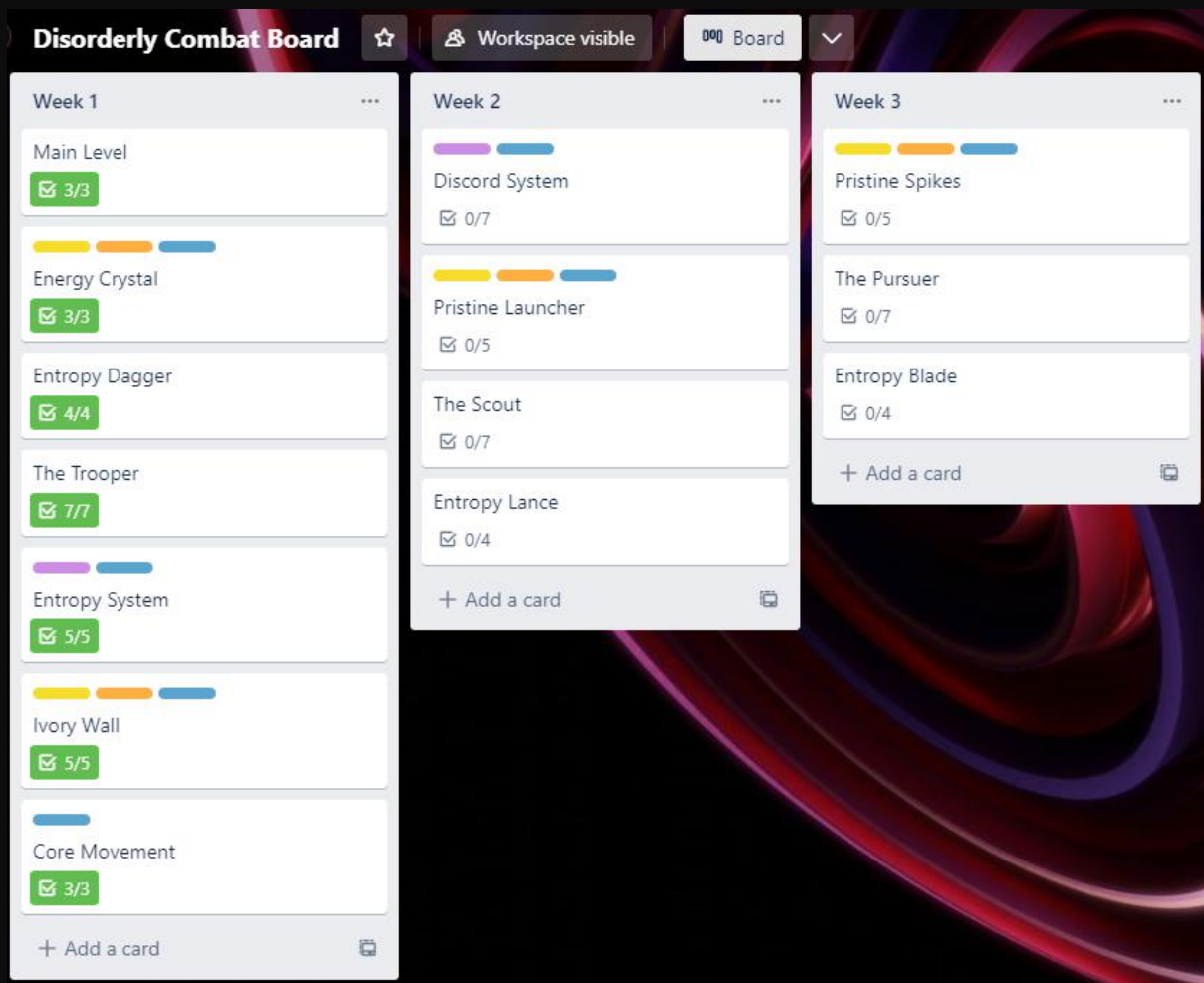
In the numeric design document, I've added the additional numeric stats for the new entropy weapon. For the sake of time, I will not be implementing this new entropy weapon this week. If I manage to finish all the implementation tasks on Trello, then I will implement the dual dagger. The next issue to solve the lack of feedback when the player absorbs a projectile.

Currently when the player absorbs an enemy projectile, the projectile actor gets destroyed and the player gets provided with energy. While this works well enough, during the heat of battle when the player will be focusing on many different elements, absorbing something like a small projectile could be easily missed. Providing the player with feedback will help notify the player that they have successfully absorbed a projectile, so they can stay focused on fighting enemies or platforming. A fun way to notify the player while sustaining the flow of gameplay, is to create a particle effect to signify successful absorption.



Using a Niagara particle system, I created a simple burst particle effect that will be created when the player absorbs an enemy projectile. This will provide the player with visual feedback while also fitting into the theme of Disorderly Combat. To take this even further, I created an event in blueprints to slow down time for a split second when the player absorbs a projectile. The purpose of this event is to help aid the player to focus back on the enemy after diverting their attention to a projectile. Also, it feels very satisfying to slow down events to show the player that they've performed a cool action such as absorbing an enemy projectile.

Trello Updates



Week 5 Retrospective

With week 5 coming to close, I can say that I've nicely adjusted to the workload. Before signing up to "Search for A Star", I had zero experience working in the Unreal Engine. Most of my development experience has been in the Unity Engine. Nevertheless, developing a game in the unreal engine wasn't as difficult as I assumed it would be. A case could be made that I've transferred the skills I've learnt in Unity to Unreal, but I still feel like I'm developing a new skill which is always a good thing.

Translating my design work into a working prototype has been a very enjoyable experience. The entropy system already feels very satisfying to use. The act of firing entropy weapons already works well since the Unreal Engine was made to develop first person shooters. Absorbing energy from crystals, walls or projectiles also works well. The pristine walls/platforms, both look cool and communicate to the player their function. Developing the AI for the trooper, while challenging, has created structure for the different types of enemies that I will be showing in the showcase. The main takeaway from this week is the feedback from the first test build of the game.

During the development of the MVP, I got my flatmates to sit down and play the first section of the showcase level. The feedback the players have provided have helped open my eyes to some current and future design issues with the game. The main feedback that was received was that the game felt like a fast first-person shooter. One of the players even said, "I can see the neon white inspirations". It's time to ask **the question**.

"Have I designed engaging combat system within a first-person shooter?"

The answer to this question, using feedback given by the testers, this answer would be yes. The entropy system provides a unique challenge where the player must manage their energy during a level. Entropy weapons provide a unique twist where a tool is used for both platforming and dealing damage to enemies. Absorbing energy from objects and projectiles creates an interesting dynamic where the player needs to pay attention to their surroundings to ensure they don't run out of energy and die. We can now tick off the first criteria of the brief since we have enough to show a simple use case for our unique mechanic. Next week we will be implementing features that will help us fulfil the remaining criteria of the brief. These features are the Discord system, the Scout enemy, Entropy Lance, and the launcher.

Week 6 – Further Development

Week 6 has arrived, and we are starting off strong. This week we will be continuing development on the prototype of Disorderly Combat. We will be implementing features that meet the 2nd and 3rd criteria for the project brief. As a reminder the criteria are:

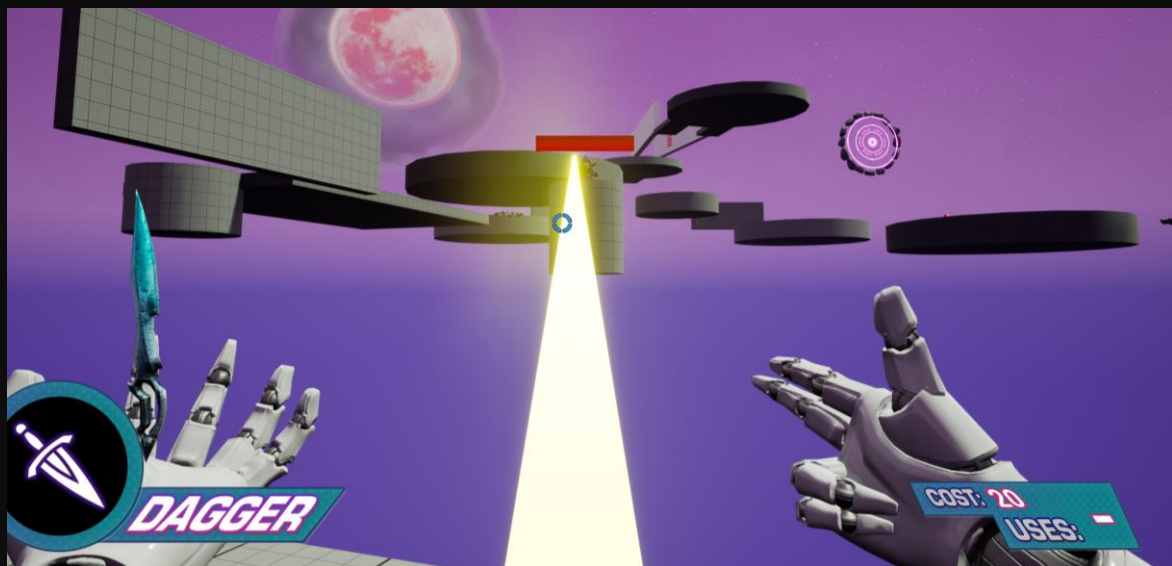
2) *A trickier or more thought-provoking implementation*

3) *An implementation working in tandem with another mechanic*

The features that will hit these criteria for the brief are the Scout, Entropy Lance, Discord System, and the launcher. The Scout and Entropy Lance will focus on the 2nd criteria. The Scout is an enemy that will charge up and fire an attack that will kill you in one hit. The player will need to absorb that attack and then fire it back at the scout to stop them from firing another attack. The Entropy Lance is the weapon the player will receive when absorbing that attack. This weapon will be a limited use weapon like the Dual Dagger. The Discord system will be the underlying mechanic that hit the 3rd criteria. This system will essentially turn the game's playstyle on its head. This system rewards the player for playing in an aggressive trigger-happy manner. The launcher is another platforming obstacle that will be necessary for many platforming challenges in the showcase level. So, let's dive into the development process for these features.

The Scout

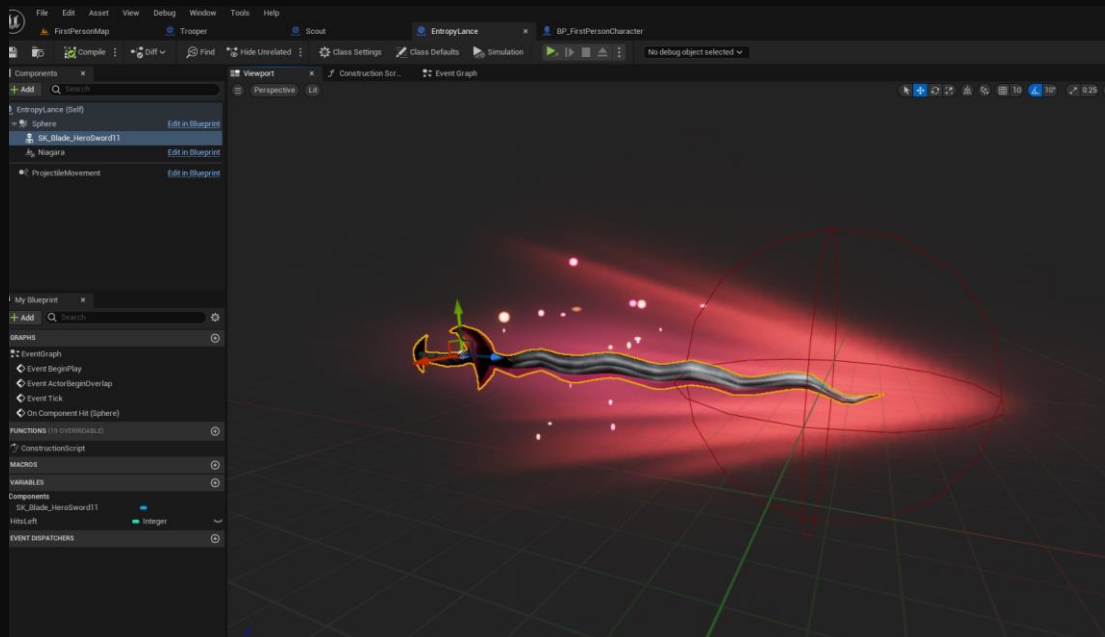
Implementing the features of the scout was a simpler process than developing the trooper. This is because I used the blueprint created for the trooper as a starting off point for the scout. In terms of behaviour, the plan for the scout is that when the player enters the scout's view zone, a laser will pinpoint onto the player. Gradually the colour of the laser will change in colour to show that the scout is about to attack. Once a timer has reached, the scout will fire a fast-moving fireball that will obliterate the player in a single hit.



Enemy detection was created using the same functionality as the trooper's detection collision sphere. To create the laser, I created a Niagara particle system that emits a line that glows and locks onto the player. In the Scouts blueprint, the colour of the laser changes depending on the value of a timer value to determine the charge time of the enemy's attack. Once the time reaches its max value, the enemy will fire their attack and will start to charge their next attack. Once the player absorbs that attack, they will gain access to the Entropy Lance.

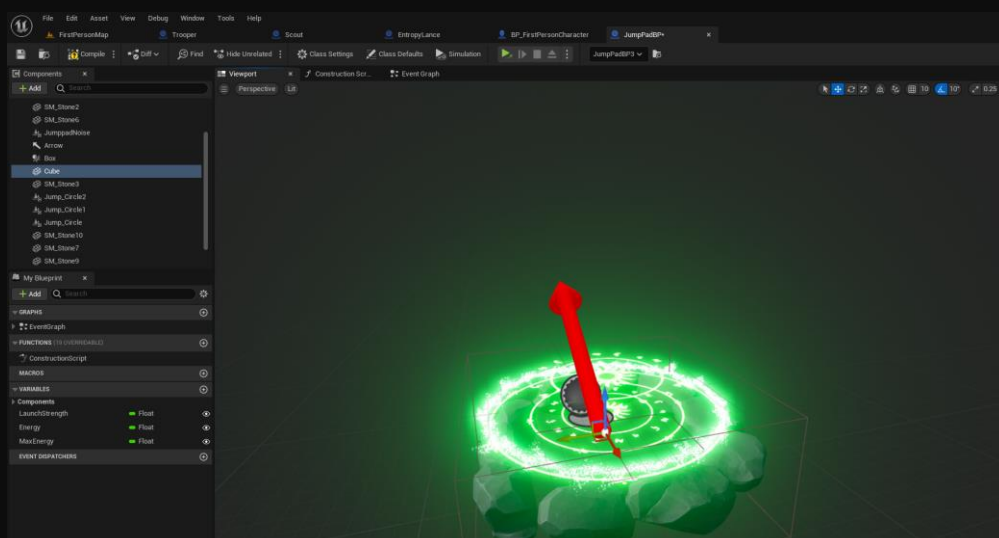
Entropy Lance

Similarly, to the development process of the scout, most of the work for the Entropy Lance has already been done when developing the Entropy Dagger. Most of the functionality including, firing and energy costs already have variables I created to prepare for this stage. Like the dagger, the player can fire the lance by pressing [LEFT MOUSE CLICK]. The lance also has uses that correspond to the amount of uses that can be found in the numeric design document. The lance can provide energy to objects like the dagger but provides more energy and deals more damage. Another feature to implement was the piercing capabilities of the lance. This achieved by creating a float that stores the number of enemies the lance can pierce through. An overlap event takes this and deducts 1 each time the lance hits an enemy. Once it hits zero, the next enemy it collides with will cause the lance to dissipate.



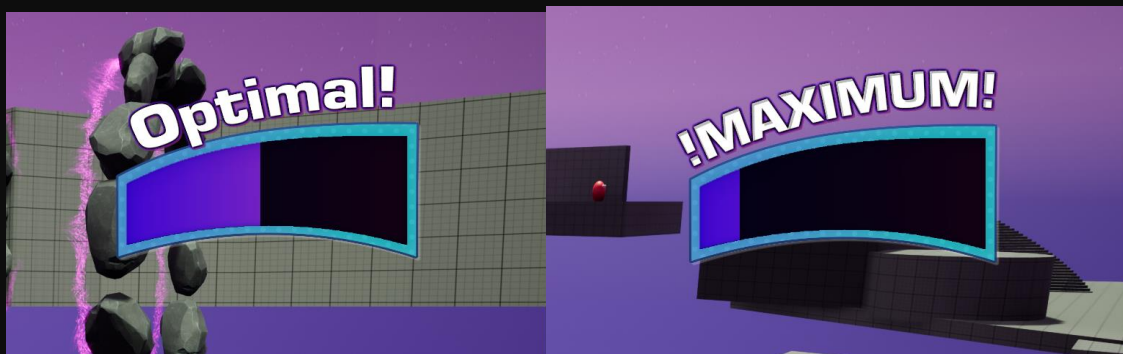
Pristine Launcher

The launcher works similarly to the pristine walls/platforms in its base functionality of receiving energy from entropy weapons. Where these similarities end is in the unique functionality. When the player collides with the launcher, the player will be sent flying in the direction the launcher is facing. This was quite simple to get working as all it needed was a launch character node and to adjust the strength of the launcher with a float variable. In terms of the aesthetics, I used the same style of magic circle to keep consistency.



Discord System

The discord system was the most involved process of this week as it involved creating some UI elements for the showcase. Mentioned prior the discord system is an underlying mechanic that promotes aggressive play. In a game where the player can end up conserving energy, a mechanic like this rewards player who play fast and trigger-happy. In terms of functionality, the discord system is split into four states, none, minimal, optimal and maximum. The player will be constantly switching through Enum states during a level, with each state changing the value of different variables. The two main variables that are affected are the players maximum energy value and maximum movement speed.



The player can build discord by absorbing enemy projectiles and hitting enemies with entropy weapons. When this discord value reaches a certain threshold, the player's discord state will change, and the players energy and speed will update to the value kept in the Enum state. Once the player enters that state, they state won't be dropped until discord drops to 0. For example, when the player reaches the "Maximum", they will stay in "Maximum" until discord drains completely. Each state also has a float value associated with its decay rate. The discord bar will always decay once the value of discord goes over 0. The discord bar decays faster when in "Maximum".

Other Features

A minor feature I would like to discuss is how I got the players arms to animate. When the player fires an entropy weapon, the player's left hand clenches, and the right hand fires the weapon out of their fingertip. To do this, I just created some animations using the first-person skeletal mesh.



Trello Updates

The image displays three Trello boards, each representing a week. Each board contains several cards with progress bars and checkboxes. The cards are organized into columns, and each card has a title, a progress bar, and a checkbox with a count.

Week 1

- Main Level: 3/3
- Energy Crystal: 3/3
- Entropy Dagger: 4/4
- The Trooper: 7/7
- Entropy System: 5/5
- Ivory Wall: 5/5
- Core Movement: 3/3

Week 2

- Discord System: 7/7
- Pristine Launcher: 5/5
- The Scout: 7/7
- Entropy Lance: 4/4

Week 3

- Pristine Spikes: 0/5
- The Pursuer: 0/7
- Entropy Blade: 0/4

Each board has a '+ Add a card' button at the bottom.

Week 6 Retrospective

This week marks the second week of pure implementation, with 2 more weeks left. I would be lying I didn't say that I'm starting to feel pressure of time, but at the same time, I'm extremely excited. I want to see what everyone else that is taking part in the "Design" brief has conjured!

The process of implementing this week's features, has been a somewhat smooth experience. Sadly, I couldn't find the time to get some of my friends to test the prototype. To recap what has been implemented, I managed to implement functionality for the scout, entropy lance, jump pad and discord mechanic. Regarding the brief, I can say that I've hit brief the second and third criteria this week.

The scout is an enemy that can kill the player in a single hit and at the same time, the player won't be able to reach the scout with most entropy weapons. Dealing with the scout forces the player to pay attention to their attack window to absorb their attack. Absorbing that attack will reward the player with the "Entropy Lance" which allows them to take the scout out. The Discord system is an underlying system that rewards the player based on their performance with the core mechanic. The most efficient way to build discord is to land attacks on enemies, this promotes aggressive play in a bid to prevent players from staying on the defensive. With that said, its question time:

"Have I designed engaging combat system within a first-person shooter?"

Branching from last week's retrospective, I will answer this with a double yes! Difficulty enemies that challenge the players skill and systems that reward the player based on their skill are two aspects that generate engaging gameplay. If players aren't challenged, some may start to get bored thus engagement will drop.

It's starting to dawn on me at this point that this Game Design Document is quite big in size, but I believe its justified. One thing that was mentioned in the brief was, *"you are encouraged to push beyond the base outline for the brief"*. I wanted to show the reader of this document, and myself, that not only can I provide what is needed from the brief, but also stand out from the crowd by showcasing my design skills. Aspects of design such as Numerical Design and Level Design weren't specified, but I believe it's necessary for this project, especially for a mechanic (the entropy system) that draws its power from enemies and the level.

Next week I will continue implementing the tasks found on Trello, as well as addressing some more design issues that have come to light.

Week 7 – Further Development & Design Issues #2

This week we will continue development of the build, we will be focusing on the pursuer, entropy blade and pristine spikes. Before starting on that, I want to discuss two further design issues that have reared their head.

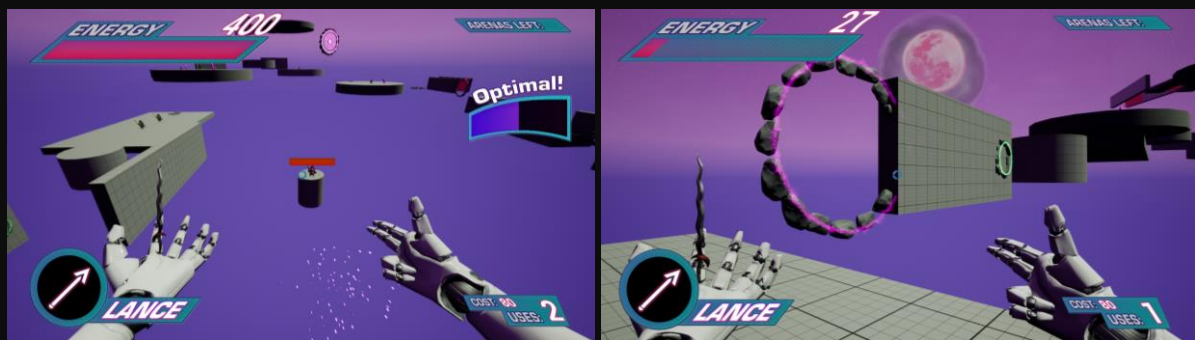
Design Issues

For context, I discovered these issues while testing the game. Each issue relates to a part of the entropy system. The issues can be paraphrased like so:

- The player is unable to switch entropy weapons (Releasing).
- The player can hold absorb without drawback, promoting passive play (Absorbing).

Issue #1: Releasing

This issue stems from the fact that players can't switch weapons during gameplay. This a problem is because some players may want to save Entropy Weapons for scenarios that would benefit with said weapons. An example of this issue showing up is with the first scout encounter. When the player encounters and absorbs the scout's projectile, they will gain the entropy lance. The entropy lance is a weapon that specialises in taking out enemies at a distance and can pierce through enemies. Problem is that the next section after the scout encounter is a platforming section with the launch pads. Though the lance will be able to activate the pristine wall in 1-hit, some players may deem this a waste.



To remedy this issue, I've designed a minor feature that will allow players to switch weapons, this feature is called "Stockpile". Instead of one slot for an entropy weapon, the player will have two slots. The first slot will always be reserved for the entropy dagger, while the second slot is free. When the player gains a new entropy weapon, that weapon will be stockpiled into the second slot. For instance, when the player gains the entropy lance for the scout, that lance will be placed into the second slot. It's important to note that the Entropy Dagger can't be replaced and will always be reserved in the first slot. If the player absorbs a new entropy weapon, then it will replace the Entropy Weapon in the second slot. The purpose of this new feature is to provide the player with more opportunities to experiment with each entropy weapon. For example, once the player completes the platforming section above, the player will run into an opportunity to use the lance against some troopers arranged in a line. Without this feature the player will be forced to use the lance before they can get to this point. One of the key design pillars of Disorderly Combat is "experimentation", so creating a feature like this is key to allow for more opportunities to experiment.

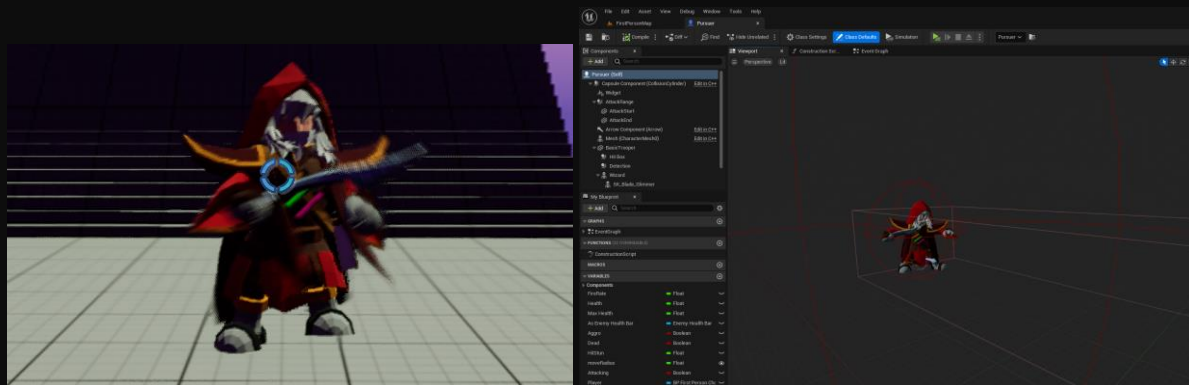
Issue #2: Absorbing

The next issue revolves around absorbing enemy projectiles. During a combat encounter, the player should have an active balance between absorbing projectiles and firing projectiles. Currently, there is no mechanic in place that stops the player from absorbing for an extremely long amount of time. Two of the key design pillars in Disorderly Combat are "frantic" and "aggression", this describes the pace and flow of combat. A tactic such as holding the absorb button for a long time kills the flow of gameplay, promotes passivity while also trivialising much of the difficulty. To combat this issue I created another mechanic, which I call "Absorb Deterioration".

“Absorb Deterioration” is a mechanic that reduces the amount of energy absorbed from a projectile depending on the length of the absorption prior. How it works is that there is a float value that will be multiplied to the value of any energy gained from a projectile. When the player presses the absorb button, this value will be high, while the player holds the absorb button, this float value deteriorates over time. For example, if the player presses absorb, the multiplier value will start at 1.5x, over time the value will diminish to 0.5x, this means the player will gain extra energy when they execute absorbing at the right time but will gain half energy when holding down the button for too long. When the player lets go of the absorb button, then this value will go back to its original value, in this case 1.5x. Players will also only be able to gain entropy weapons by absorbing attacks when this value is above 1x. This mechanic will penalise players that stay on the defensive too long by reducing the energy they gain. At the same time this mechanic rewards players who only absorb when necessary.

The Pursuer

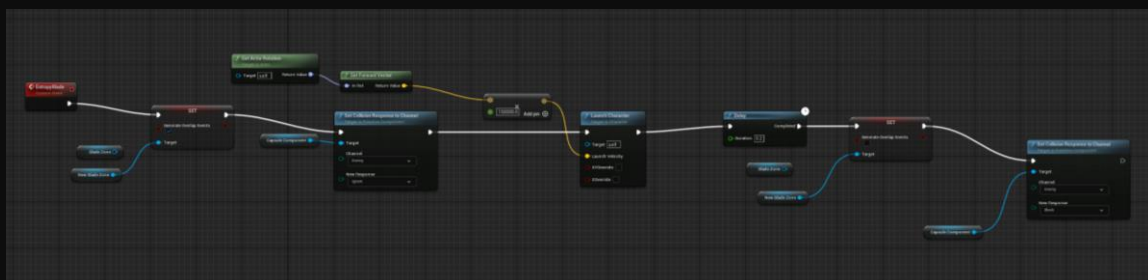
To create functionality for the pursuer, I started off using the trooper actor as a basis. When the player enters the pursuer's sights, the pursuer will begin to chase the player. Once the pursuer gets into close enough range, the pursuer will attack. During a specific frame of the pursuer's attack animation, a line will be cast. If this line hits the player, a branch statement checks if the player is currently absorbing. If the player is absorbing, then the player successfully absorbs the attack and receives the Entropy Blade. This also stuns the pursuer and leaves them vulnerable to attack. If the player isn't absorbing, then the player takes the hit and receives damage.



The most challenging aspect of the pursuer was getting the chasing AI to work. The pursuer is the only enemy in this prototype that can chase the player. Another aspect that proved difficult was getting the pursuer's attacks to land. The next feature I implemented was the Entropy Blade.

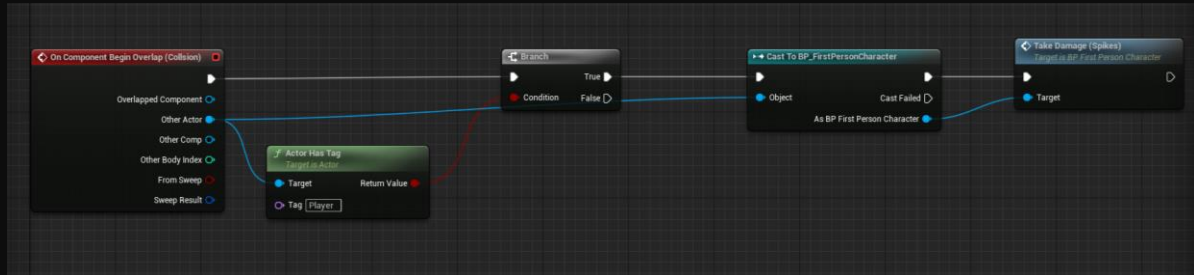
Entropy Blade

In concept the Entropy Blade was a weapon that allowed the player to home into a target and dash to their location, but for the sake of time, the behaviour of this entropy weapon was slightly changed. In the prototype, the Entropy Blade allows the player to dash forward in the direction they're aiming. Any enemy that gets caught in the dash gets hit by the player. To get this weapon to work, when the player fires the weapon, the player will dash forward at a set velocity, during the dash, I changed the collision behaviour of the player so that they would ignore enemy colliders.



Pristine Spikes

The pristine spikes were the last obstacle to be implemented for the prototype. This feature was quite simple to implement. In design, the spikes deduct a set amount of energy every time the player collides with it. To get this to work, all I needed was a box collider with an `OnCollisionBeginOverlap()` function. It checks if the player has overlapped and if they have, then a set amount of energy gets deducted.



Entropy Stockpile & Absorption deterioration

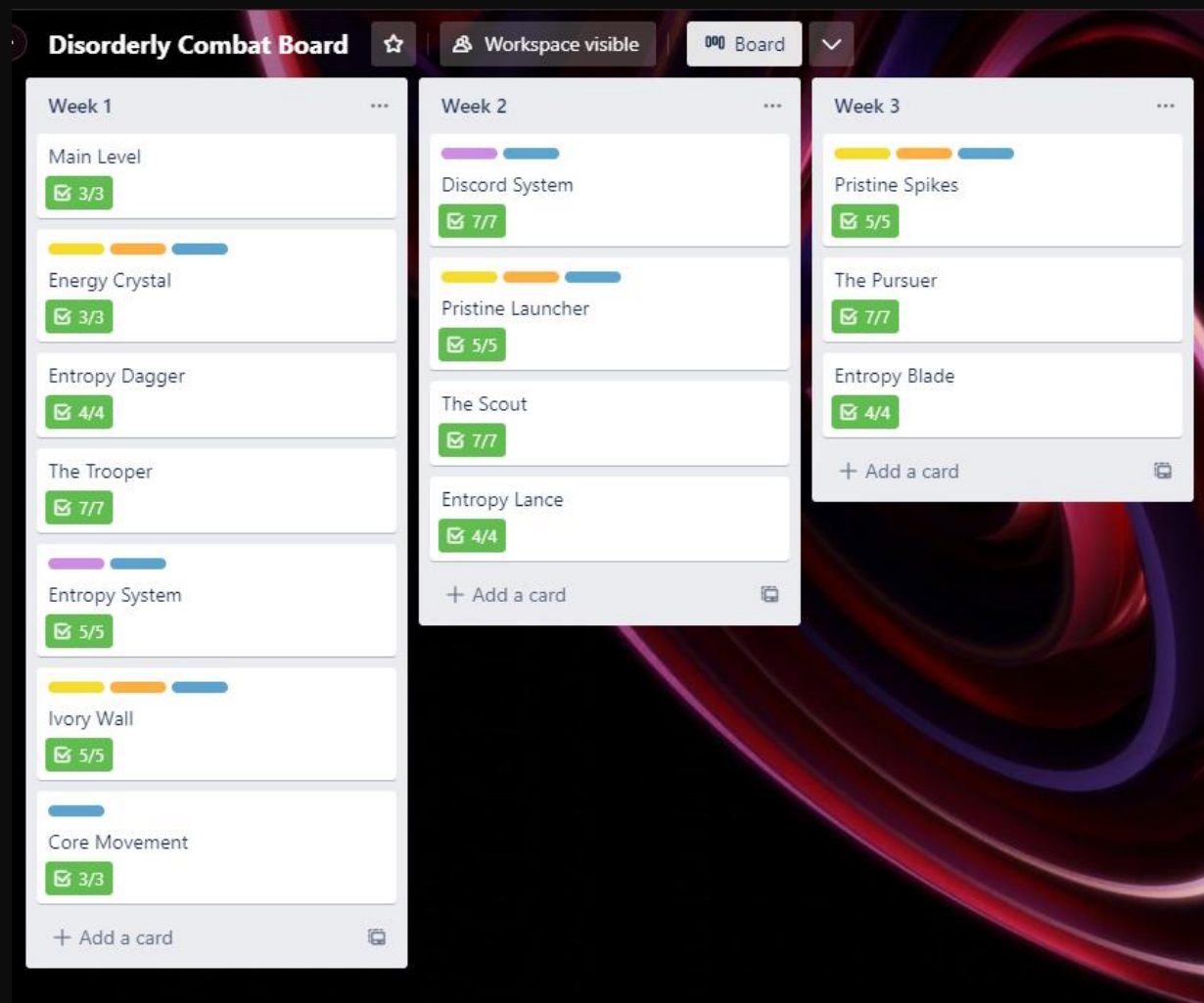
Starting with the Entropy Stockpile, to get this feature working, I need to create two separate slots for Entropy Weapons. Slot 1 would always contain the Entropy Dagger, while Slot 2 contains either the Entropy Lance or Blade. I created an integer value to track with slot is currently active. Depending on the slot that is active, behaviour in the first-person weapon blueprint and the UI changes. When the player absorbs either the lance or blade, slot 2 becomes visible and will become the active slot. When the player presses a button, they can switch the current slot.



Moving onto the Absorption deterioration mechanic, mentioned prior, I had to create a float value that represented a deterioration multiplier. This value is multiplied against any energy absorbed from a projectile. This value starts at 1.5x and can reach 0.1x at the lowest. This value diminish overtime while the player is absorbing. To indicate this information to the player, I created a circular UI bar that will represent this value. This UI bar will replace the reticle when the player is absorbing. I also created a text element that will display the multiplier value during real time.



Trello Updates



Week 7 Retrospective

This week has been an interesting one, as with the beginning of next week, I would've been working on this project for two months. Time has flown by! This week has also been the last development focused week. Checking the Trello update image to the left, I've officially completed all the mandatory tasks for three-week sprints.

Diving deeper into development, I spent time focusing on adding extra features to further hit the criteria set by the brief. Like the Scout, the Pursuer is another enemy that tests the players reaction skills. To dispatch this enemy, the player needs to absorb at the right time to stun the enemy. The Entropy Blade was another entropy weapon that was implemented, this weapon is more difficult to use than the dagger and lance as the player needs to get up close for this weapon to be effective. The Pristine Spikes were hazard implemented to increase path traversal difficulty. In regards to the third criteria of the brief, both the stockpile and deterioration systems were designed to address issues regarding the feel of the game. Question Time:

"Have I designed engaging combat system within a first-person shooter?"

Though I've said yes multiple times throughout the course of this document, I feel like solving design issues relating to the feel of the game has helped cultivate a more engaging shooter experience. Experimentation and aggression are both two important design pillars of the project, so making sure the game remains true to these pillars can help me design a more engaging experience.

With next week will be the final week working on this project. As such, that week will serve as a reflection on the project. Outside of that, I will be focusing on planning the showcase video, adding hi-fi assets to the project, and then finally recording the showcase video.

Week 8 – Polishing & Publishing

This week will serve as the final week to work on this project. As it's the last week, nothing major will be developed or designed as I will be focusing on polishing the project I already have. Let's start with assets.

Polishing: Hi-Fidelity Assets

This section will detail the process of transition the level from grey boxes to using hi-fidelity assets. Since I won't be marked on assets, I've paraphrased my process isn't short bullet points.

1. Changed all arenas and paths to rock-like podiums instead of grey-boxed rectangles.
2. Created a custom 3-way blend material and applied it to all arenas and platforms.
3. Used the mesh paint tool to create pathways that will lead the player in a general direction.
4. Added blue grass using the foliage tool.
5. Replaced all wall-run walls to wooden walls.
6. Added trees and other plants.
7. Added light sources to brighten up the level.

Screenshots



Stretch Goal: Entropy Dual Dagger

Since I had enough time left, I decided to add the entropy weapon I designed due to player feedback. It functions like the dagger except it has homing capabilities. To implement this, I just needed to copy the homing functionality from the enemy bullets. The player will be able to receive this weapon by absorbing the troopers' projectiles.



Showcase Video: Planning

This section will detail the planning stages for the video that will showcase my unique core mechanic. The general goal for this video is to inform watchers on how the core mechanic works and how I hit each criteria in the brief. This video will be of high quality with feature commentary to complement it. I will be using Premiere Pro to create this video. Let's start with a rough plan.

Rough Structure

The rough structure of the video should detail all the topics I will discuss in the video. The topics I will discuss include:

1. Introduction
 - a. Introduction of game title, background info on the project.
2. Short Overview
 - a. Discussing the core mechanic of the game, the entropy system.
3. Basic Use (criteria #1)
 - a. Showcasing and discussing core movement, energy management, absorbing, and releasing.
4. Advanced Use (criteria #2)
 - a. Showcasing and discussing enemies, other entropy weapons and their behaviour.
5. Underlying Systems (criteria #3)
 - a. Showcasing and discussing absorption deterioration and Discord mechanic
6. Conclusion
 - a. Short summary of deliverables.

Detailed Structure

Introduction

The introduction will be at the start at the video. It will introduce Disorderly Combat with a flashy entrance and a few sentences about what the background of this project. The introduction will be short.

Short Overview

This section will give a short overview on the game. I will be reading out the executive summary of the game to give the watcher some context on the game. I will overlay this section with general gameplay footage of the game.

- Video: Gameplay Montage
- Commentary: Overview on the game

Basic Use (Criteria #1)

This section will discuss, core movement and the basics of the entropy system. This section will be segmented into different parts, with each part discussing a different topic. The topics include:

- Core Movement
 - Details: Movement options (moving, wall running)
 - Video: Clips of movement in play
- Entropy System
 - Details: Introduction of core mechanic, energy drain while moving, energy drains depending on movement options. Energy can be gained or drained by absorbing or releasing.
 - Video: Two clips side by side representing absorbing and releasing.
- Absorption
 - Details: What is absorption is (the process of gaining energy from objects), how to absorb energy (holding rmb), what can the player absorb energy from (objects found around a level).
 - Video: Clips of player absorbing energy from objects.
- Releasing
 - Details: What is releasing (the process of channelling energy into physical projectile weapons known as Entropy Weapons), how to release (Left Mouse buttons), what can the player release (Entropy Weapons such as the entropy dagger and its behaviour).
 - Video: Clips of player firing a projectile, the entropy dagger and its stats displayed
- Other
 - Details: Description on objects that are affected by the Entropy System. These include the energy crystal, pristine platforms, and pristine launcher. This section will detail their behaviour regarding absorbing and releasing.
 - Video: Three clips of the energy crystal, platform, and launcher side by side. When focusing on a single obstacle, the video will zoom into the clip of that obstacle.

Advanced Use (Criteria #2)

This section will detail some advanced use cases of the Entropy System. This section will mainly focus on the enemies, showcasing their behaviour as well as the different entropy weapons you can gain from them. The topics include:

- Advanced Use Overview:
 - Details: Discussing that the player can use the entropy system to gain powerful weapons and retaliate against enemies
 - Video: General montage fighting against enemies.
- Enemy Overview:
 - Details: Description about the core design philosophy of enemies (Enemy types). Introduce the trooper, scout, and pursuer.
 - Video: Idle clips of each enemy side by side.
- The Trooper:
 - Details: Discussing the trooper, stats, and behaviour
 - Video: Clips showing idle of trooper with stat table scroll in. Each fight against the trooper.

- The Scout:
 - Details: Discussing the scout, stats, and behaviour
 - Video: Clips showing idle of scout with stat table scroll in. Each fight against the scout. Show scout kill player.
- The Pursuer:
 - Details: Discussing the pursuer, stats, and behaviour
 - Video: Clips showing idle of pursuer with stat table scroll in. Each fight against the pursuer. Show pursuer kill player.
- Special Entropy Weapon Overview:
 - Details: What are these weapons. How they can generally be obtained. How to switch entropy weapons and introduce entropy stockpile.
 - Video: Clips that replace the three clips of the enemies from above. Show weapons in hand spinning.
- Entropy Dual Dagger:
 - Details: Stats and Behaviour. How to obtain the dual dagger
 - Video: Clips of player using the dual dagger with stat table overlayed.
- Entropy Lance:
 - Details: Stats and Behaviour. How to obtain the lance
 - Video: Clips of player using the lance with stat table overlayed.
- Entropy Blade:
 - Details: Stats and Behaviour. How to obtain the blade
 - Video: Clips of player using the blade with stat table overlayed.

Underlying Systems (Criteria #3)

This section will detail the underlying systems at play that relate to the entropy system. These are energy deterioration and discord mechanic.

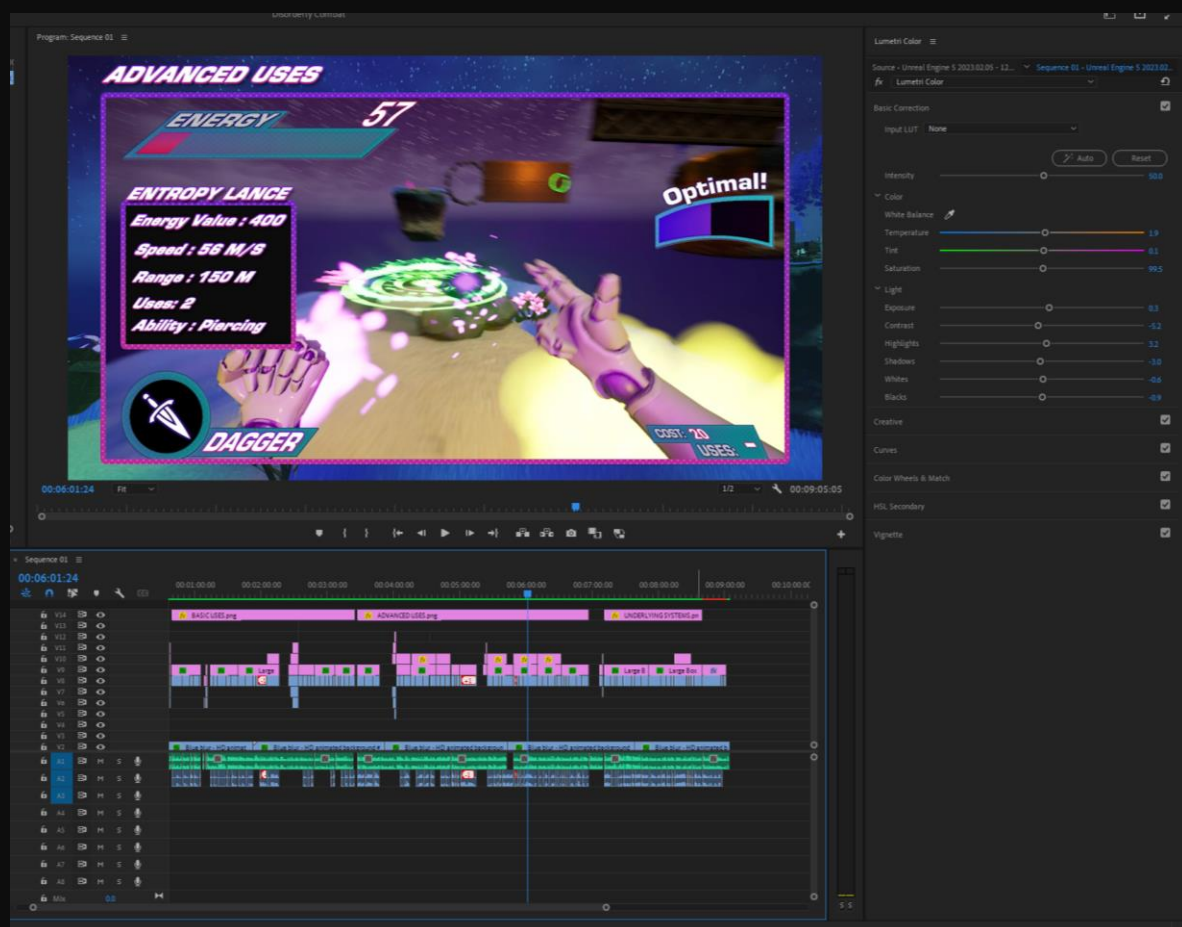
- Underlying system overview:
 - Details: introducing the two underlying systems that relate with the entropy system. One major and one minor.
 - Clips: General gameplay clips
- Minor System: Energy Deterioration:
 - Details: How it works, what its connected to (absorbing), why this feature is important
 - Clips: Video showing the player gaining energy at different rates. (20 energy, holding absorb vs tapping absorb)
- Major System: Discord Mechanic:
 - Details: What is this mechanic, how to build discord, different states and those states effects.
 - Clips: Showing the player build discord, and its behaviour.

Conclusion (progress showcase)

This section will close out the video. I will be showing the actual design of the game. Showing the GDD, Numeric Design, Level Design (Whitebox) and a general timelapse of development.

Showcase Video: Production

After planning the structure of the video, I created a script that contained the everything that I wanted to say during the showcase video. Once the script was finished, I used Audacity to record my speech as well as use some effects to clean up the audio, so I sounded clearer, removing microphone buzz for instance. The next step was to collect footage from my built prototype, this step took several hours as I needed to capture specific gameplay scenarios to discuss in my showcase video. I then used Adobe Premiere Pro to edit and cut the video together, while using Adobe Photoshop to create some nice text elements for the video. Overall, creating the video was the most taxing part of this project, just due to time, but I am extremely happy with my video as it showcases everything I've worked on so far in a stylish and professional manner. With the prototype and the showcase video done, there isn't much left to do for this project. It's now time to close off this document.



Project Evaluation

With the project ending soon, this section will serve as a retrospective for the project. I will be asking myself questions to generate some helpful dialogue regarding this project.

How would you describe the design process of the Entropy System?

Honestly, designing the entropy system has been a treat and I say that with the knowledge that my prototype feels good to play. Pre-production started off immediately strong as I was coming up with a ton of ideas. Initially, the core mechanic I was going to design was a system where you could control light to build structures. The genre would be a first-person stealth game akin to Dishonoured. Comparing the game, I have now to the very initial concept, it feels like night and day. The production stage was the most full-on process of this project as I wanted to flex my "Design" muscles and see if I could really push the concept I created to higher planes. I managed to design an interesting core mechanic, some varied enemies/weapons, an entire level using the core mechanic as a basis for the design and the UI elements to display the core information. Next came prototyping and that was an interesting experience. Translating my design into hard code was far simpler than I assumed it would be. Finally, the most important part, the fact that the prototype was engaging to the people who managed to play it. Regarding the question, "Did I make an engaging combat system within a first-person shooter?", the answer, in my heart, is yes!

Did you identify any problems with the design or the project in general?

One thing that I encountered quite early is the problem of passive play. Since my mechanic promotes energy conservation, this can lead players down the road off avoiding aggressive play, which is not what I wanted for a mechanic such as this. The goal of this system is to get the player to be proactive and to rush into danger, so I needed to solve this issue by week 3 before I start implementing stuff into Unreal 5. To remedy this issue, I designed the Discord system, which rewarded the player for aggressive play. Around week 6, I created another mechanic to further reduce this issue, this mechanic was the Energy Deterioration Mechanic. Overall, most of the design issues stemmed from the fact that at times the game played to player to play too defensively. Since one of my design pillars was aggression, I deemed anything that broke this pillar as a problem.

What would you do differently if you had more time?

If I had more time, I would have implemented the slide movement option, since I already calculated the numeric for this system. I would've been interesting to design another level around the sliding. Another element I would tackle is Entropy Weapons from the obstacles. I think I'm really disappointed about is not being able to implement the Entropy Weapon for the Pristine Launcher or the Pristine Spikes. Other than features, I feel like I organised and handled this project well, so I wouldn't do it differently.

What did you learn during the project?

Unreal 5, credits roll. In all seriousness, I feel like I've picked up an invaluable skill over the course of this project. I essentially taught myself a new game engine for two months. Other than game engines, I feel like this project has given me the chance to flesh out my design skills as before this project, I would've never attempted a project of this size.

Final Thoughts?

Whoever is reading this, thank you, this was extremely long winded, but hey, I developed prototype from pre-production to publishing. This project was extremely fun and I'll miss typing away in this document and I'm sure the passion will shine through.

Asset References

- Fantasy Forest – Magic Forest: <https://www.unrealengine.com/marketplace/en-US/product/stylized-fantasy-forest-magical-forest-stylized-forest?sessionInvalidated=true>
- Infinity Blade Weapons: <https://www.unrealengine.com/marketplace/en-US/product/infinity-blade-weapons>
- Niagara Magic Projectiles: <https://www.unrealengine.com/marketplace/en-US/product/niagara-magic-projectiles-sounds-included>
- Wizard for Battle Polyart: <https://www.unrealengine.com/marketplace/en-US/product/battle-wizard-polyart>
- Stylised Dynamic Nature: <https://www.unrealengine.com/marketplace/en-US/product/stylized-dynamic-nature>