Demo Prototype V.1

Summary Pitch: The player must escape a claustrophobic arena occupied by squad oriented combatants. Rules of engagement: eliminate known threats. That goes both ways.

2.5 D Pixelated shooter Arguments for:

- **Worry only about x and z, not y (verticality)**
- **Not demanding of processor**
- **Visual flare of entire game map**

Demo Prototype Objective: This demo implements the core gameplay mechanics of the Game and tests that they are working co-dependently during runtime. This demo version primarily focuses on the mechanics and has few animations besides placeholders. Art including pixelated sprites, audio, and soundtrack will be postponed for the official demo release. The core principle designs of this demo prototype are to be transferred to the official demo.

Bare Minimum:

- GameManager, handles things in the aGameScene like the spawner system.
- Weapons and a health system. Have working pistols and support for automatic weapons
    - Ammo Capacity - necessity.
    - Bonus (Melee).
    - Bonus PickUp weapons and Drop weapons.
    - Bonus Throw weapons.

- [NPC behaviors - still work in progress]
- Dialogue system, for Agents to bark what exactly they're doing. [Middle Priority]

- Need a Basic Level with a start and end goal.
- Backtrack music (some audio). **Least of our priorities for now.**
    - Bare Necessity, find some good soundtrack with free licensing sample online
- **Sound effects (Shooting and Walking)**
    - Shooting sound effect
    - And walking sound effect

[Capsules, move the gun models in front of the forward direction or pivot of the Characters]

- Simple Animations [Need to be able to billboard sprites toward the camera]. For the sprites, we might be able to pull something from a CodeMonkey video.
    - [Bonus] Death Animation, cheap alternative make the Enemy sprite disappear
    - Walking
    - 4 directional - bare minimum necessity [Hopefully whatever is found on asset store]

    - Bonus: 8 directional movement.
    -
- **Simple VFX effects and Shader.**
    - [Bonus] Upon death perhaps censorship shader like in News Videos
- UI Pause menu and beginning start menu [Bare Minimum, just gets the job done].
- **Also we need to present how the Player can use the controls. Put it in the pause menu.**

Have so far:

- **Weapons system and Health system**
    - **Still need to apply Health System to NPCs of GOAP architecture**
- **Thirdperson camera**
- **Prototype Level**
    - **Serve its purpose for testing.**
- **Squad NPC Behavior**
- **UI for slow Motion**
- **Implementing SLow Motion [Combining our three parts so far]**

Scheduling:

- Ignoring sprites, rather we will uses simple capsules of different colors to indicate enemies and the Player, as well as using free VFX effects for indications.

1. Sound effects for weapons and Agents and Player walking.
2. VFX effects. Muzzle flashes when guns shoot, and dim lighting for the level, to make guns firing pop more, and blood splatter that pops out when a shot hits.
3. Fun Prototype Level primarily focused on combat. Some days we work on this together.
4. More functionality to Player Controller
    a. Player Health and a UI element for it.
    b. Ammo Capacity - necessity.
    c. Bonus (Melee).
    d. Bonus PickUp weapons and Drop weapons.
    e. Bonus Throw weapons.
5. Start Menu and Pause Menu.
6. Dialogue for NPCs, some barking system[Worry about at end]
7. Sprites [Highly doubt]

Assignments:

Assigned 8/31

- Tyler working on Player Health and blood splatters
- Vlad is assigned to sound effects
- Ryan help with VFX, and also works on Tyler's prototype level.

Core Gameplay Mechanics

GameManager:
Presumably the manager will handle the following:

- Rebaking NavMesh areas in the GameScene to have different costs, depending on the traversal of bullets including both players and other agents.
- What AI agents are alive or dead in the GameScene. Upon exiting a GameScene, all dead NPC bodies are to be recycled!
-
-

AI Behavior:

- Squad behavior Manager:
- Independent behavior driven AI agents:
- Every defined goal should only have three actions in their subsequent plan maximum.

    - When an agent dies, the action that its running will fail and its current actionState is to exit and set to null, and in no circumstance upon death should the agent call to calculate a new goal which will cause another job request to the planner to calculate a new goal and subsequent plan. Note, the agent has goals that complete every Update() interval to tell the agent to compare their priorities to the current goal's priority in order for the agent to call to calculate a new goal.
    - The following must happen upon DEATH:
        - CalculateNewGoal() must NOT happen upon action failure. Can call to remove components of type IReGoapAgent, but first get rid of all components that reference IReGoapAgent in the Agent Prefab Game Object.
        - Goals must not compete during Update() in ReGoapGoalAdvanced to have the agent compare their priorities to the current goal. Perhaps we can call OnDestroy() on all Goal components in the agent to recycle them in a stack which acts as a cache. Or remove all Game components of type IReGoapGoal.
        - There is a visual indication of a corpse.
    - The agent has the following defined goals and actions:

Goals	Actions

    - Agent has the following sensors:
    -
- hh

Weapon System:

Every weapon can be picked up and equipped (E or F keys) by either the player or enemies. Upon death, the weapon is dropped.

Bullets can penetrate certain objects that will disperse particle effects. Bullets will have vapor trails following them, and in real time bullets look like exaggerated tracer rounds. If slow motion bullet time is implemented, then the player is able to see the vapor trails before they disappear. Bullets CANNOT penetrate arena walls,

All guns when fired light up the area within proximity including the character(s).

If either the playable character or enemy combatants is hit with a bullet, then they die immediately upon impact.

All guns have finite ammo and there is no reloading, except enemy combatants may reload. Delays between using the weapon depends on the weapon.

There are three main weapons in the game:

- Pistols: pistols release a single bullet whenever used and are highly accurate to hit toward the direction of the mouse-cursor which serves as the cross hair. If hit by a projectile by a pistol, then two types of death animation will happen:
  - Center mass death: the body of the NPC is hit, and there will be no censor. This has a high probability upon death
  - Headshot death: the head of the NPC is popped, and there is a sensor filter on the head. This has a lesser probability upon death.
- Shotguns: releases multiple large pellets that disperse more as they travel farther from origin. Mimic the effect of buckshot to demonstrate that this is the most devastating weapon in the game.
  - If hit by a projectile from a shotgun, then the entire upper torso is a bloody mess with a censor filter covering the upper body. This has a 100% probability upon death.
- Blade: Default weapon if the player has no gun equipped. Has a smaller range of attack. Upon death, the NPC's upper torso is hacked from left shoulder to middle of lower abdomen if hit from the front. If hit from the back, right shoulder to middle of back side is hacked. Any other NPC position, the neck is slit.
  - Blade resets to initial position in reference to the player after use and then ready to be used again.

All weapons are throwable, including the blade by using the right-click mouse button. To use the weapon click the left-click mouse button.

Positioning the gun to the cursor. Instantiate a bullet asset and (Projectile system or raycast)
Bullet collides with enemy or player, then apply damage.

Level Design:

Compact arena, can probably copy a map from Valorant or CSGO for starters and scale it down. The player should be able to analyze and strategize immediately upon entering the arena. Arena must be open-ended, simple, but not too simple.

The arena will have few open areas, meanwhile, alleyways and corridors will have some shootable objects that will disperse particles. Walls when hit with bullets will disperse particle effects. Also, walls should have some destructibility to them with indents from the bullets, but of course this being a 2.5D shooter mimicking the same camera from Enter the Gungeon, only the walls facing the camera matter. There may be a lack of depth with this static camera angle, so it's likely not worth it to have destructible wall textures, but just a texture overlay of bullet indents from the Unity asset store.

Lighting in the arena will have more areas that are dimly lit compared to brighter areas. This is to emphasize the feeling of gunplay which upon shooting will light up the proximity area of the gun. There is a balance of the lighting though, as to not obscure the arena, and allow the player to see the visual eye candy.

Simple Animations:

Utilize a Particle System already created! (Visual Effects Graph e.g.)

Create pixel art VFX effects for the guns when shooting and when a bullet collides with walls and characters. Additionally, there needs to be pixel art VFX effects for certain penetrable GameObjects that disperse particles upon being hit. Most importantly, there needs to be a VFX Effect for blood splatters from entry wounds. And there needs to be a way to apply a censor filter onto parts of corpses.

https://www.youtube.com/watch?v=JZDlCuIpq9I

Pixelated Animations:

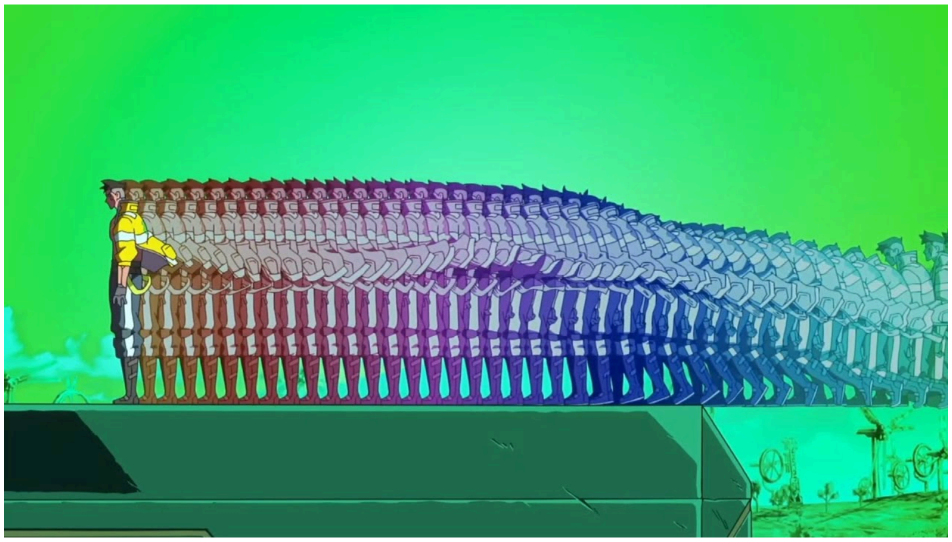
- Upon death, if the character were shot by a gun, then the character will fall in the direction the projectile traversed, but if killed by a blade they will perform the same death animation (could be varied based on how the character is positioned).
    - Death upon by shotgun, special censored death animation
    - Death upon by pistol, high probability for default death by pistol and lesser probability for censored head popping death animation.
- Cover: Just hide character behind world geometry, no special animation required
- Cover Blindfire: Have the gun extend from character to blind fire. May need to create an arm extension pixelated example.
- Dodge: Dodge animation with invisibility frames
- Shoot animation for the gun (some recoil) and lighting effects
- Guns are able to move 360 degrees around characters.
- Slash effect when player is using a blade

Player:

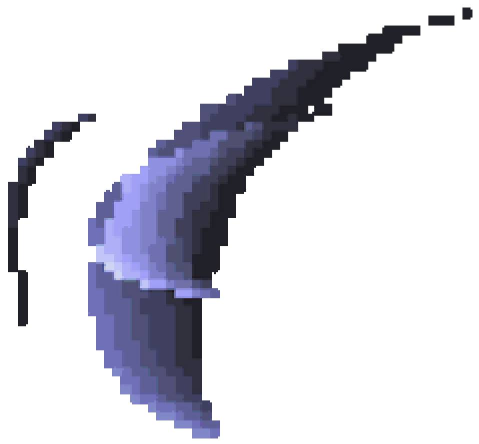
- Smooth controller that handles input effectively.
- Aim wherever the cursor is on the screen. The mouse cursor dictates the direction of the player.
- Handle Input:
    - Left click mouse, then shoot equipped gun.
    - Move in WASD directions
    - Sprint or (Bungie Halo, keep a constant speed).
    - Pick Up weapons (E or F) No Reloads and finite ammo
    - Default weapon (Machete or Katana)
    - Throw guns (stuns enemies)
    - Standard slow mo (without the flare). When slow mo is enacted (GameScene color is changed.Similar to COD, but more appealing, but indicative that you're in slow mo. When slow motion is enacted, then every NPC will be slower than the player. Slow mo will fill up overtime.
-

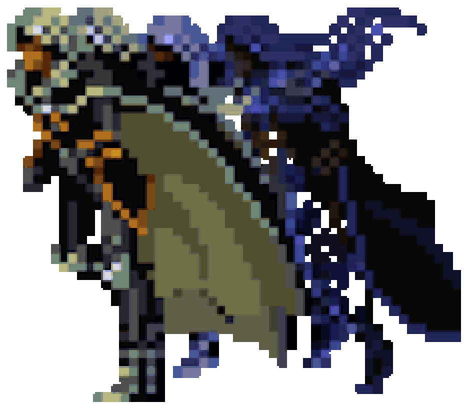
Feature Creep:

The following are additional gameplay mechanics that deviate from or extend the core gameplay mechanics. The central objective of the Prototype is to ensure that all the listed core gameplay mechanics properly function in the Game Scene with little to no bugs.

- Shootable objects that output particles to the GameScene
- Slow Motion/Bullet Time:
    - Meter: If the meter is full, the entire game scene stops in time, except for the player.
    - 
      The trail effect disappears one at a time after the slow motion ends. And if enemies were killed during time stop, then the death animations will delay by one second before enacting. (You're already dead feeling).

If the slow motion meter is not full, then enter standard bullet time where everything is slowed down and the trail effect is not as large in which only a certain set of repeated animations can stay, then disappear.
Somehow need to implement this after-image effect





- Throwable weapons: Very similar to Hotline Miami
- Telekinetic Blade: Blade that can be thrown and cleaves through AI agents and impales onto collision objects. The blade, upon mouse click, will travel in the opposite direction the player faces upon mouse-click, until it hits/impales another collision object/obstacle, or is caught by the player.