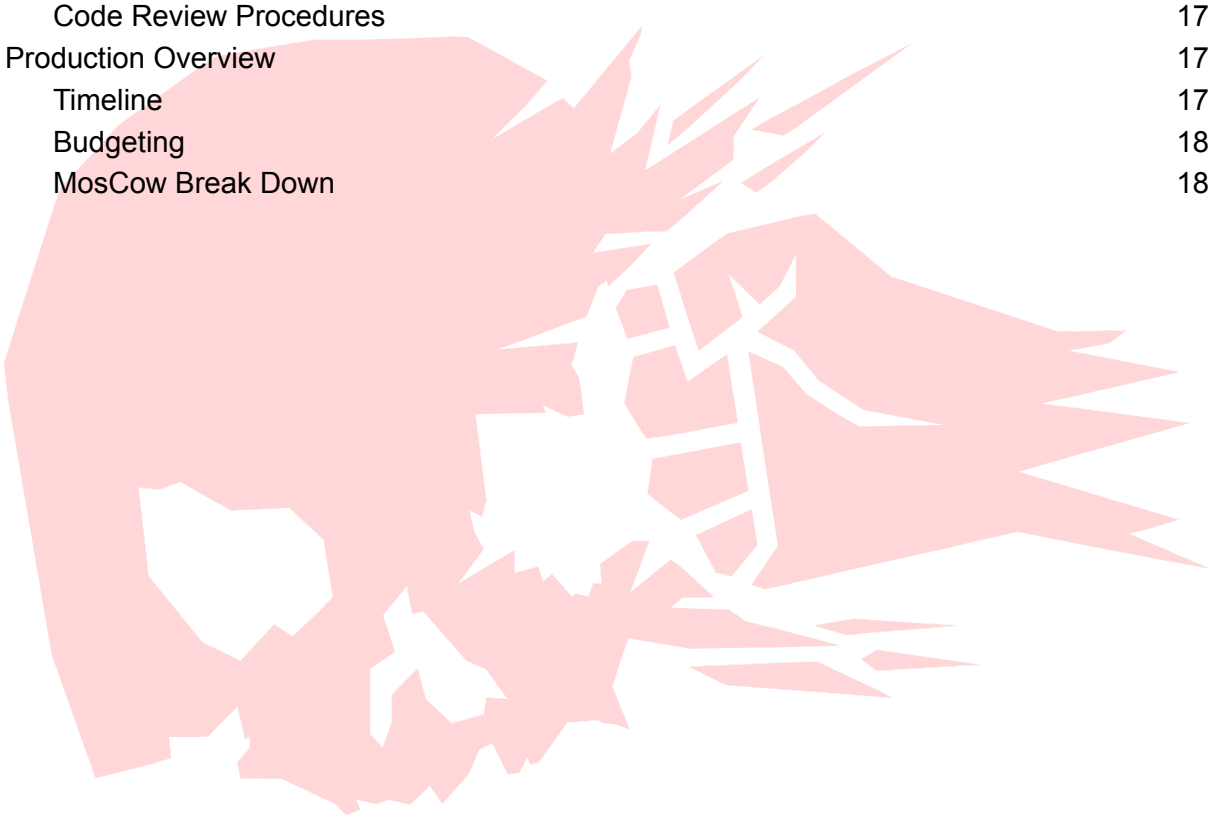
Game Design Document - BloodDrunk

Better Than Nothin' Project

"Room to Room Killer"

Table Of Contents

Contents



Description – USP

Dive into the relentless action of BloodDrunk, a VR arcade shooter where your survival hinges on a constant stream of kills. If you stop killing, you stop living. Navigate through an endless sequence of rooms, each presenting new environments, challenging encounters, and formidable enemies. How long can you keep up the carnage?

Goals Of Project

Our mission is to develop and launch a new VR title on the Steam market under the name BloodDrunk. To achieve this, we have outlined the following core goals:

- **Create an Immersive VR Shooter**: Develop BloodDrunk as an engaging and intense VR arcade shooter where players must keep getting kills to survive.
- **Enhance Proficiency in Unreal Engine**: Improve our team's understanding and skills in using Unreal Engine to deliver a high-quality gaming experience.
- **Successfully Launch on Steam**: Ensure a smooth and successful release of BloodDrunk on the Steam platform, reaching a wide audience of VR gamers.
- **Deliver an Enjoyable VR Experience**: Focus on creating a thrilling and enjoyable experience for players, with diverse environments, encounters, and enemies to keep the gameplay fresh and exciting.

Potential Titles

Potential Titles for Our VR Shooter

- Redmist
- BloodDrunk (current working title)
- Paint It Red

As of now, the game is being developed under the title BloodDrunk. However, we are open to considering a more fitting title if one is discovered.

Animation

Animation for the game will include relatively simplistic animations, this will be for the enemy characters in the game. Due to the team's lack of an animator we will need to be cautious to not overscope the animation for the project.

Sound And Music

For the audio experience in BloodDrunk, we aim to blend arcade-style sound effects with the intensity of death metal music. We are exploring a collaboration with the band "Denied Legacy" to provide the soundtrack, ensuring a thrilling and immersive audio experience that complements the high-energy gameplay.

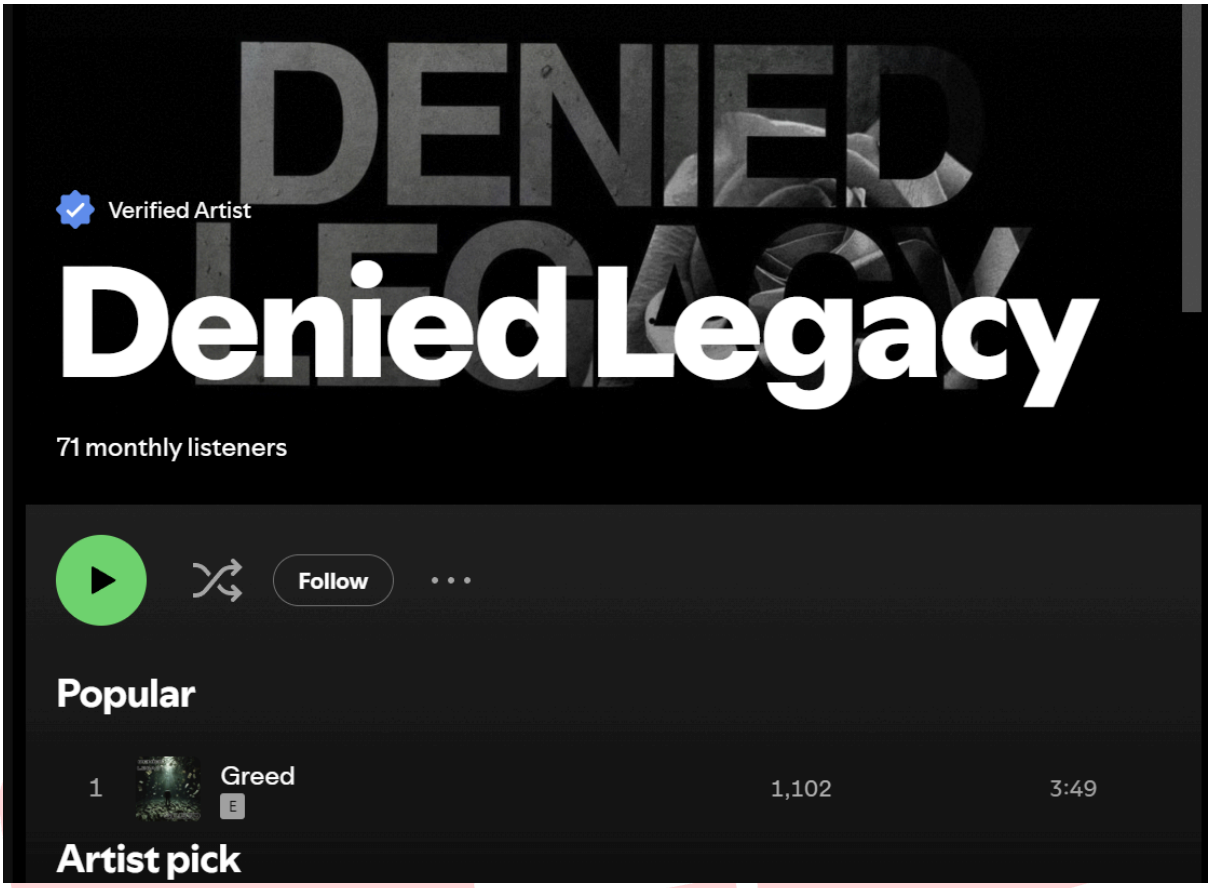


Figure X, Music Reference

Overview Design Document

Core Gameplay loop breakdown

Mechanic Breakdowns

Shot lines
Enemies' attacks are telegraphed using rays, spatial indicators that indicate when and where the attack will occur. Player needs to avoid intersecting the line to not take damage

Teleportation
Killing an enemy allows the player to teleport to the location at which it died, which doubles as further motivation to kill outside of preserving their own life. Useful in a pinch, but players must also take note of enemies that die in dangerous places (e.g., over a pit).

Blood Shield
Occasionally the player can use a blood shield to block incoming projectiles aimed at the player. This would be a handheld shield.

Weapon Possession
Player can grab weapons and turn them into his own weapon to shoot, there are different types of weapons that behave differently. Examples include:

-Pistol
-Shotgun
-Sniper
-Minigun

Blood Smear
Players can get covered in blood on the visor from combat, for the player to remove they must wipe the blood off with their hand.

Punch
Should enemies get within punching distance the player can punch the enemies to defeat them.

BloodDrunk
Players has a BloodDrunk bar that progressively goes down over time, for this to be updated the player must get kills, Should this meter hit 0, the player dies forcing them to restart

Rage
Should the player get enough kills the player can unleash a rage mechanic, temporarily the player is immune to damage and can teleport anywhere on the map. This lasts a set amount of time before the player needs to earn it again.

Inspiration

BloodDrunk draws its inspiration from several iconic sources to create a unique and exhilarating VR experience:

- **Superhot**: Incorporating fast-paced gameplay and arcade-style mechanics, creating an intense and immersive shooter experience.
- **Ape Out**: Emulating the chaotic and relentless need for constant movement, adding to the game's dynamic and frantic atmosphere.
- **John Wick**: Drawing from its stylish and precise shooter mechanics, delivering a polished and action-packed combat experience.

These influences come together to shape BloodDrunk into a thrilling VR arcade shooter that demands quick reflexes, constant action, and strategic gameplay.

Features

Player Motivation

The primary motivation for players in BloodDrunk is to achieve the highest score possible in an endless shooter format. Key aspects driving player engagement include:

- **High Score Competition**: Players aim to top the leaderboards, fostering a competitive spirit and encouraging repeated playthroughs to beat their own or others' scores.
- **Endless Challenges**: The game's endless nature means there is always another room and another wave of enemies
- **Diverse Environments and Encounters**: Each new room offers a unique environment and set of challenges, adding an element of exploration and discovery that keeps the gameplay fresh and exciting.

By combining these elements, BloodDrunk ensures players are constantly motivated to push their limits, explore new strategies, and master the game's mechanics in their quest for the highest score.

Mood

Mood for the game is kept relatively light with it being an arcade game with elements of simple violence in the game, with the game being aimed for 16 up it is important to make sure it is rated for that genre.

Plot

The main character is the last member of a powerful and rich family, who locked himself in his giant manor in the middle of nowhere (maybe literally) because of a curse ridden upon him. He stayed there in a weakened state for a 100 years (probably not but you get the idea), but now he is awakened because of a cult/rival family/mercs (pick one) and now he can't contain his desire to kill.

Levels / Areas

Technical Design Document

Hardware Requirements

Blooddrunk Minimum/Recommended Hardware Requirements	
GPU (Graphics Card)	Nvidia 2060 RTX
CPU (Central Processing Unit)	Core I5-7600
Dedicated RAM	8 GB
Software	Windows 10/11

Figure X, Minimum Hardware Requirements

Platforms

Engine Specific Specifications and Limitations

Engine Summary

Unreal Engine, developed by Epic Games, is a game engine known for its advanced graphics capabilities, including photorealistic rendering features such as dynamic lighting, complex material, and realistic effects like reflections and shadows. It has content creation tools, including a powerful level editor, animation system, material editor, and particle system, streamlines the development process, while its Blueprint visual scripting system enables developers to create complex game logic without traditional coding. Moreover, Unreal Engine's cross-platform support for PC, console (PlayStation, Xbox), mobile (iOS, Android), and virtual reality (Oculus Rift, HTC Vive) ensures broad accessibility, while its active community, frequent updates, and strong technical support further solidify its position as a top choice for developers aiming to create immersive and visually stunning gaming experiences.

Systems and Diagrams

Player

Health

The health of the player is indicated by their blood bar, i.e. the timer which is represented by a blood meter. It starts with 15 seconds. If the blood bar reaches zero, the player dies and the game ends.

Movement

The movement of the player is based on fixed positions, which are represented by blood. These positions are set for each level and are represented by a specific opponent model.

Spawn

The player is spawned in a certain area which is specified by the stage.

Shooting

The player can hold two weapons (see ‘Weapons’), one in each hand. When a weapon is empty, it gets dropped to the ground. The player can always use his pistol. It is accessible through the holster.

Blooddrunk mechanics

The player can use skills (see ‘Skills’) based on their cooldown.

Enemies

Spawn

In the first version the enemies are spawned randomly (TODO later changed with ‘the concept of difficulty budgeting’) at certain spawn points. The game keeps a certain number of enemies on the screen, i.e. 4, depending on how many enemies are left in the area.

Shooting

The enemies are equipped with different weapons (see ‘Weapons’). These create bullet lines based on their weapon types which the player has to dodge. When a shot hits the player, it depletes the bloodbar.

Health

The enemies have one health and they die if they are hit. They can wear armor, which can withstand a certain amount of damage before being shot off.

Movement

When the player is at a certain point the enemies are coming into combat from their spawn points and go to their positions.

Dying

When an enemy dies, it drops its weapon, which the player can pick up through the room.

Stage generation

Layout

A stage has three different areas. An area consists of obstacles, i.e. cover, the enemies and a field in which the player is standing. A certain enemy deploys a blood spot, where the player has to move into. By this, the player can traverse through the stage.

Enemies

The number and types of enemies which are spawned are based on the difficulty (see ‘Difficulty’)

Obstacles

The obstacles are placed randomly on certain points, which is specified by the layout.

Start

When everything is set up, a 3 second counter tells the player when the game is about to start.

Weapons

Player View

Name	Pistol
Firemode	Single
Magazine-Size	12
Damage	1
Bullet-Number	1
Fire Rate	0.4
Reload Speed	3
Weapon Range	3000
Weapon Spread	2
Can Reload	Yes
Penetrates Armour	No
Has Laser Pointer	Yes

Name	Shotgun
Firemode	Single
Magazine-Size	2
Damage	1
Bullet-Number	8
Fire Rate	3
Reload Speed	2
Weapon Range	1000
Weapon Spread	20
Can Reload	Yes
Penetrates Armour	No
Has Laser Pointer	No

Name	Uzi
Firemode	Automatic
Magazine-Size	20
Damage	1
Bullet-Number	1
Fire Rate	0.0675
Reload Speed	2
Weapon Range	1750
Weapon Spread	4
Can Reload	Yes
Penetrates Armour	No
Has Laser Pointer	Yes

Name	Deagle
Firemode	Single

Magazine-Size	9
Damage	1
Bullet-Number	1
Fire Rate	0.7
Reload Speed	0 TODO
Weapon Range	4000
Weapon Spread	0 TODO
Can Reload	Yes
Penetrates Armour	Yes
Has Laser Pointer	Yes

Name	Grenade Launcher
Firemode	Single
Magazine-Size	1
Damage	1
Bullet-Number	1
Fire Rate	0.4
Reload Speed	0
Weapon Range	0 TODO
Weapon Spread	0
Can Reload	False
Penetrates Armour	Yes
Has Laser Pointer	Yes

Enemy View

Name	Pistol
Enemy Type	Pistol Enemy

Walkspeed	150
Weapon Spread	0
Blood points	75
Weapon Damage	-20
Time to Shoot	3

Name	Shotgun
Enemy Type	Shotgun Enemy
Walkspeed	150
Weapon Spread	20
Blood points	75
Weapon Damage	-20
Time to Shoot	5

Name	Teleporter
Enemy Type	Teleport Enemy
Walkspeed	0
Weapon Spread	0
Blood points	100
Weapon Damage	-20
Time to Shoot	0

Skills

Name	Shield
Ability	Shields the player from damage
Cooldown	15 seconds
Process	The shield is up for 10 seconds or for 3 hits

Optimisation and Profiling

Profiling Systems

For this project, the profiling methodology follows these steps:

1. **Initial Testing**: Load the project and test the function or feature to check if it produces the intended result. Identify any bugs or issues.
2. **Review in Blueprint**: If issues arise, review the blueprint to identify any simple fixes that can be made.
3. **Breakpoint Analysis**: If the problem persists, use breakpoints to debug. Place breakpoints and examine each node to ensure the chain of events is functioning correctly. Investigate if any events fail to trigger.
4. **Print String Debugging**: If breakpoints do not resolve the issue, use print strings to gain insight into what may not be working as intended.
5. **Rubber Duckie Approach**: If troubleshooting is still unsuccessful, employ the "Rubber Duckie" approach. Explain the problem aloud or in writing to an imaginary or real rubber duck to clarify and potentially find a solution.
6. **Blueprint Debugger**: As a last resort, utilize the Blueprint Debugger to gather additional information necessary for the project.
7. **Online Research and Consultation**: If all other methods fail, search online for potential fixes or seek assistance from lecturers or peers to resolve the issue.

Profiling Graphics

Graphics testing for the project will be conducted at a later stage to ensure a consistent FPS (frames per second) in the game. Specific profiling will also be performed and explained in detail in the upcoming weeks.

Coding Standards

This project will not compromise on coding standards, ensuring they are upheld to the highest level. In maintaining this commitment, the team will prioritize readability and appropriateness in their methods. This includes minimizing the use of event ticks, spawning assets only when necessary, and keeping sections modular. Additionally, a strict naming convention will be implemented to ensure consistency and enhance the understandability of variables and code throughout the project.

Programming Standards

When conducting blueprint work, the team aimed to incorporate multiple methods to maintain a high standard of code. This involved thoroughly commenting on work and annotating smaller sections of code to facilitate understanding for any reader. Additionally, a strict naming convention will be applied to all items created for this project by the team,

ensuring consistency and clarity. These conventions were adhered to in programming to uphold organization, consistency, and readability within the codebase.

Naming Conventions

Variable names camel cased - myVariable

Class Variables/Member Variables – m_variable

Global Variables (instanced variables) – g_variable

Class Name	Class Prefix
Blueprint	BP_
Material	MM_
Material Instance	MI_
Texture	T_
Game Mode	GM_
Game Instance	GI_
Game State	GS_
Player State	PS_
Player Controller	PC_
Animation	Anim_
Animation Blendspace	BS_
Animation Blendspace 1D	AnimBS1D_
Animation Blueprint	AnimBP_
Animation Montage	AnimMon_
Master	Mast_
Child	Child_
Blueprint Interface	BPI_
Widget Blueprint	UI_
Function Library	FL_

Structure	Struct_
Data Table	DT_
Enumeration	Enum_
WAV Audio File	WAV_
Sound Cue	Cue_
Skeletal Mesh	SK_
Static Mesh	SM_
Physics Asset	PA_
Particle (Niagara)	FX_
Emitter	Emit_
Editor Utility Widget	EUW_
Editor Utility Blueprint	EUBP_

Figure X, Suffix Sheet

Style Guide

For the Style Guide for the project plugins with agreement can be used, current tolls used , purely visual aids. These include:

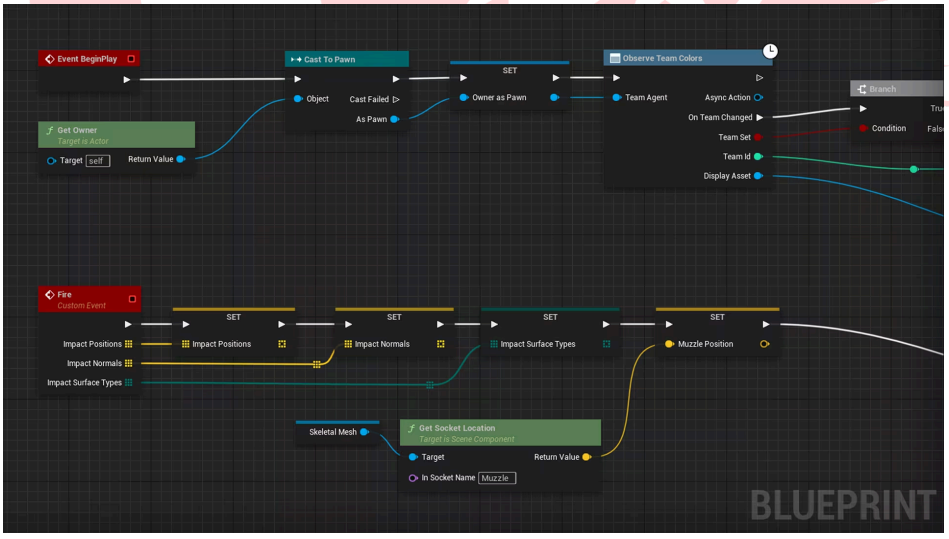


Figure X, Flat Nodes Example

[Flat Nodes Link]

Flat Nodes Plugin - A Visual Scripting addon that changes the appearance of nodes to a flat color, this helps with visual clarity and makes it less confusing.

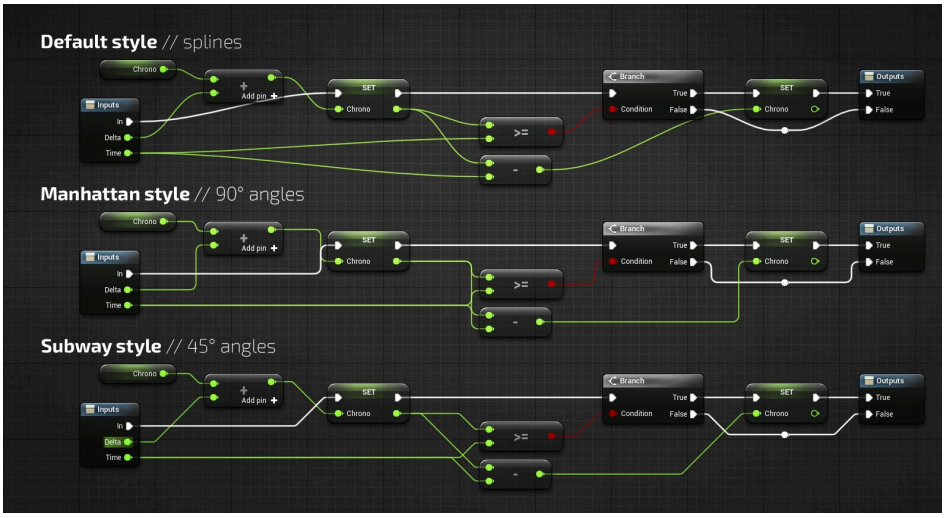


Figure X, Electronic Nodes Example

[Electronic Nodes Link]

Electronic Nodes - A blueprint plugin tool that straightens connection nodes and prevents spaghetti code from occurring, this makes it easier to read code.

Commenting Rules

Throughout the project, the team will ensure that commenting work is conducted extensively. Comments will be broken down into chunks and will provide a detailed description of what each section of code does, rather than just a title. This approach will aid in revisiting and understanding the work, especially if changes are made during the development process. Furthermore, it will enhance readability for other team members who may be reviewing or working on the project.

Code Review Procedures

Regular code review sessions will be conducted and strongly encouraged to ensure the development of a robust framework.

Team members will be urged to engage in open communication with one another, both seeking feedback from others and self-assessing their own work. This collaborative approach will foster an environment where the best possible work can be achieved, promoting continuous improvement and the implementation of best practices.

Production Overview

Timeline

The initial project plans aim to commence in May and extend throughout the summer until our final submission deadline in August. During this period, the team will collaboratively

determine the core goals and subject areas for the project. Over the coming weeks, discussions among team members will inform these decisions, ensuring that all perspectives are considered and contributing to the development of a cohesive and well-defined project plan.

Phase	Timeline	Description
Pre-Development Phase	28.05 - 07.06	Creating the core concept of the game. Creating the first designs.
Milestone #1	10.06 - 05.07	Creating the basic features and combining them to create a working and buildable game. Look into publishing on steam.
Milestone #2	08.07 - 27.07	Adding features, guns, maps, skills and flesh out the game as well as possible.
Milestone #3	29.07 - 09.08.	Featurestop. Fix bugs, polish the game and publish it onto steam as a full release.

Budgeting

This project will incur no financial costs; the only investment will be the hours dedicated to its development. The team aims to allocate approximately two months of work for the initial submitted build. This timeframe ensures sufficient time to implement the project's goals and deliver a high-quality vertical slice without the need for external financial resources.

MosCow Break Down

MoSCoW

Must Have

- Shooting Mechanics with Different Classes
- AI Enemies
- Endless Shooter Gamemode

Should Have

- Procedural Room Generator (New Objects in rooms)
- Sound, Art and VFXs
- Complete Game Loop

Could Have

- Different Game Modes or modifiers
- Further Powerups and abilities

Won’t Have

- Flying car