SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Multiplayer-FPS

Abstract— In the 21st century, the first-person shooter is the most commercially viable video game genre, and in 2016, shooters accounted for over 27% of all video game sales. The aim of this project is to build a simple FPS game with bull body animation and has multiplayer capabilities.

INTRODUCTION

First-person shooter (FPS) is a sub-genre of shooter video games centered on gun and other weapon-based combat in a first-person perspective, with the player experiencing the action through the eyes of the protagonist and control king the player character in a three-dimensional space. The genre shares common traits with other shooter games, and in turn falls under the action game genre. Since the genre's inception, advanced 3D and pseudo-3D graphics have challenged hardware development, and multiplayer gaming has been integral.

The first-person shooter genre has been traced back to Wolfenstein 3D (1992), which has been credited with creating the genre's basic archetype upon which subsequent titles were based on. One such title, and the progenitor of the genre is wider mainstream acceptance and popularity, was Doom (1993), often considered the most influential game in this genre; for some years, the term Doom clone was used to designate this genre due to Doom's influence. Corridor shooter was another common name for the genre in its early years, since processing limitations of the era's hardware meant than most of the action in the games bad to take place in enclosed areas, such as in cramped spaces like corridors and tunnels.

1998's Half-Life--along with its 2004 sequel Half-Life 2--enhanced the narrative and puzzle elements. In 1999, the Half-Life mod Counter-Strike was released and, together with Doom, is perhaps of the most influential first-person shooters. GoldenEye 007, released in 1997, was a landmark first-person shooter for home consoles, while the Halo series heightened the console's commercial and critical appeal as a platform for first-person shooter titles. In the 21st century, the first-person shooter is the most commercially viable video game genre, and in 2016, shooters accounted for over 27% of all video game sales

Software and Hardware Requirements

Minimum requirements	Windows	macOS	Linux (Support in Preview)
Operating system version	Windows 7 (SP1+) and Windows 10, 64-bit versions only.	High Sierra 10.13+	Ubuntu 20.4, Ubuntu 18.04, and CentOS 7
CPU	X64 architecture with SSE2 instruction set support	X64 architecture with SSE2 instruction set support	X64 architecture with SSE2 instruction set support
Graphics API	DX10, DX11, and DX12-capable GPUs	Metal-capable Intel and AMD GPUs	OpenGL 3.2+ or Vulkan-capable, Nvidia and AMD GPUs.
Additional requirements	Hardware vendor officially supported drivers	Apple officially supported drivers	Gnome desktop environment running on top of X11 windowing system, Nvidia official proprietary graphics driver or AMD Mesa graphics driver. Other configuration and user environment as provided stock with the supported distribution (Kernel, Compositor, etc.)
	For all operating systems, the Unity Editor is supported on workstations or laptop form factors, running without emulation, container or compatibility layer.		

Assets used in the game

Character: https://www.mixamo.com/#/?page=1&type=Character

Gun and arm models: https://sketchfab.com/search?q=fps&sort_by=-relevance&type=models

Other assets:

https://assetstore.unity.com/packages/essentials/tutorial-projects/unity-particle-pack-127325

https://assetstore.unity.com/publishers/31837

https://assetstore.unity.com/packages/tools/network/pun-2-free-119922

https://assetstore.unity.com/packages/3d/environments/sci-fi/sci-fi-battery-pack-free-19738

https://assetstore.unity.com/packages/3d/environments/3d-free-modular-kit-85732

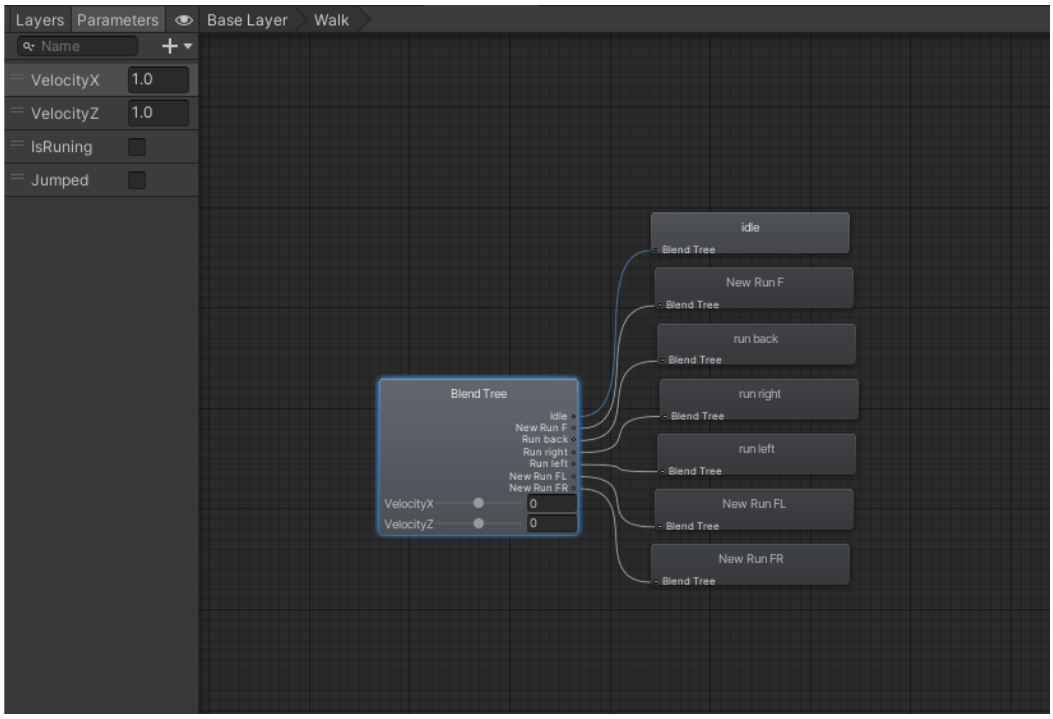
https://assetstore.unity.com/packages/essentials/asset-packs/unity-particle-pack-5-x-73777

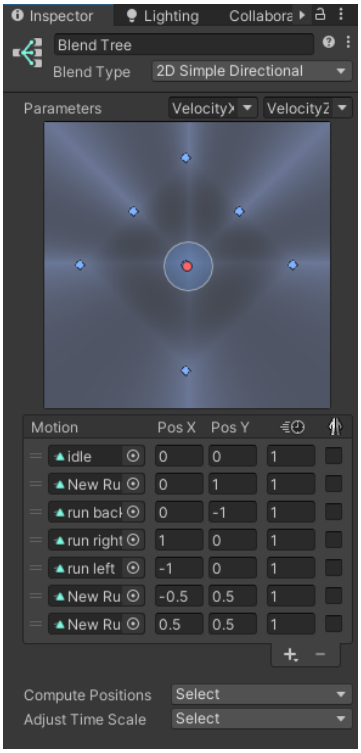
DESIGN PROCEDURE

Fully animated and rigged character:

The character used Unity's Animation Rigging to overwrite the animations for more accurate positioning of the weapon and the rest of the body to look where the player intended to look.

Blend Trees are used to animate the characters movements:

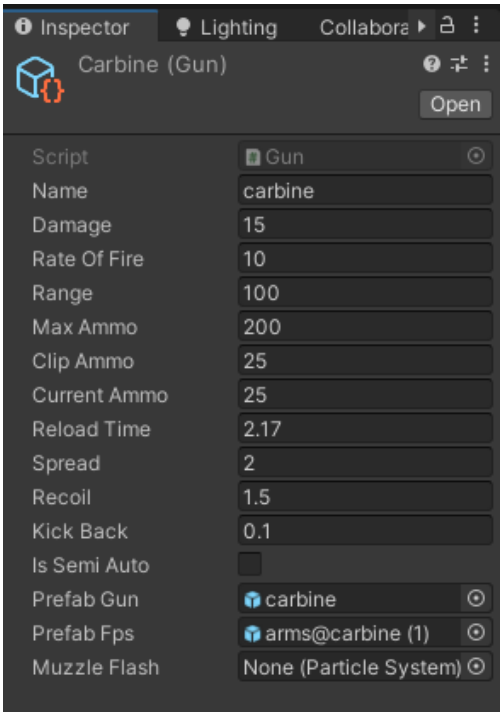




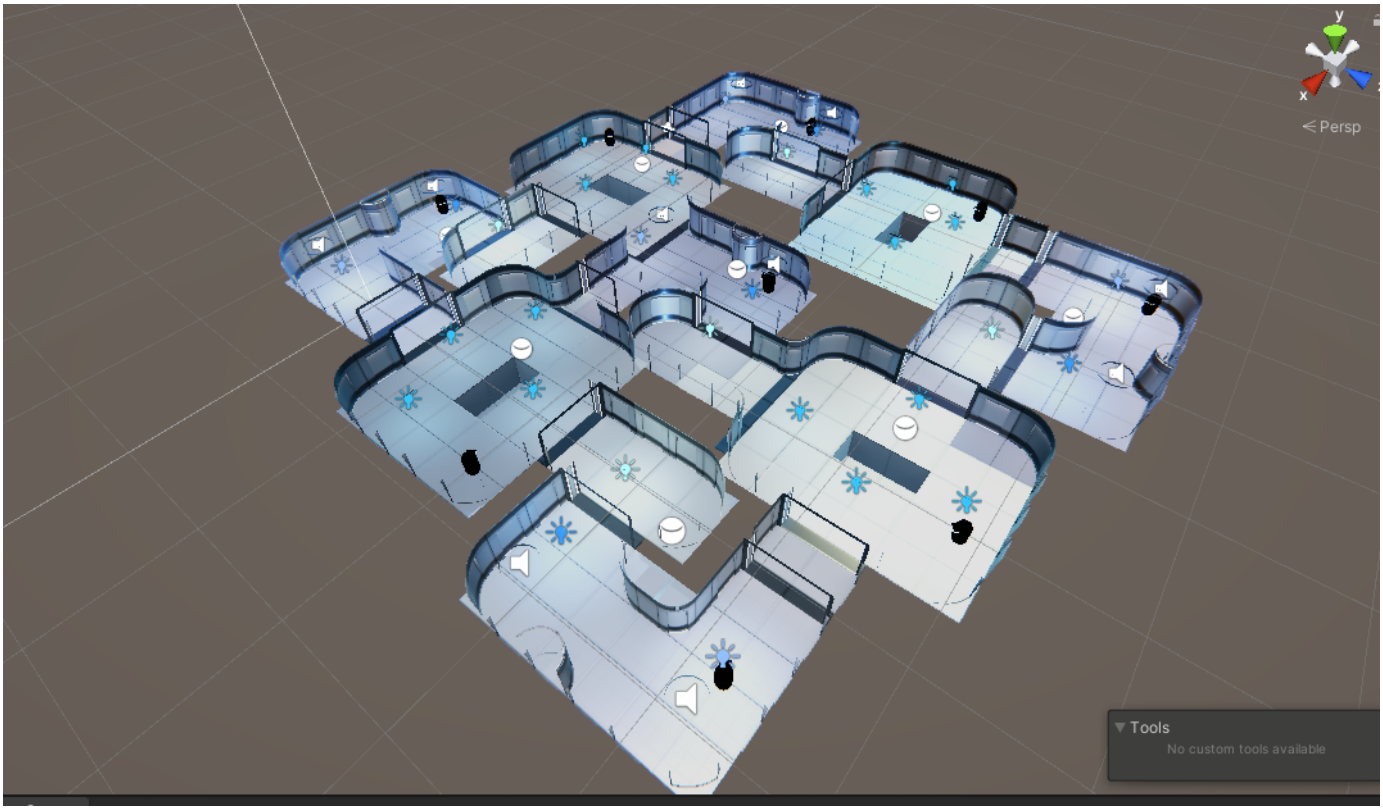
Use of Scriptable objects to implement different guns:

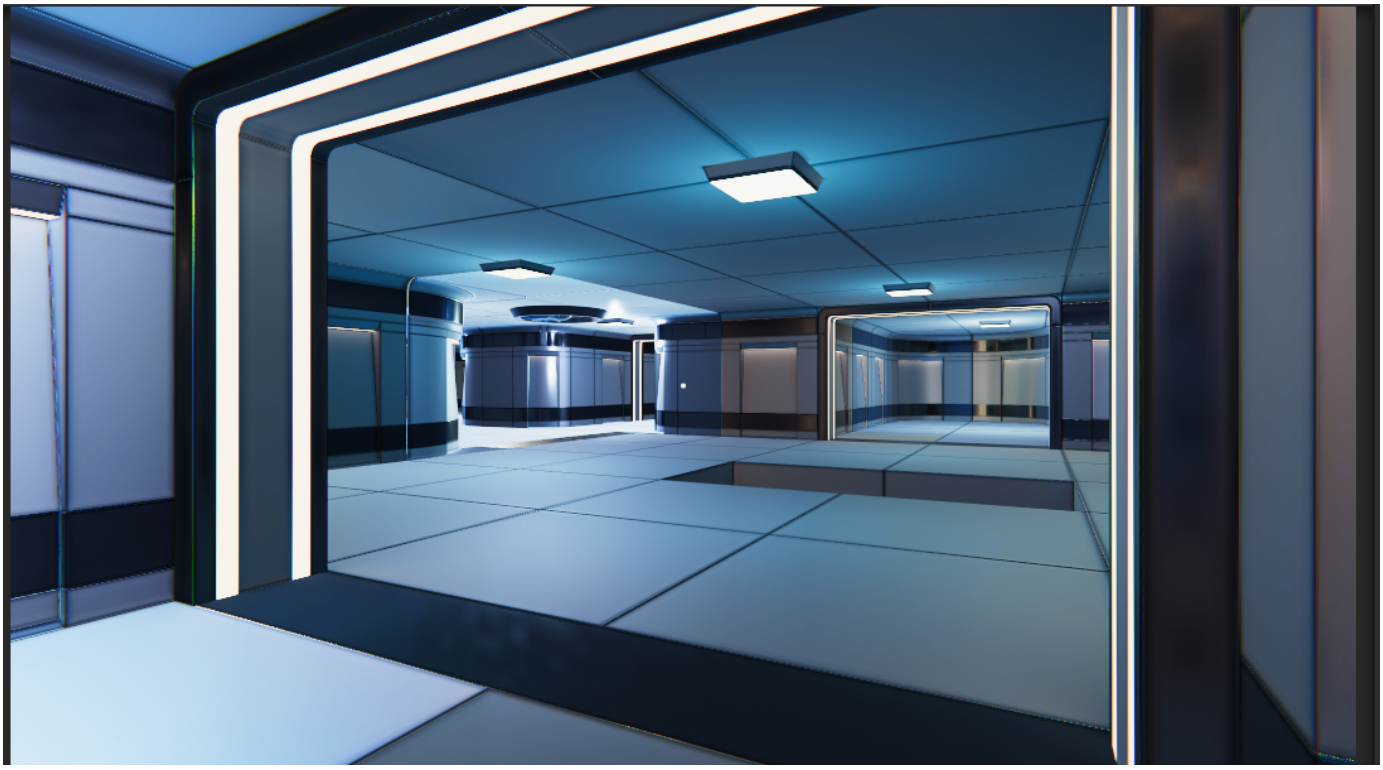
By the use of scriptable objects, it makes it easier to add new weapons and fine-tune their characteristics.

Just by adjusting some variables, you can end up with a weapon that behaves totally differently.

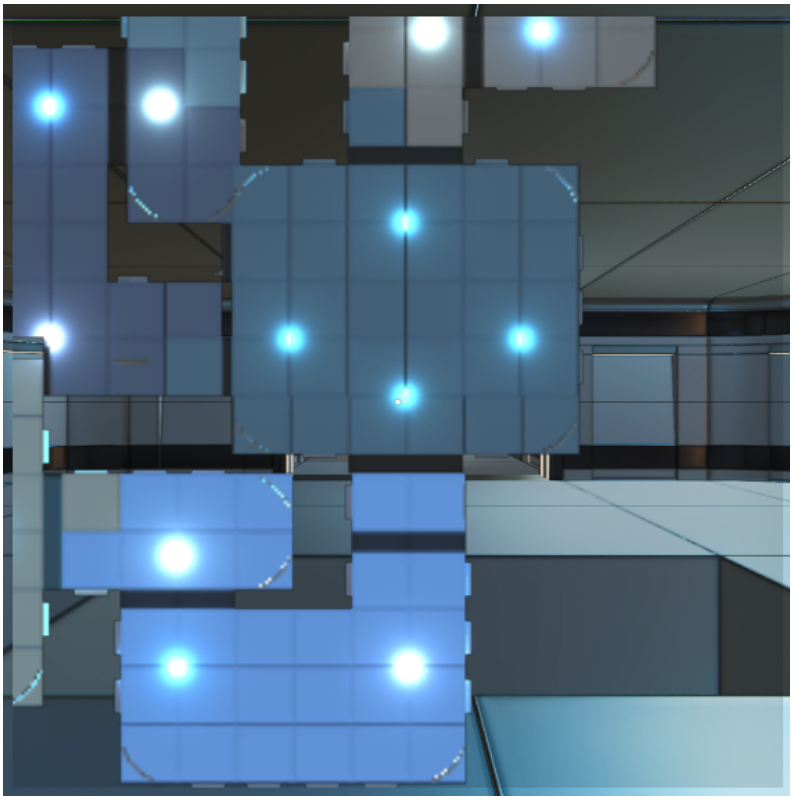


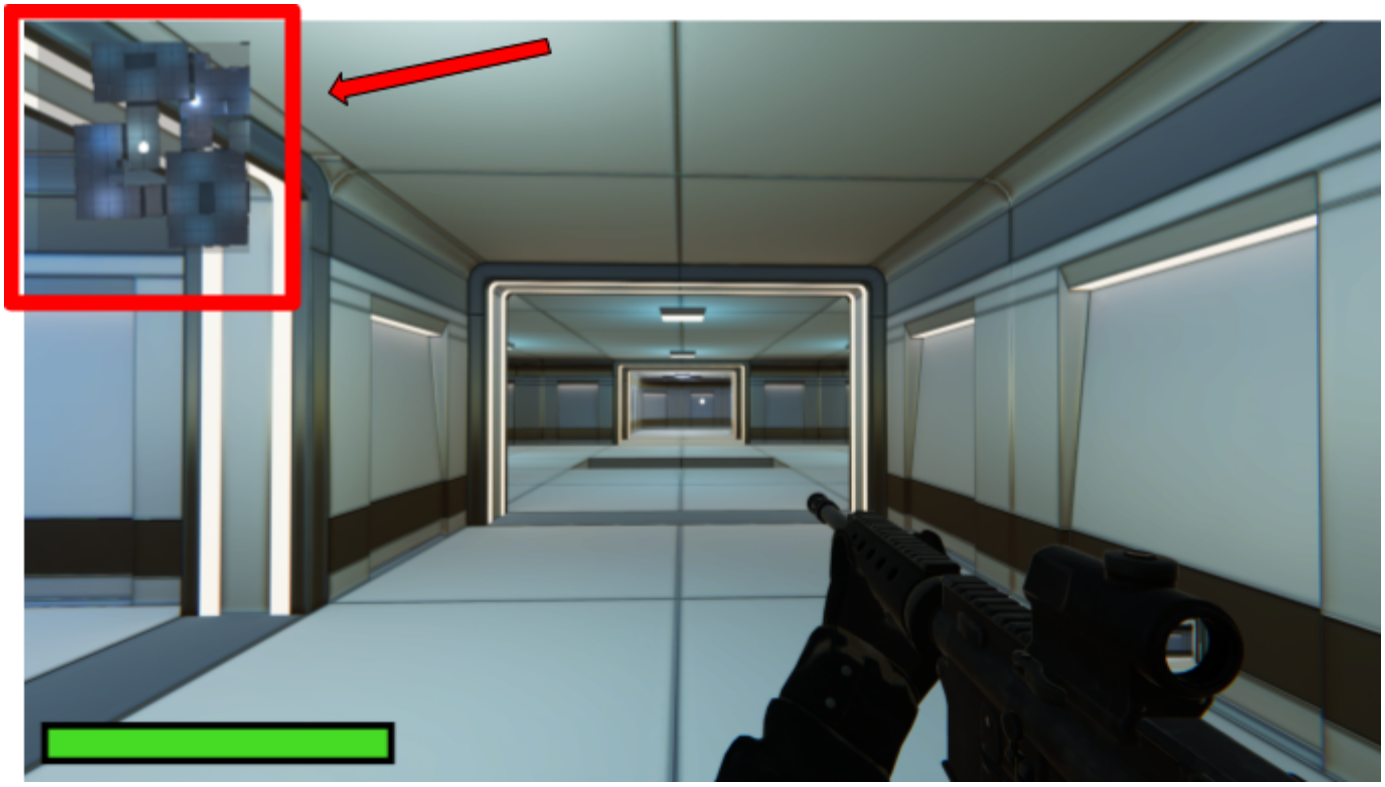
Map Layout:


y
x
Persp
Tools
No custom tools available



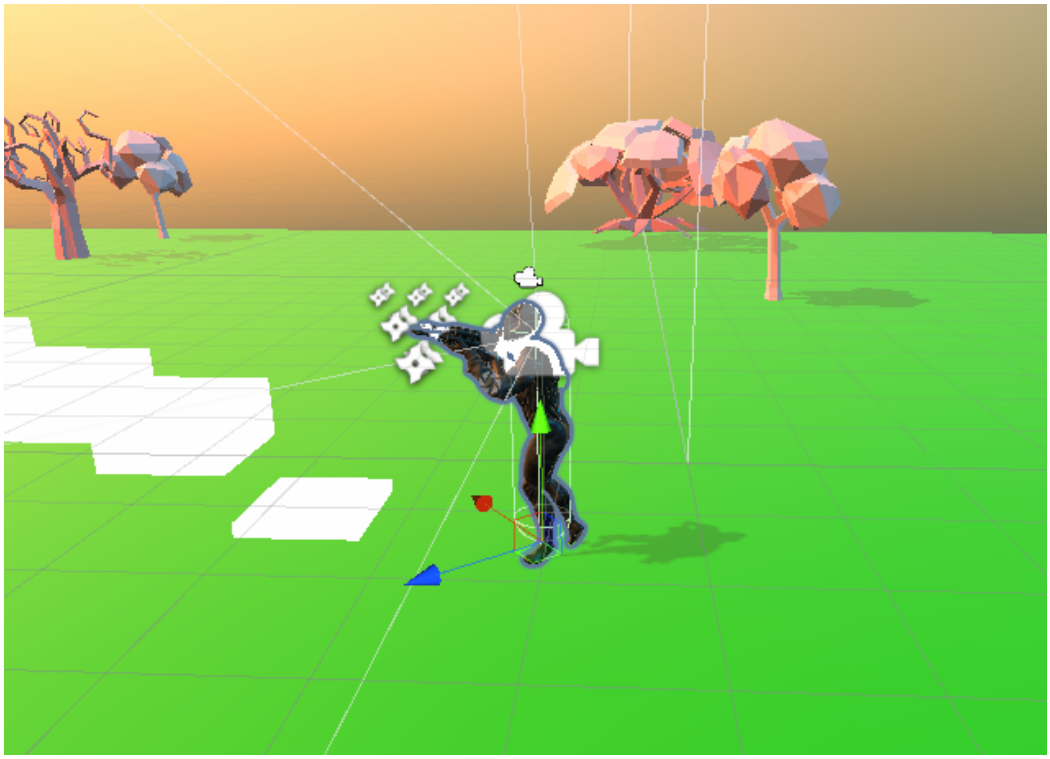
Implementation of Mini Map for the HUD:
The Minimap is implemented with the help of render texture.





The arms are rendered separately to achieve the first person POV:

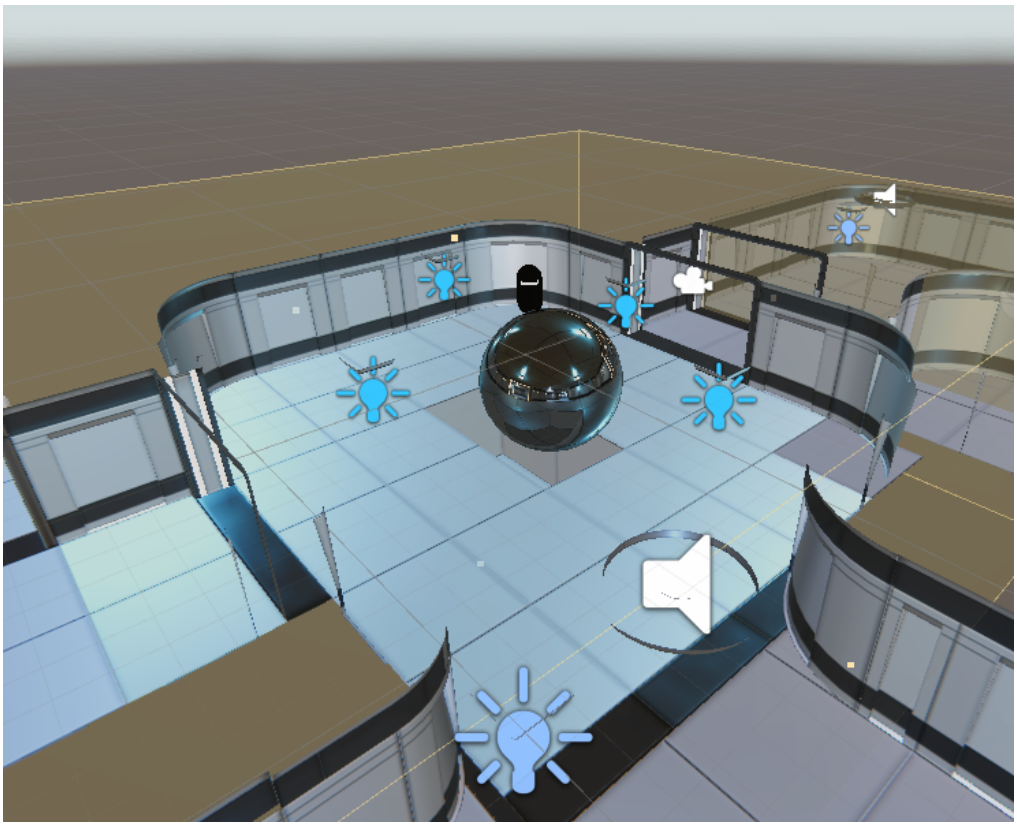
To achieve the first-person perspective the arm and the gun is supposed to move and rotate with the camera. This cannot be achieved if we just place the camera where the character's head lies. One way the first-person perspective can be achieved is to render the arm and the gun separate from the rest of the body. the player can only see his or her arms and the gun. only the opponent and see the whole character but not the sepreatly rendered arms.



The Scope is implemented by the use of render texture:

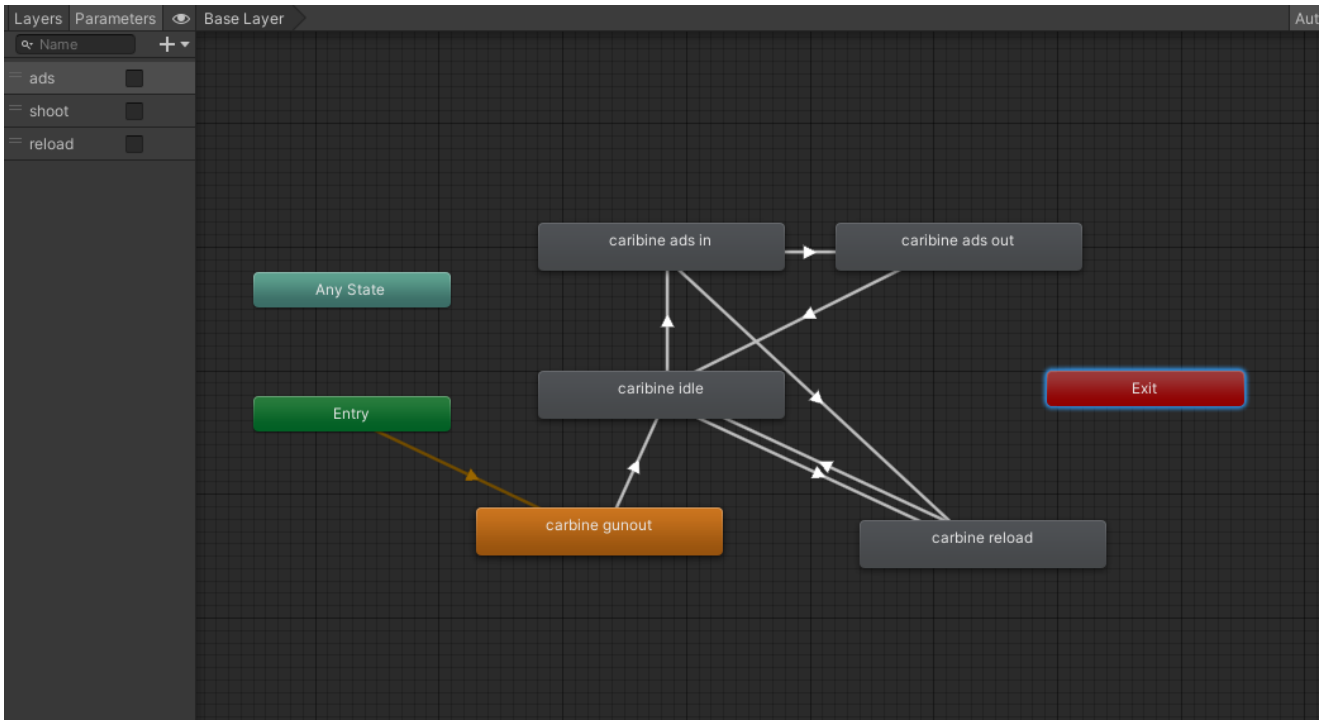
The output camera mounted in front of the gun is projected onto a render texture on the lens.

Use of reflection probes and backed lighting to achieve better graphics:



Each weapon is animated separately:

Each gun is separately animated with its one animation controller to achieve gun specific animations like reloading etc…



Photon Unity Networking:

Photon Unity Networking (PUN) is a Unity package for multiplayer games. Flexible matchmaking gets your players into rooms where objects can be synced over the network. RPCs, Custom Properties or "low level" Photon events are just some of the features. The fast and (optionally) reliable communication is done through dedicated Photo server(s), so clients don't need to connect one to one.

PUN's Structure:

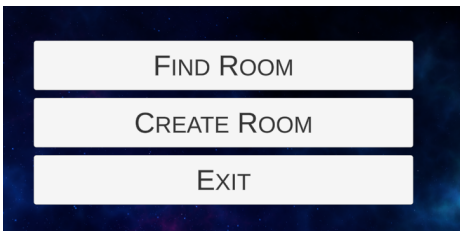
Usually, you don't have to mind the structure of the PUN package but just for the overview, here's how things stack up. The PUN package wraps up three layers of APIs:

- The highest level is the PUN code, which implements Unity-specific features like networked objects, RPCs and so on.
- The second level contains the logic to work with Photon servers, do matchmaking, callbacks and such. This is the Realtime API. This can be used on it's own already. You will notice a lot of overlap of topics between PUN and the Realtime API (a.k.a. LoadBalancing API) but that's fine.
- The lowest level is made up of DLL files, which contain the de/serialization, protocols and such.

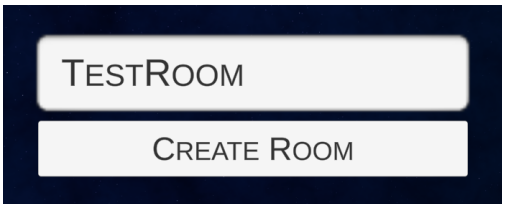
The Multiplayer aspect of this project is implemented with the help of PUN.

How to play the game?

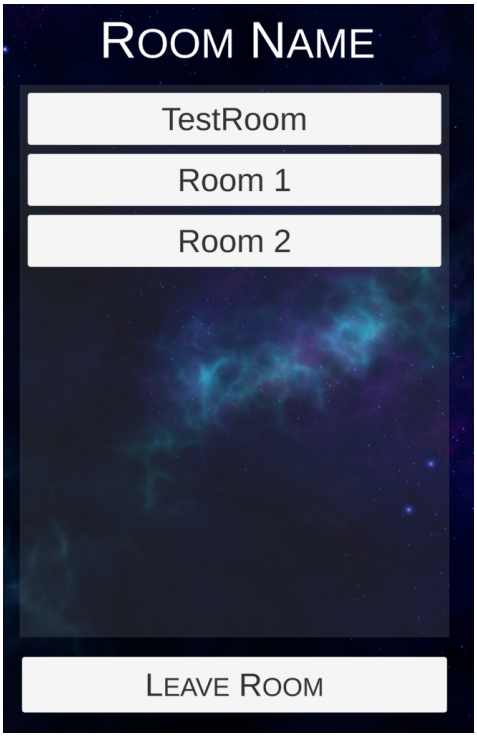
1. When you first launch the game you can create a room or join an existing room.



2. To create a room click on “Create Room” and enter a name for the room.



3. To join an already existing room click on “Find Room” and find the room you want to join.



4. Once you're in a room the host of the room will be able to start the game and everyone in the lobby will be loaded into the main level.
5. When the level is loaded the players compete to get the most number of kills. The player with the most number of kills wins.
6. Everyone in the room will randomly spawn in one of the reasons for positions scattered through the map.
7. Once a player dies he will respawn at a different random re-spawn point.

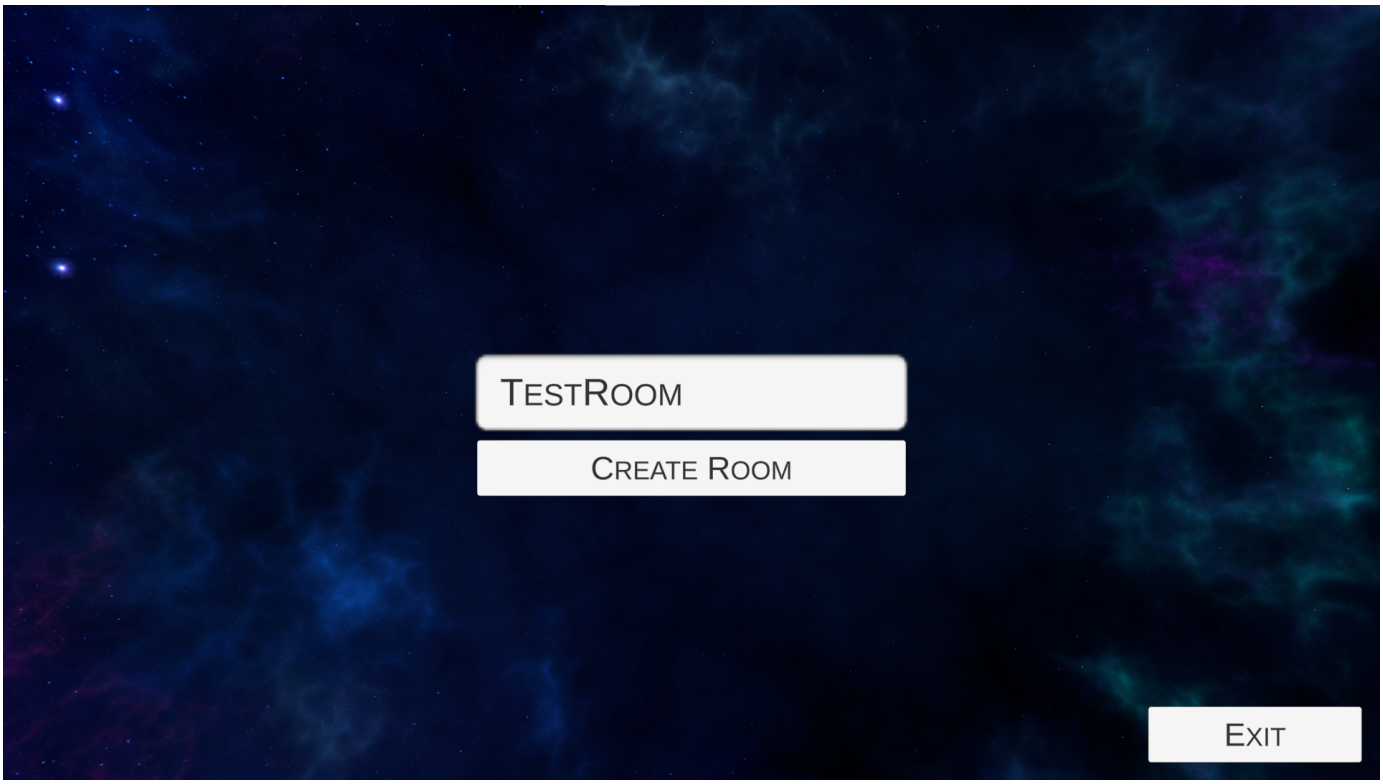
Key Binds:

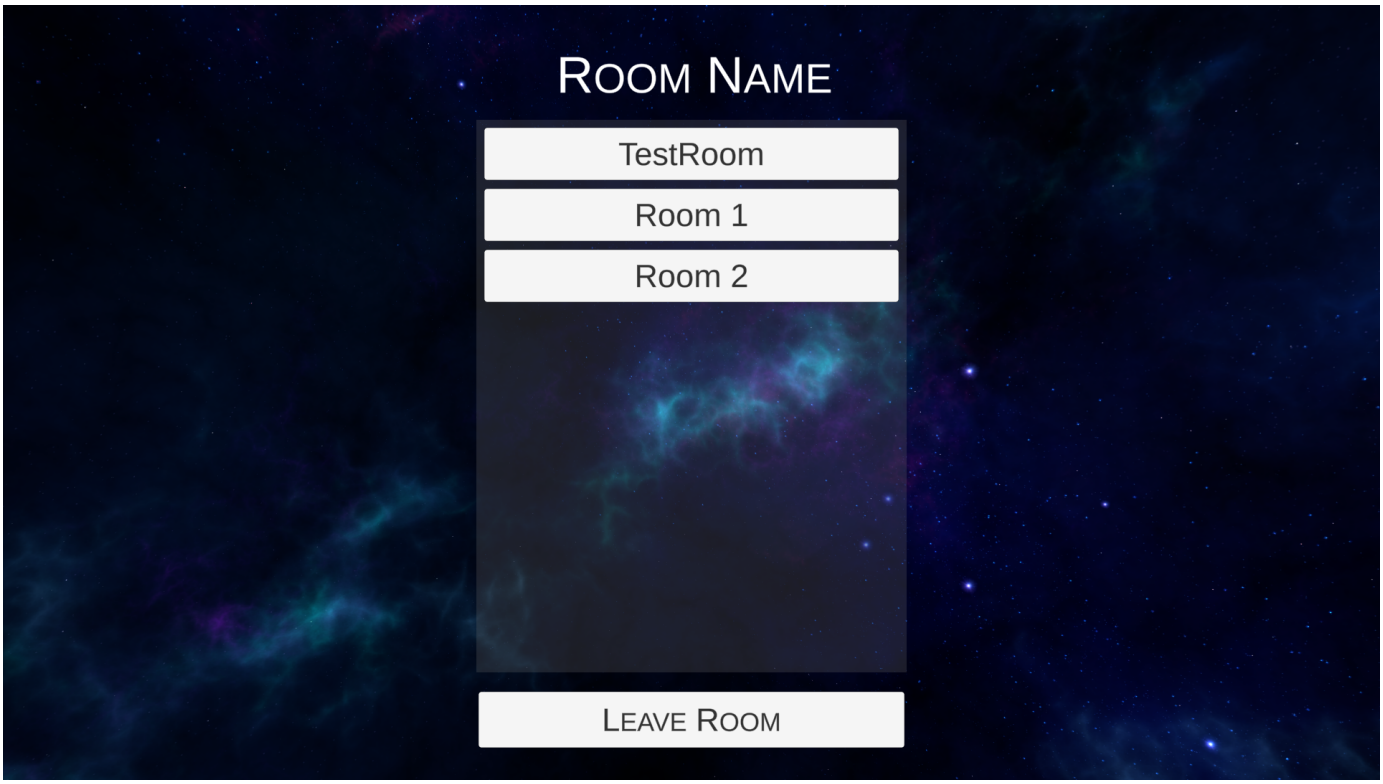
- W - Move forward
- S - Move backwards
- A - Move left
- D - Move right
- Space bar - Jump
- left Shift - Sprint
- Left Mouse button - Shoot
- Right Mouse button - Aim in (ADS)
- Number row - Switching Weapons.
- The mouse is used to looking around.

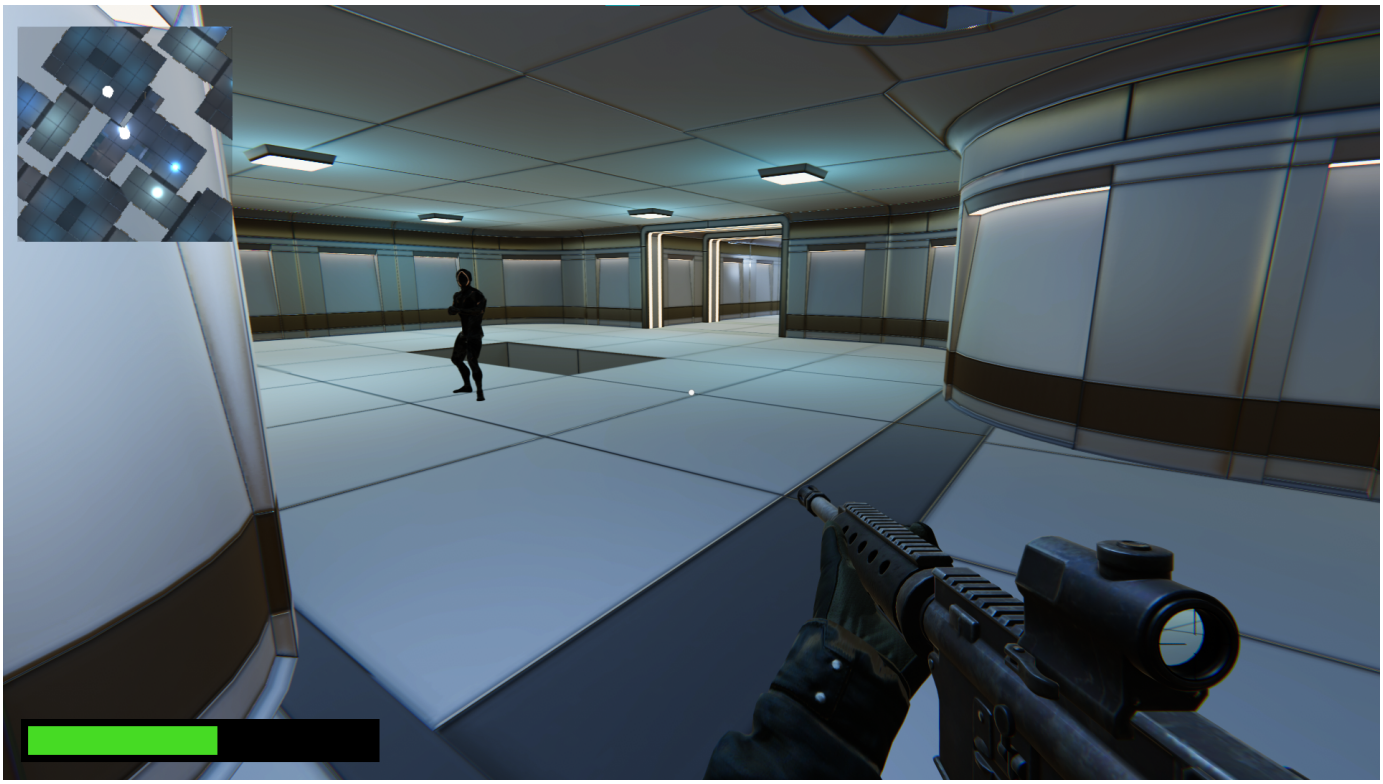
Starting position and Winning position of the game:

- Currently, the game has only one game mode which is **Deathmatch**.
- At the start of each game players in the same room are loaded into the map and spawn in one of the spawn points scattered through the map.
- The players are meant to deal damage to other players using their weapons.
- The performance of each player depends on their aim, knowledge of the map, precision etc...
- At the end of the round which usually lasts for around 10 to 20 minutes the player with the most kills wins.

SCREEN SNAPSHOTS




ROOM NAME
TestRoom
Room 1
Room 2
LEAVE ROOM



CONCLUSION

Like most shooter games, first-person shooters involve an avatar, one or more ranged weapons, and a varying number of enemies. Because they take place in a 3D environment, these games tend to be somewhat more realistic than 2D shooter games, and have more accurate representations of gravity, lighting, sound and collisions. First-person shooters played on personal computers are most often control led with a combination of a keyboard and mouse. This system has been claimed as superior to that found in console games, which frequently use two analog sticks: one used for running and sidestepping, the other for looking and aiming. It is common to display the character's hands and weaponry in the main view, with a heads-up display showing health, ammunition and location details. Often, it is possible to overlay a map of the surrounding area.

The process of making this game helped me to understand and learn various concepts and tools. This project made me familiar with unity and working with its assets, understanding networking and photon, learning about animating 3D characters and objects, working with Blender etc….

FUTURE WORK

Some improvements that can be made to improve the game are:-

- Adding more Weapons into the game.
- Adding game modes like Team Deathmatch, King of the Hill, Search and Destroy, Capture the Flag, Battle Royale, etc...
- Adding new maps which the player can choose.
- Making a networking solution from the ground up, helps us expand the survey capability of the game as we don't need to rely on photon networking.

REFERENCES

1. https://www.youtube.com/user/Brackeys
2. https://docs.unity3d.com/Manual/index.html
3. https://doc.photonengine.com/zh-cn/pun/v2/demos-and-tutorials/pun-basics-tutorial/intro
4. https://www.youtube.com/channel/UCtARM5yWhsJ-0MPKnQD-_zA
5. https://www.youtube.com/user/ImphenziaMusic