

**City, University of London**

**Computer Science with games technology**

**Final Year Project Report**

**Academic Year: 2020-21**

**3D FPS game with procedurally generated levels in Unity**

**by**

**Kevin La**

**Project supervisor: Mr. Eddie Edwards**

**January – May 2021**

## Table of Contents

Abstract .....	5
CHAPTER 1: introduction .....	5
1.1 Problem description .....	5
1.2 Beneficiaries .....	6
1.3 Assumptions and limitations .....	6
1.4 Project Objectives.....	7
1.5 Work performed.....	7
CHAPTER 2: Output Summary .....	7
2.1. Unity 2019.4.10f1 Editor Project .....	7
2.2. Project Windows Build .....	8
2.3 User and Installation Guide .....	9
CHAPTER 3: literature review .....	9
3.1 Introduction.....	9
3.2 Existing Games.....	10
3.3 Methods to achieve PCG .....	10
3.3.1 Dungeon Generation in my project.....	10
3.3.2 Cellular Automata.....	11
3.3.3 Binary Space partitioning .....	11
Chapter 4: Method .....	11
4.1 Analysis .....	11
4.1.1 Tool choices and reasoning .....	11
4.1.2 Unity version 2019.4.10f1 .....	12
4.1.3 Choice of algorithms.....	13
4.1.3.1 Binary Space Partitioning (BSP) .....	13
4.1.3.2 Cellular automata .....	13
4.1.3.3 Room based algorithm .....	14
4.2 Design .....	14
4.3 Implementation .....	14
4.3.1 Development methodology.....	14
4.3.2 Binary space partitioning.....	15
4.3.3 Predefined rooms algorithm. ....	16
4.3.3.1 Generating the levels.....	16

4.3.3.2 Object placement .....	16
4.3.1 The Player .....	16
4.3.1.1 Movement .....	16
4.3.1.2 Camera.....	17
4.3.2 Combat .....	17
4.3.3 NPC AI .....	17
4.3.3.1 Normal enemies .....	17
4.3.3.2 Mini bosses .....	18
4.3.3.3 Ally NPC .....	18
4.3.4 Loot.....	18
4.3.5 User interface .....	18
4.3.5.1 HUD.....	18
4.3.5.2 Menus .....	18
4.3.7 Save stats and Loading states.....	19
4.3.8 Leveling system.....	19
4.3.9 VR MODE .....	19
4.3.10 Evaluation .....	19
Chapter 5: Results.....	20
5.1 Analysis – Outputs .....	20
5.1.1 Tool choice.....	20
5.2 Implementation – Outputs .....	21
5.2.1 Binary Space partitioning .....	21
5.2.2 Predefined rooms algorithm .....	22
5.2.3 Player .....	25
5.2.3.1 Movement .....	25
5.2.3.2 Camera.....	26
5.2.4 Combat .....	27
5.2.5 NPC AI .....	30
5.2.5.1 Enemies .....	30
5.2.5.2 Mini bosses .....	31
5.2.6 Loot.....	32
5.2.7 UI.....	34
5.2.7.1 Heads up display (HUD) .....	34
5.2.7.2 Menus .....	35
5.2.8 VR MODE .....	35
5.2.9 Evaluation .....	37

Chapter 6: Conclusions and Discussion .....	38
6.1 Overview .....	38
6.2 Literacy Review – Relevance.....	39
APPENDICES.....	40
References .....	70
Unity Assets used.....	72

## ABSTRACT

The main objective of this project was to use Procedural Content Generation within my game demo to increase the replayability. I aimed to use the PCG algorithms in generating the levels so that each playthrough, the enemies, items and corridors will be different. I hope to have achieved a working prototype that allows players to play games repeatedly where every playthrough is different from the last.

The main problem researched was whether an open world game with many narrative pathways, side quests and a huge amount of freedom leads to a game with a lot of replayability. I wanted to see whether there was an alternative way to add replayability while keeping the production cost and time low by using PCG algorithms. The research aim was to investigate whether PCG algorithms would add value to the game by theoretically adding an infinite number of new levels and making things fresh.

## CHAPTER 1: INTRODUCTION

---

### 1.1 PROBLEM DESCRIPTION

In recent decades, video games have seen a diverse global audience and has had an impact on multimedia culture as a whole. They 1<sup>st</sup> started as text-based games and slowly evolved into 2D pixel games, to 3D games thanks to the help of new technology that allows game developers to create such astonishing games and allows companies to produce next generation hardware to allow for such photo-realistic games to be displayed onto a screen. Video games is an effective way to dive into a virtual world, being fully immersed into the game in many ways. Having a 1<sup>st</sup> person camera perspective is 1 of the main methods in immersing the player into the game.

The problem with FPS games now is that despite them offering a fully immersive experience, many of them are story driven and follow a linear structure which in turn leads to very limited freedom and choices. Having a linear game means that the player must follow what the story has in store for the player which results in subsequent playthroughs to be tedious as the player will already know what is going to happen, limiting the replayability value of the game.

There are many examples of FPS games which gives the player a lot of freedom as well as a story based on narrative choices. Games like Fallout series is a great example that allows the player to explore whatever they desire and progress at their own pace while affecting the story outcome based on their decision. This is a great way to increase replayability of a game as each playthrough the player may choose different choices throughout the game resulting in a different ending however this requires high production value.

The project objective is to give an alternative solution to the level design for FPS games. This can be used to improve the replayability of games as well as reducing the need for level designs, especially for a smaller team.

The project will explore different methods of generating levels algorithmically. The project will then evaluate which method of PCG best suits a rogue-like fast paced action game (similar to Doom and Strafe).

The project attempts to see whether the inclusion of Virtual reality further enhances the user's experience and adds even more replayability to the game.

---

## 1.2 BENEFICIARIES

Project Beneficiaries may include the following:

- Me, as the sole developer of this project, due to having learned a lot of new functions in Unity/C# and also learned a lot about PCG techniques.
- Indie developer/individual developers may benefit from this as they can build upon this PCG algorithm to create a game that seeks PCG algorithms to generate levels for them.
- Research students who want to dive deeper into the uses of PCG where this project can serve as a starting point and develop the algorithm even further. (Project uploaded on GitHub)

---

## 1.3 ASSUMPTIONS AND LIMITATIONS

Assumptions that were made:

- My level with Unity and VR development in Unity will be enough to successfully achieve my objective while overcoming any potential engine-level difficulties faced during the development of the project.
- Skills such as 3D modelling is not needed as they carry no significance and may impact the overall result of the project due to being too time consuming.
- I could reuse free pre-made rooms from the Unity asset store for the level generation in order to save time to focus on other parts of the game

Because of the type of game I wanted to make, I have imposed limitations for what the final project should achieve.

- Optimization of the game has not been a huge focus in the project. Where optimization implementation is easy and quick is where it has been employed.
- Areas such as 3D assets, UI and sound effects have been set as low priority due to those only enhancing the game allowing more time to be focused on PCG, Gameplay and VR mode.

- The final project is not fully developed game but rather a prototype to demonstrate PCG techniques and my skills in Unity.

---

#### 1.4 PROJECT OBJECTIVES

The main goal of this project was to produce at least 1 working procedural generation method in an FPS Game with additional goals being to add VR support for the game.

The project created was split into 3 iterations with the 1<sup>st</sup> iteration aiming to deliver the basic functionality of an FPS game and Level generation; the 2<sup>nd</sup> iteration focusing on improving content generation and adding other features seen in most if not all FPS games; the 3<sup>rd</sup> iteration focusing on adding more gameplay features to add variety such as difficulty selection, polishing the game, and implementing the Virtual Reality mode.

Throughout the project, majority of the assets were either created in blender or imported from the Unity Asset Store (Either Free or Paid) and the link has been provided in the reference section. As for sound and music they were obtained from Freesound.org and Bensound.com, both provide royalty free soundtracks.

---

#### 1.5 WORK PERFORMED

Throughout the project, the main objective has stayed consistent however in terms of features I initially planned to implement, these have changed over time. Some features (the Save/Load System , Ally NPC, and Levelling system) stated in the build requirements in the 2<sup>nd</sup> build iteration and from 3<sup>rd</sup> build iteration have been fully removed as they were insignificant and did not affect the project fulfilling its primary goal as well as the extra feature - Virtual Reality mode. More detail on reasoning in the methods section.

Despite all these removed features, the project still managed to deliver on its main goal which is to provide an alternative method of content creation to add replayability that requires no high production value while offering players an additional way to experience the game – Virtual Reality.

### CHAPTER 2: OUTPUT SUMMARY

---

#### 2.1. UNITY 2019.4.10F1 EDITOR PROJECT

Unity 2019.4.10f1 Project	File type: .zip                      Total size: 2.68GB
Description	<p>This zip file contains the project that was built in Unity including all source codes, assets and engine related configuration files and packages.</p> <p>The project folder consists of:</p> <ul style="list-style-type: none"> <li>• Source code with roughly 9221 words (3070 lines) of code written by me</li> <li>• Assets used were taken from Free asset store or from elsewhere that has provided them for free (Link in reference section)</li> <li>• Packages and dependencies such as the OpenXR plug-in</li> </ul>
Intended recipients	Project consultant, markers, other academics and developers.
Benefit	Users can see the work done through the Unity project. Developers can also modify the PCG algorithm or add their own spin of the already product algorithm.
Linked Appendices	Appendix H

---

## 2.2. PROJECT WINDOWS BUILD

Windows application	File type: .zip,.exe                      Total size: 139mb
Description	<p>This file is what the intended recipients will open to play the game. This is the latest build of the Unity project and has been compiled for Windows 32bit and 64bit machines only.</p> <p>Support for other OS has not been tested.</p>
Intended recipients	Project consultant, markers, testers, other academics and developers.
Benefit	Allows users to test play the game and have a feel of the VR mode as well as the PCG algorithm implemented.
Linked Appendices	Appendix H



---

## 2.3 USER AND INSTALLATION GUIDE

<b>User and Installation Guide</b>	File type: .pdf	Total size:160kb
<b>Description</b>	To guide users on how to run the game and inform them of the controls and objectives of the game.	
<b>Intended recipients</b>	Project consultant, markers, testers, other academics and developers.	
<b>Benefit</b>	The guide informs the user of important aspects of the game required to play without needing to figure them out themselves.	
<b>Linked Appendices</b>	Appendix G	

## CHAPTER 3: LITERATURE REVIEW

---

### 3.1 INTRODUCTION

The use of procedural content generation in video games are becoming more and more popular and it refers to content being created using algorithms. Many modern games implement this for the same reasons – Replayability and reducing cost production. This technique is used on game content such as levels/stage, quests, characters and many more.

PCG is not only present in games, there are many other applications for this technique, and this can be seen in software's or in film. An example of PCG being used in film is the popular TV show "The Mandalorian". *"The landscapes shown in the show have been largely generated by a computer that is photo-realistic and is wrapped around the physical set and real actors to create a seamless effect"*[1].

Despite it increasing in popularity, it is not easy to implement or effective. 1 of the main disadvantages of using PCG in games is that the content generated may not be visually unique compared to content that is hand crafted. There are ways to mitigate this by making the procedural content generator configurable where you can change the size of a level or state how many objects can spawn in that level.

Another disadvantage is that it will be hard to code game events. *"If the entire world around you are generated exclusively procedurally and randomly, then it makes it almost impossible to script fixed game events"*[2]. So to tackle this we could use both Procedural content generation and premade content which will allow players to explore and interact at their own leisure while being able to experience game events when progressing.

---

## 3.2 EXISTING GAMES

Many games used PCG in different ways however the most prominent use of PCG is map or level generation. Many of the games that use it for map generation are usually open world.

Minecraft has been 1 of the most popular games out now and is still booming with how much exploration and new discoveries there is in the game. An article states that Minecraft achieves these biomes that are generated and seamlessly connected by “using Perlin noise calculations, like the kind you’d use to create a rough-looking texture for a 3D model.”[3] With this method, Minecraft can create randomly generated mountains that are a delight to explore.



*An image of a mountain in “Minecraft” that is generated using the Perlin noise method*

Another game that uses PCG but more for its weapons system is “Borderlands”. There is no direct information from the developers themselves on how this was done however, other game developers have created a weapon system that is heavily influence by Borderland’s weapon system and has shared how it is achieved *“when the weapon is being generated it spends the points on the modifiers, that affect the stats. And the higher the rarity, then the more powerful, or more stats it can have”*[4].

---

## 3.3 METHODS TO ACHIEVE PCG

---

### 3.3.1 DUNGEON GENERATION IN MY PROJECT

There are 2 ways to achieve generated dungeons. We can create rooms that have random objects inside and connect these rooms randomly with a corridor using various procedural techniques and building our own algorithm as seen in the FPS Game called Strafe. Another approach is to place corridors using a maze generation algorithm which adds slightly more randomness and many pathways which may lead you to the goal or not.

The first method may be the easiest and least time consuming to implement which is what I plan to do in the project. The only problem with the 1<sup>st</sup> approach is that *“it does not guarantee connectivity between all parts of the dungeon for every dungeon generated, so may require an existing dungeon*

*layout be discarded and started again.”[5].* The problem with the 2<sup>nd</sup> approach is that I have no flexibility over what is inside the rooms and the game events that may be implemented and that a complete maze level is not what I hope to have in my game. I envision my game to have levels with rooms that connect to each other with each room being different.

---

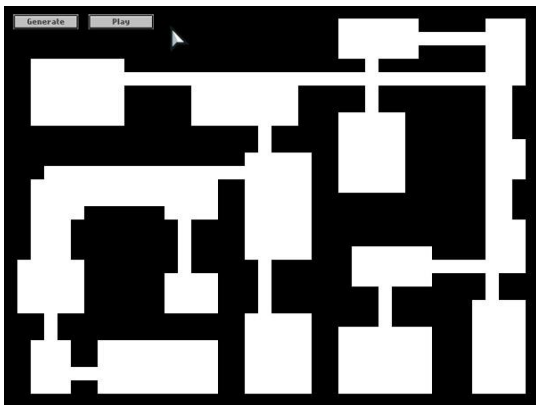
### 3.3.2 CELLULAR AUTOMATA

Cellular automata aim to generate cave-like structures. This may be better to create a more natural and organic level. The advantage of using cellular automata is that the process time of this algorithm is very low. Since the computational cost of this algorithm is very low, the game can then allow *“Realtime content generation, and the proposed map representation provides sufficient flexibility with respect to level design.”[6].*

---

### 3.3.3 BINARY SPACE PARTITIONING

This is another method that is that is used for content generation and is 1 of the top choices for me to use when creating this project due to how the result of the algorithm is very close to what I envision the game’s level to look like.



[7]An image of the resulting algorithm

How this algorithm works is that it takes an area and splits it up into smaller areas. These areas are split either vertically or horizontally. This is then repeated until their number of maximum splits is reached. 1 of the main advantages of BSP is that it handles collision detection well as only certain parts of an area are checked for collisions.

## CHAPTER 4: METHOD

---

### 4.1 ANALYSIS

---

#### 4.1.1 TOOL CHOICES AND REASONING

Upon starting this project, there were many different engines I had in mind that the project could have been developed in for example, Unreal engine 4, Unity, CryEngine. Of all the different Game engines,

In order to meet the objectives support for VR development was required. Unity and Unreal Engine both support VR development and can output extensive VR games. They are similar in terms of what can be developed in VR. The difference comes with how strong the engine is and the graphical limitations which Unreal Engine wins. The reason for this may be that Unreal coming from a Triple A game background meaning that its focus is on graphics therefore, Unreal offers better shaders and Visual effect out of the box however gameplay and content was my main focus, so I did not require such graphical strength.

[illegible]

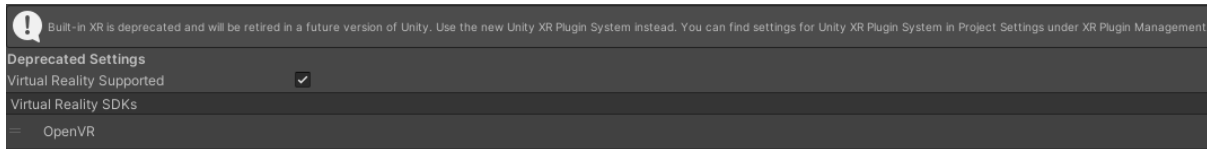
There are other options like creating my own game engine with PCG however with the time available this was out of the question as most of the time would be developing the game engine leaving little to no time to actually developing the game.

#### 4.1.2 UNITY VERSION 2019.4.10F1

12 / 73

it a separate plugin called OpenXR. However, there is a work around for 2019.3 and 2019.4 without the need to download a separately new plugin.

Despite Unity not supporting OpenVR in the previously mentioned Unity versions anymore, they still allow you to install it as seen in the below picture.



In future versions this work around will not be present and will require the developer to install the new OpenXR plugin separately until they have added the OpenXR plugin built into future Unity versions.

Overall, due to practicing VR development on Unity version 2019.4.10f1 previously, I stuck with this version for this project since I know the OpenVR plugin works in this version of unity and my knowledge of VR development stems from this version of Unity and OpenVR Plugin.

---

#### 4.1.3 CHOICE OF ALGORITHMS

##### 4.1.3.1 BINARY SPACE PARTITIONING (BSP)

The 1<sup>st</sup> algorithm BSP, was implemented following a tutorial provided by Sunny Valley Studio [9]. This method of PCG was a starting point for me as with no experience in PCG I required guidance. Additionally, this way of generating content is mainly used for content generation in 2d dungeon crawler games so implementing it in a 3D FPS game was an interesting area to explore. This algorithm was also the closest to producing a room-based structure that I had wanted.

##### 4.1.3.2 CELLULAR AUTOMATA

The 2<sup>nd</sup> algorithm researched was Cellular Automata. This was not implemented at any stage of the project due to the end results of the tutorial watched, not fitting the initial game design which is room-based and was more suitable for open world games or dungeon crawlers. The tutorial in question was provided by Sebastian Lague[10] where he goes through step by step on how to produce a cave-like system using the algorithm and gave me an insight of how it works and what the end results look like.



*A top-down view of the generated*

*map created by Sebastian Lague.*

---

#### 4.1.3.3 ROOM BASED ALGORITHM

The 3<sup>rd</sup> algorithm implemented being room-based algorithm. I wanted flexibility with my level and something similar to the game Strafe where they work with predefined rooms and connect them together using rooms that imitate corridors. The flexibility comes from creating the rooms themselves instead of setting parameters for the room to be generated. I can structure the rooms how I like and set where things spawn giving the sense of a hand-crafted natural room instead of an artificially generated room.

The only possible drawback of this method is that it may not give the natural feel like Cellular Automata however with the game I am making (Rogue like fast paced action), this may not be necessary due to the player wanting to progress through the level instead of exploring a cave like map.

---

### 4.2 DESIGN

Before implementing the game, I have created UML class diagram for the core gameplay. The diagram does include the main scripts due to having a good idea of how these will be implemented. Classes for the level generation as well as VR mode have not been represented in the class diagram due to these being unknown from the start and they have highest chance to change throughout the project. The ending results may differ to what the diagram depicts as UML Diagram goes against the chosen methodology – Agile which revolves around flexibility and producing results quickly thus the diagram was not used throughout the whole project but more of a basic guide to follow to visualise how things will operate and with each other. The diagram (Appendix F) has been created using Visual Paradigm Online.

---

### 4.3 IMPLEMENTATION

---

#### 4.3.1 DEVELOPMENT METHODOLOGY

The development methodology used in this project was Agile due to its flexibility and ability to adapt to change which is frequent in the project this may go against the use of UML diagrams however the UML diagram was only made to give a rough idea and be used as a guide. For game development, I feel that UML diagrams are restrictive in a way so it should not be followed throughout the whole project.

The main reason of choosing agile is because of how game development usually is. As you develop a game and more features get implemented, features will need to adapt to new features being implemented so that they work together so being able to change and improve easily is much needed. I feel like agile allows for a more improvisational approach to the project which suits my workflow. Agile also allows for functionality to be developed quickly in order to be demonstrated to get self-evaluated or by others.

---

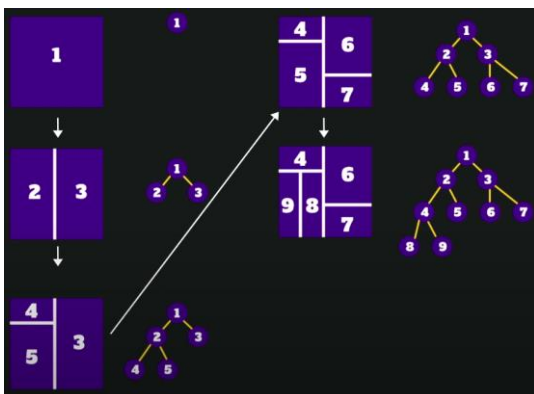
#### 4.3.2 BINARY SPACE PARTITIONING

The 1<sup>st</sup> algorithm tested was Binary Space Partitioning which was produced following the tutorials by Sunny Valley Studio mentioned previously. This method uses BSP to divide a space into 2 and is done recursively until the algorithm cannot divide the spaces anymore. It divides a space using a line vertically or horizontally with cross section of the line being randomized. After space has been divided by line, it checks if the space can be divided any further (While adhering to the minimum room size).

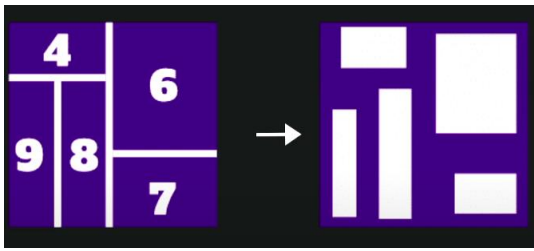
Parameters are set to configure how the algorithm divides rooms such as minimum room size, maximum room size, how many times to divide the space. This can of course be modified as I see fit.

For every divided space, create room by choosing the corners of the divided space boundaries. Once that has processed, the algorithm proceeds to draw corridors using nodes from the youngest tree branch and traverse the layers while drawing corridors for the newly selected branches.

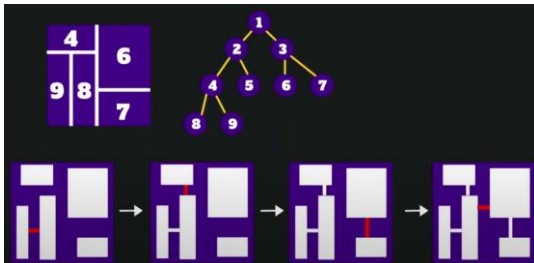
Here is a visual representation of the BSP concept taken from the aforementioned tutorial:



Dividing a space



Generate rooms within the spaces



Create corridors connecting the rooms.

---

### 4.3.3 PREDEFINED ROOMS ALGORITHM.

#### 4.3.3.1 GENERATING THE LEVELS

The final algorithm used in the final build is very much inspired by the game *Strafe* as discussed in the research section. The difference being that the algorithm used in *Strafe* connects rooms not only using door nodes but also using a “tile plug” which “connect two spaces and the purpose of this connection is to help change the feel of rooms and not just connect them at the doors.” [10].

I have implemented the PCG algorithm using predefined rooms created and imported from blender. Almost all of the code and logic used for this algorithm was from the following tutorial from a channel on YouTube called ProjectShasta which goes through the steps to achieve a procedural level generation using predefined rooms [12]

After completing the tutorial using my own imported blender rooms plus rooms provided by the tutorial to start me off, I have gained a lot more knowledge on PCG and Unity itself – this may be due to the tutorial explaining very well how the algorithm works.

#### 4.3.3.2 OBJECT PLACEMENT

I added more to the algorithm by implementing my own object placement script. The method I went with was to add spawn points to each room. Each spawn point has a specified X,Y,Z position. These spawn points are then chosen in the script and a random game object is instantiated at the position of the selected spawn point. There are multiple “Spawner” scripts to handle spawning different types of game objects like enemies, power ups or bosses.

---

### 4.3.1 THE PLAYER

#### 4.3.1.1 MOVEMENT



The player that the user controls is made up of several components and game objects operating concurrently. The player prefab has a Character controller which handles moving the player according to the environment colliders, a custom [Player](#) script that handles button inputs to move the player, the properties for movement values, player collision and player stats like Health and speed. The movement functionality (Walk() and Jump() function) was produced following a tutorial by Brackeys [\[13\]](#). The rest of the code was produced by me.

The [Player](#) script attached to the main game object is what translates the user's key presses into values by getting the axis which will be used to move the player in a direction. The player can move with the usual W,A,S,D key and jump using the space bar.

---

#### 4.3.1.2 CAMERA

The player will be able to look around in a 1<sup>st</sup> person perspective by moving the mouse. As it was my 1<sup>st</sup> time working on Unity 3D, Majority of the code has been produced following the same tutorial provided by Brackeys for the movement. My only addition to the script was to hide and unhide the cursor depending on if the game has been paused.

---

#### 4.3.2 COMBAT

Combat is simple but a good foundation to work upon in the future. The player can equip weapons (When close enough) to a weapon slot (Player has access to 2 gun-slots and 1 melee weapon slot) using the dedicated equip key and shoot/attack with left mouse click. The pickup system was produced following a tutorial by Dave/GameDevelopment [\[14\]](#) and has been adapted so that it works with the [SwitchWeapon](#) script.

In FPS games, players are usually given the option to hold onto multiple weapons that they find and can easily swap between them at any time. Due to my inexperience in FPS games, to achieve weapon switching I used a tutorial from the channel Brackeys [\[15\]](#)

As it was my 1<sup>st</sup> time dealing with shooting in Unity, I used a tutorial provided by Dave/Gamedevelopment [\[16\]](#) to implement the Shoot and Reload/ReloadFinished function. There is also a simple sword combat to deal damage in case of no guns or ammo.

The logic to reload has been adapted by myself and added on top of the Reload function produced by the tutorial from Dave/GameDevelopment to produce a more realistic reload.

---

#### 4.3.3 NPC AI

---

##### 4.3.3.1 NORMAL ENEMIES

With the enemies, they will be spawned at points within the room. Enemies have a simplistic AI using the NavMesh component and Finite State Machine produced by a tutorial from

Dave/GameDevelopment [17] The FSM contains the basic states such as patrol, chase, and attack plus an additional shoot state. Each state is activated depending on how close the player is to the enemy.

---

#### 4.3.3.2 MINI BOSSES

For the mini bosses, they have the same AI as normal enemies (Spawning of the boss as well as and FSM). 1 of the differences the boss has compared to the regular enemies is the boss properties are higher like health, attack times and so on. The boss has both close attack and shoot state much like the normal enemies. In order to make the mini boss have its own identity I gave them a new state which is called “Enraged” which enhances the boss’s stats when health is low.

---

#### 4.3.3.3 ALLY NPC

Due to time constraints I decided to not implement this feature. This feature was selected to be left out as this has no significance impact to the core gameplay and implementing this would be very similar to implementing enemies. The only difference being the target its shoots at which would be the enemies and setting the player’s location as the NPC’s destination so that the ally follows the player.

---

#### 4.3.4 LOOT

Throughout the game, the player will be able to obtain power ups and ammo simply by touching them and depending on the power up, different changes will occur to the player.

These items will be dropped upon killing enemies or spawned inside a room randomly. The health orbs have been given a magnetic force so that it gravitates towards a player and spawned loot from defeating an enemy have a stylized effect. The magnetic effect was produced using tutorial by MakingMagic [18]. The stylized drop effect was coded with the help of a tutorial from a YouTube channel by One Wheel Studio [19].

---

#### 4.3.5 USER INTERFACE

---

##### 4.3.5.1 HUD

The HUD consists of important information such as health, ammo count and a crosshair on the screen with the health and ammo count having their own script. The information displayed gets updated as the game is played. The [HealthBar](#) script was produced following tutorial by Wayra Codes [20] and has been tweaked to optimize the script. The [AmmoCount](#) script however was produced by me.

---

##### 4.3.5.2 MENUS

The main menu is where the player will be able to select the different modes they would like to play as well as changing the setting of the game or quit the game. Due to time constraints and graphics not being the main focus, I imported Unity’s sample menu from the asset store and adapted the logic of the menu to suit the project.

As with most games, the player can pause the game pressing Escape key which brings up the pause menu. The pause menu allows players to stop the game in its current state while having the option to select the different functionality such as resume game, look at controls and instructions or quit to main menu. The pause menu was created without any tutorials and the shapes used for the buttons and the texts were assets provided by Unity itself.

---

#### 4.3.7 SAVE STATS AND LOADING STATES

Saving and Loading states was left out due to it being too long to learn as this was something new to me. Not only that but I felt like Saving and Loading states is not required for the prototype of this game as it does not in no way help justify my points stated in the PDD. This kind of feature is in my opinion a quality-of-life feature which saves time or allows players to take a break without having their hard work gone to waste.

---

#### 4.3.8 LEVELING SYSTEM

Levelling system was also another feature that was left out in the final build of the project. This feature was planned to give players an incentive to defeat enemies and gain new abilities upon levelling up. Main reason why it was left out is due to it not being relevant to the main goal and due to the amount of content currently available in the demo, the levelling system would not impact the gameplay as much as I would have liked.

---

#### 4.3.9 VR MODE

With the VR mode, the player will be able to play the game but with full VR controls meaning how the player plays will be adapted for VR. The aforementioned keys mapped to different actions like Shoot, Reload, Move, Jump have all be configured for VR controls. In order to look around, the camera has been mapped to the headset being tracked so the rotation and orientation of the headset is reflected on the camera.

Not only was inputs and actions adapted for VR but also weapons. Weapons will be grabbable using the grip button on the controllers. The shoot function was mapped to the triggers with the reload function being completely changed and adapted to suit VR. Most of the VR implementation was produced following a playlist from Valem [21].

---

#### 4.3.10 EVALUATION

In order to evaluate each feature I implemented and specially to see if the PCG algorithm implemented worked, I decided to get other people's opinion on the game. This was done by distributing online survey from Qualtrics to the hand selected people. After researching, I decided to set the online survey as Anonymous mode meaning identifiable data of the respondent is not collected. This decision was made due to research saying that anonymous surveys produce more accurate results as knowing

that their identity is visible may affect what answer they pick. Having it anonymous will encourage respondents to take the survey as honest as possible resulting in more reliable and accurate data.

The surveys would be created based on each build iteration of the project. The player will receive a build of the project based on the which build iteration has been completed. This would allow myself to see which feature worked and to quickly go back and fix the bugs before moving onto the next feature or iteration. After the 3<sup>rd</sup> Iteration has been completely developed, 2 new survey will be distributed. A survey for VR users and a survey for the final build.

The VR survey would consist of questions relating to the controls for VR and to see if they work with everything else. The final build survey would consist of questions from Iteration 1 survey up to iteration 3. This is because developing games, some features may interfere with each other so to best evaluate the prototype, I included questions based on every feature like:

- Can you move the player around with WASD keys? Yes or No?
- Can you look around with your mouse? Yes or No?
- Does the game generate a different level structure every playthrough? Yes or No?
- Any other feedback? (Text entry)

The answers from the survey would inform me whether certain features worked and to see if the generator does indeed randomize the level structure each playthrough.

## CHAPTER 5: RESULTS

### 5.1 ANALYSIS – OUTPUTS

#### 5.1.1 TOOL CHOICE

In the final project, the tool choice used throughout was indeed Unity 2019.4.10f1 which has followed my initial plan and has delivered results as expected. This unity version is no longer being updated as Unity has moved onto 2020 version which can affect the future development as recent versions of the engine may introduce new features and bug fixes to ensure engine stability. Due to these reasons I attempted to import the project into the 2020 version of Unity however there was a major problem as the way the 2020 version of the engine handles VR development is much different to the 2019.4.x version. With 2019.4.x they have OpenXR plug-in built into the engine which can be toggled on or off but with 2020 edition, they do not support the plug-in and require the developer to install it separately which comes with its own complications. This led to ultimately sticking with my initial choice throughout as I knew everything works.

Before I get into the implementation results of this report, some key concepts of Unity should be detailed first to give a better understanding of how the engine works.

Unity revolves around components and objects. Each entity in the game is a game object and every game object can have multiple components attached to them. The components determine the functionality of the game object it is attached to. Every game object will always include a transform component which specifies the position, rotation, and scale in the game.

Another important concept Unity has which helped speed up the development of this project is Prefabs which is used often in this report. Prefabs can be seen as a template which can easily be added into the game at runtime or in the editor. When a prefab is made, any changes to that original prefab is reflected in all the instances of that prefab within the game. However if you edit an instance of a prefab in the game (Not the original prefab), It will not update the other prefabs, but you do have the option to have this change reflect on every other instance of that prefab.

Scripts is almost always used for each game object within my project, and they handle the logic of what a game object does. They also store variables and provide variables for other scripts to access globally. A script allows the ability to reference other data from another script that may be attached to another game object within the scene.

---

## 5.2 IMPLEMENTATION – OUTPUTS

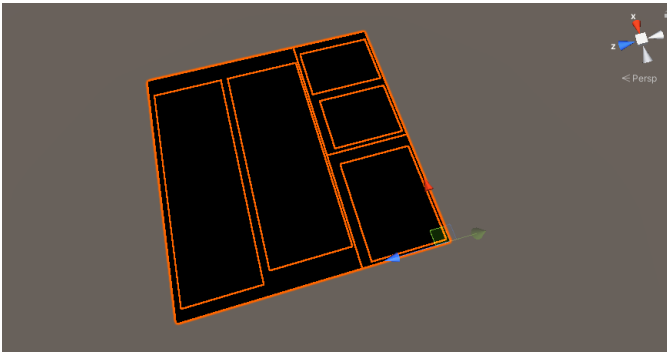
---

### 5.2.1 BINARY SPACE PARTITIONING

The BSP algorithm has been partially implemented and replaced completely in the final build of the project however there were some early results outputted from using this algorithm. Even though the algorithm is not complete, I kept the source code and scene in the project for further testing in the future and if needed, I can always refer back to Sunny Valley Studio's tutorial to complete the tutorial and experiment.

Despite the algorithm being completely based on a tutorial by Sunny Valley Studio, it had little to no value to the project because it was fully produced from a tutorial and I was not able to make it my own after successfully implementing it. Reason being that I hardly understood the logic of the algorithm which is a downfall of following a tutorial as the tutorial used was dealing with syntax that I have not learned before without explaining the uses of it. Despite not knowing a lot of the code, I was able to get a basic grasp of how the algorithm works but not enough to improve on it myself.

It did allow me to visually see how BSP works by splitting up a square into multiple squares which will then be connected via a corridor once the algorithm is fully fleshed out. As of now, I have halted the work on this algorithm and have moved to explore Room based algorithm.



*Image showing very early results of the partially implemented algorithm.*

---

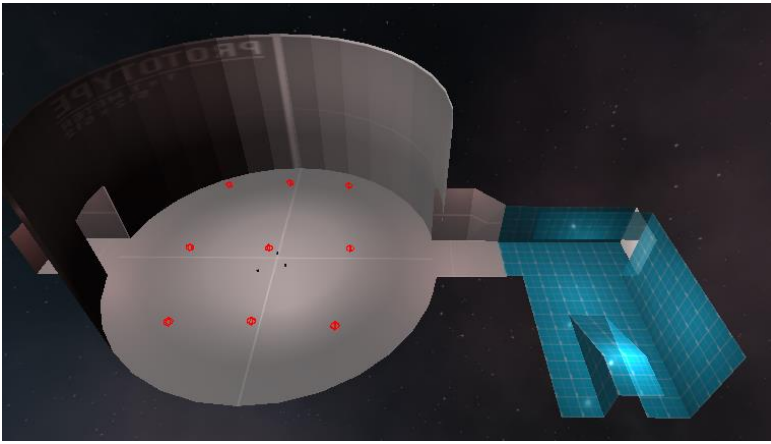
### 5.2.2 PREDEFINED ROOMS ALGORITHM

Before starting, I had the impression that I would be able to use premade assets from the Unity store as rooms for the algorithm. Upon starting the tutorial, this was false, and I was forced to create each room in blender myself with the exception of the starting, end room and corridors provided by ProjectShasta.

The 1<sup>st</sup> step in this algorithm was to define the nodes which in this case will be the doors of the rooms. Each room will have multiple doors and each door has a [Doorway](#) script to specify it is a door which will be used later to join the rooms at the correct orientation. I position the door to face outwards and to ensure that this is achieved I used a debugging tool to draw a line representing the facing direction.

With the doors set, the placing of the start room was produced which was simply instantiating the start room at position 0,0,0. The algorithm then adds the available doors of the starting room to a list where we will select a door at random to spawn another room.

Spawning additional rooms is done with a `PlaceRoom()` function. It attempts to spawn another room prefab but before it does, it adds all the doors of that room into the `availableDoorways` list. After the current room to be spawned has all its doors added it also adds the doors to another list containing doorways of the room to be spawned. After, it loops through each element in the `availableDoorways` list and all the doors of the room to be spawned. The algorithm proceeds to position the new room using its doorways and matching the position and orientation of an available doorway (an existing room that has a doorway not connected).



*An image of a room being connected to the starting room.*

The function to place rooms is called multiple times depending on the number of iterations set for the generator. By default, I have set the minimum iteration to be 10 and maximum to be 15. The script selects a random whole number between the 2 values and that is how many times a room will be placed. Once the last room has been placed, the algorithm spawns the end room which is similar to PlaceRoom() function but instantiates the ending room every time.

The algorithm is not complete as rooms overlap each other quite frequently. The tutorial gives an insight on how to check for overlaps by using mesh colliders and bounding boxes of the room while seeing if the different rooms spawned have their colliders touch another room's colliders. If a collision has been detected then it means an overlap has occurred which results in the generator resetting which destroys every room game object and reruns the functions to spawn the rooms again. The method provided by the tutorial was unreliable and still caused overlaps to occur. To fix this issue I decided to tweak the way the algorithm detects overlap.

For each room prefab I have added another child game object which will hold multiple box colliders. Based on these box colliders I can create the overlap box which will take the shape of the box colliders. The box colliders have been positioned and resized so that it covers most of the room to ensure that another room does not overlap it. This method performed much better as after testing I saw little to no overlaps and even the results from the surveys report that there were rarely any overlaps seen upon testing.

While testing the algorithm, I felt like the Boss room that I created should spawn once so to achieve this I came up with my own code. When placing rooms, I 1<sup>st</sup> check if a game object with the tag "BossRoom" has been spawned. If the boss room does exist, then only select the rooms in position 4 and onwards in the room list as boss room is not included in that range. Else, select rooms from 0 and onwards.

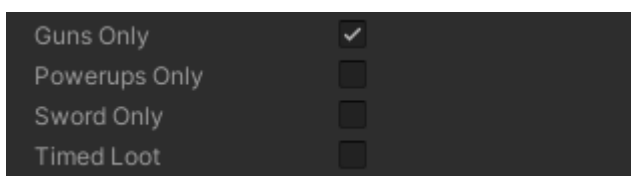


*An image of the rooms lists in the inspector. The script will grab a random room from this list when PlaceRoom() has been called.*

Once the structure of the level has been finalised, it was time to spawn objects. With each room prefab, I added game objects to act as Enemy spawner and Item Spawner. Each have their own spawner script created all by me. The spawners have child game objects specifying the positions at which the enemies or items will spawn. They are added as a child of the spawners to easily add them to an array in the script.

Once I have added each spawn point and positioned them I then add into the spawner script to add these spawn points to an array. Alongside this array is another array created for the specific Spawner for example, EnemySpawner will have an enemy array and ItemSpawner will have multiple types of arrays such as PowerUp array or Guns Array.

The enemy spawn was as simple as iterating through the spawn points using a for loop and spawning an enemy from the array randomly. As for the ItemSpawner, this was more complicated due to there being different types of items. Each room will be configurable to allow specific types of loot to spawn. If guns have been selected then it will spawn only guns and if guns and power ups have been enabled it will spawn both. This is handled by if statements and spawning the items is the same as how enemies are spawned – using a for loop.



*An image of the settings available for each room prefab.*

Due to how enemies were implemented, I had to adapt the level generator to have the rooms baked by the NavMesh component. I created another game object along with my own [NavMeshBaker](#) script which references the level builder script to check if the level generation has completed. This was



simply calling the function `hasRoomFinished()` which returns a bool saying if the generation process has completed. The `NavMeshBaker` script then adds all the rooms to a list by finding objects of type `Room` and using the lists `Add()` function. I then add all the rooms mesh's to a list by getting the 1<sup>st</sup> child game object of the room which was the mesh. Once all rooms have been added, the script then calls `BuildNavMesh()` function which bakes the meshes allowing enemies to move



*An image showing final results with each room being baked (Baked rooms represented with blue surface)*

---

### 5.2.3 PLAYER

#### 5.2.3.1 MOVEMENT

The player is created and implemented as a prefab. With this prefab we are able to refer to it in the Level builder and spawn the player at the starting room by using an empty game object transform position upon finishing level generation. Inside the prefab, the player contains multiple components and scripts to handle the functionality of the game.

Some of the more notable components are Character controller which handles movement of the player according to its surrounding environment and `Player` script. This stores all many of the player's parameters such as maximum health, walk speed, jump force and gravity. The script also handles translating key presses into axis values which is used to move the player in the desired direction. With Unity's input manager, we can get the horizontal or vertical axis and by default, these axis are mapped to the W,A,S,D keys or arrow keys.

`Player` script also includes the jump function which works very similar to moving. When a button is pressed which in this case by default is spacebar, the transform of the object is changed. This time I change the y position of the object instead of the x and z position. I used a formula which has been provided by Unity documentation [22] that includes the jump force which can be increased to have the

player object jump higher and gravity which is used to allow the player to float (or drop instantly depending on the value of the gravity variable) values. This increases the player's Y position and makes it float accordingly.

I then let Unity's physics engine to make the player drop after reaching its peak height from jumping. This is achieved due to the player object having a rigid body component. I included an if statement to set the Y position to 0 if the velocity gets below 0. This is to stop the player from falling through the floor.

With the jump, I ran into a bug which allows player to infinitely jump. To resolve this, I implemented a ground check using a sphere collider on the bottom of the player object. This is used to check if the player is grounded so the player cannot jump infinite number of times in the air.

The snippet below creates a small sphere which returns true or false when it touches a layer. In this case, the Ground layer. I then allow the player to only jump if the sphere is touching ground. This results in the player being limited to a single jump and removes the ability to wall jump or jump infinitely in game.

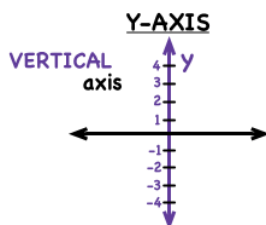
```
// Create a small physics sphere and checks collision with sphere and any Layers set to groundMask Layer and returns true or false  
isGrounded = Physics.CheckSphere(groundCheck.position, groundDistance, groundMask);
```

#### 5.2.3.2 CAMERA

To achieve a first-person camera perspective I 1st attached the camera to the player object so that it moves with the player. I then position the camera at eye level, and I get a result similar to human eyes.

Since the camera and object are different game objects, I add a new script called [MouseControl](#) which handles the rotation of the camera so the player can look around.

In the script, I use similar code for the movement. I get the axis value however with a mouse the axis value can be negative or positive numbers just like the graph below.



As the axis moves, I rotate the camera based on the values of the axis multiplied by the mouse sensitivity (This determines the speed at which the camera rotates) and Time.deltaTime. I also clamp the camera rotation so that it cannot go pass a certain value. This stops the character from looking behind itself. While rotating the camera, I also rotate the player, so the player is facing the same

direction as the camera. This was all produced following the tutorial mentioned in the Camera methods section.

---

#### 5.2.4 COMBAT

The player has access to a basic but solid foundation of a combat system that can be further improved in the future. The combat system revolves around using guns and swords.

The player can equip 2 guns of their choice and equip a melee weapon for a backup in case the player uses all the ammo. This feature has been produced following a tutorial as mentioned previously but adapted by me to work with the [SwitchWeapon](#) script. As the tutorial only allowed equipping of 1 gun, I had to tweak it so that the player can equip multiple. This was done by having if statements, if the player presses 'E' next to a weapon, child the weapon to the weapon slot if the weapon slot does not have a child already. This then results in the weapon taking the position of the weapon slot which has been positioned beforehand so that it follows the 1<sup>st</sup> person perspective tradition. The only issue with this upon testing is that when equipping 2 weapons, the 2<sup>nd</sup> weapon kept childing itself to the 1<sup>st</sup> weapon slot meaning that 2 guns will be active at the same time. To solve this I had to dedicate each weapon slot to a different key press which gave me the desired results albeit the controls do not feel great.

The player attacks or shoots with the left mouse button. This shoots bullets if a gun is equipped or performs a melee attack if a melee weapon is equipped. The shooting revolves around hit detection in order to do damage to the enemy. There are 2 types of hit detection used – Ray casting and Physics hit detection.

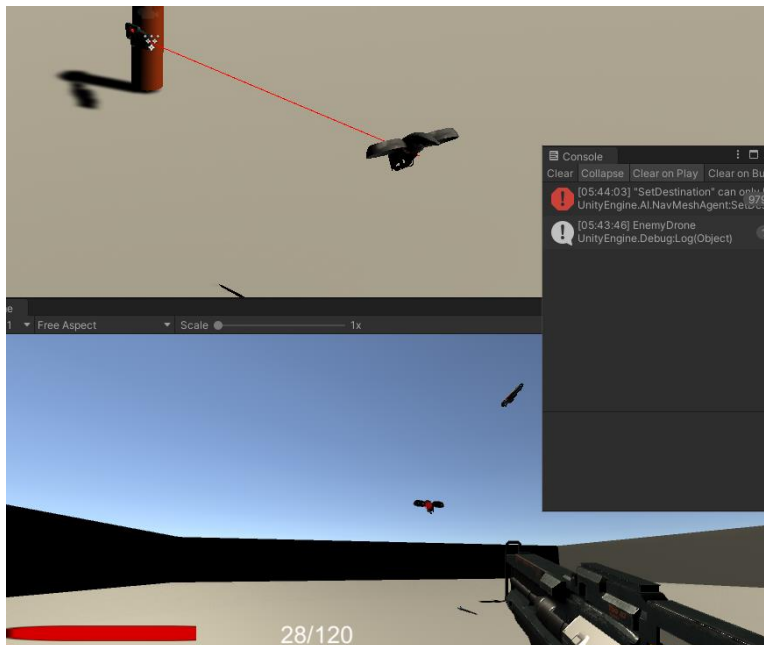
Ray casting is commonly used in video games to determine line of sight of a player (Check if a player is looking at a certain object) or to determine where the projectile will go. Majority of the ray casting logic was produced following the tutorial Dave/Gamedevelopment but adapted to suit my preference.

Upon shooting with the left mouse click, the game casts a straight line (Ray) going forward in 1 direction up to a range that is specified. This ray starts from the centre of the camera and this verifies whether an object is hit by the ray that was created.

```
if (Physics.Raycast(playerCamera.transform.position, shotgunSpread, out hitInfo, shootRange, layerMask))
{
    // Impact VFX
    GameObject impactVFXObject = Instantiate(impactVFX, hitInfo.point, Quaternion.LookRotation(hitInfo.normal));
    Destroy(impactVFXObject, 0.75f); // Destroy the impact VFX after 2 seconds.

    // Damage enemy
    Enemy enemy = hitInfo.transform.GetComponent<Enemy>();
    if (enemy != null)
    {
        enemy.TakeDamage(damage);
    }
}
```

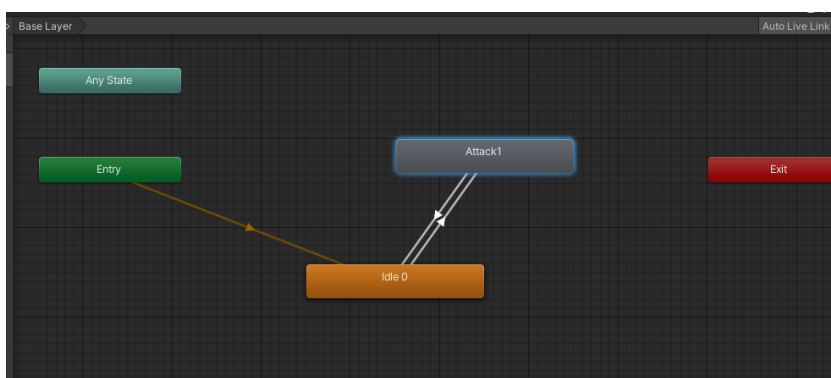
The tutorial followed creates an if statement to see if the ray is touching anything and I send the object information that the ray has touched into the variable hitInfo. I can use this variable to check for specific object and make the game do something when the ray hits it e.g. deal damage to the object.



*Image above shows a ray (Red line) touching an enemy drone and outputting the name of the object in console.*

Some guns have different properties like the range at which the ray will travel for example the shotgun has a much shorter range as naturally in FPS games shotguns have short range but shoot multiple bullets. To achieve multiple rays coming from a shotgun I use a for-loop to loop number of times with each ray starting position being randomised so simulate a shotgun bullet spread. This for loop code was produced by me and added on top of what the tutorial produced.

Raycasting is also used for the melee weapons the same way it is used for guns. The only difference being the range at which the ray travels when attacking as well as an animation to represent the sword slashing. The animator controller for the sword then manages the different states and transitions such as attacking from idle animation once the left mouse button has been pressed.



The other type of hit detection is Physics detection. This method is often used when the gun fires a visible but slow traveling projectile like rocket launcher or a laser. Compared to raycasting, this uses colliders and physics to move the projectile at a given speed. The gun that uses physics detection in the project is the Energy Cannon gun (And enemy laser projectile) which fires a ball of energy from the centre of the camera in the direction the camera is looking towards (transform.forward) at a slow speed but does immense amount of damage upon contact.

In order to move the projectile, I simply update it's transform position by adding the forward direction of the bullet by the speed at which I want the bullet to travel and Time.deltaTime. For optimization, I added an if statement to destroy the projectile after a period of time has passed to conserve memory incase the bullet does not collide with anything.

Colliders are added to the projectile which will allow for collision detection to work. When this collider touches another collider the [EnergyBullet](#) script attached to the projectile creates a visual effect similar to an explosion and specifically calls takeDamage function of the enemy when the bullet collides with a collider attached to the enemy as seen below.

```

10 Unity Message | 0 references
private void OnCollisionEnter(Collision collision)
{
    // If we want collision to occur when specified object is hit add "if (collision.gameObject.tag == "Something")"
    GameObject impactVFXObject = Instantiate(impactVFX, transform.position, transform.rotation);
    DestroyObject(gameObject);
    Destroy(impactVFXObject, 1.7f);

    // Damage player
    Enemy enemyObject = collision.transform.GetComponent<Enemy>(); // Gets Player component when collides Player - Allows us to call functions in Player script.
    if (enemyObject != null) // If playerObject HAS FOUND the Player component and does not equal NULL
    {
        //Debug.Log("I'm hit");
        enemyObject.TakeDamage(damage);
    }
}

```

For guns, the player is able to reload by pressing the 'R' key in order to increase the clip amount. For the player to reload the gun there needs to be ammo in the magazine. The magazine value gets reduced by the number of bullets shot when the player reloads. There are if statements in the [Gun\\_raycast](#) script to ensure that clip amount cannot exceed max capacity and magazine size cannot go above max capacity when picking up ammo or below 0 when reloading. I store the number of bullets shot in a variable which is used to increase the clip size back to maximum amount and decrease the magazine size as seen in the snippet of code below.

```

{
    tempclipSize = gunClipSize; // Temp variable = the gun's clip size that's been set
    clipDifference = tempclipSize - bulletsRemaining; // clip difference is temp clip size minus bullets remaining
    bulletsRemaining = bulletsRemaining + clipDifference; // Add the difference to bullets remaining
    magSize -= clipDifference; // Subtract the difference to magSize

    // If statement to stop magSize from being minus value.
    if (magSize <= 0) { magSize = 0; }

    reloading = false;
    anim.SetBool("isReloading", false);
}

```

---

## 5.2.5 NPC AI

### 5.2.5.1 ENEMIES

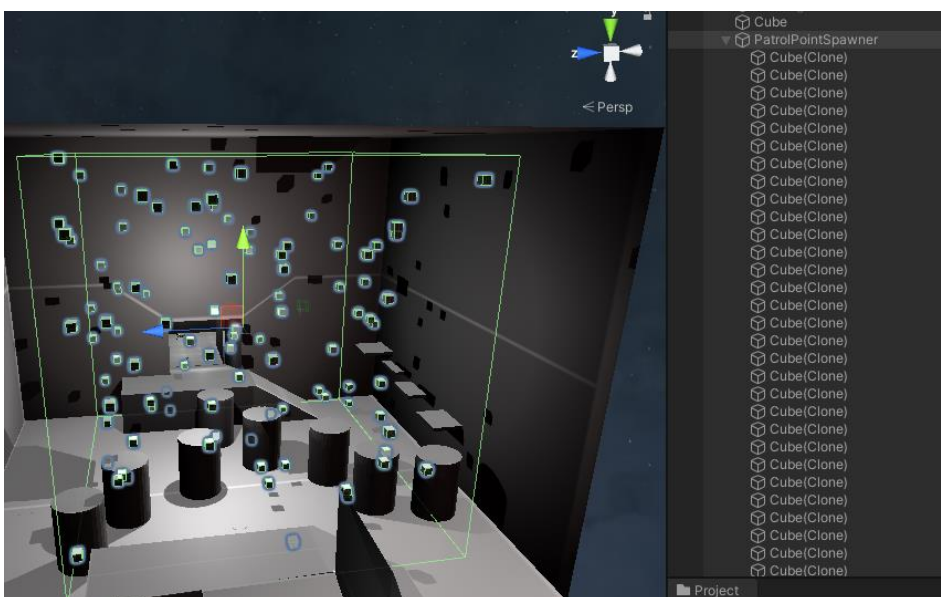
---

Upon creating enemies, I had a rough idea of Finite State machines however I still needed guidance. As stated in the methods section for enemies, I followed a tutorial provided by Dave/GameDevelopment which helped me produce Patrol, Chase, and Shoot state and introduce to me a new component - NavMesh.

However the resulting code for patrol did not perform as well as I would have liked. So I experimented with my own way to patrol rooms. The initial patrol worked by specifying what is “walkable”. Done by using NavMesh package which handles all the pathfinding algorithms. I applied a NavMeshSurface on every room to tell the enemies that this is walkable. The enemies are given a NavMeshAgent component which is needed to have the enemies move around on a NavMeshSurface. With the NavMeshAgent component, I am able to change speed at which the enemy travels, radius at which the enemy avoids walls or other objects and more. So once enemy can now move around (not without its destination target) I had to add patrol points. The use of NavMesh component was not the issue but rather the different points the enemy will travel to.

In order to achieve a natural patrol I randomly placed patrol points within a box collider and placed the box collider inside the room. The script then spawns point randomly using the box colliders X,Y,Z bound values. The maximum range the random number can go up to depends on how big the box collider is resulting in points spawning only within the box. The existing patrol points are randomly selected as the destination for the enemies to move towards.

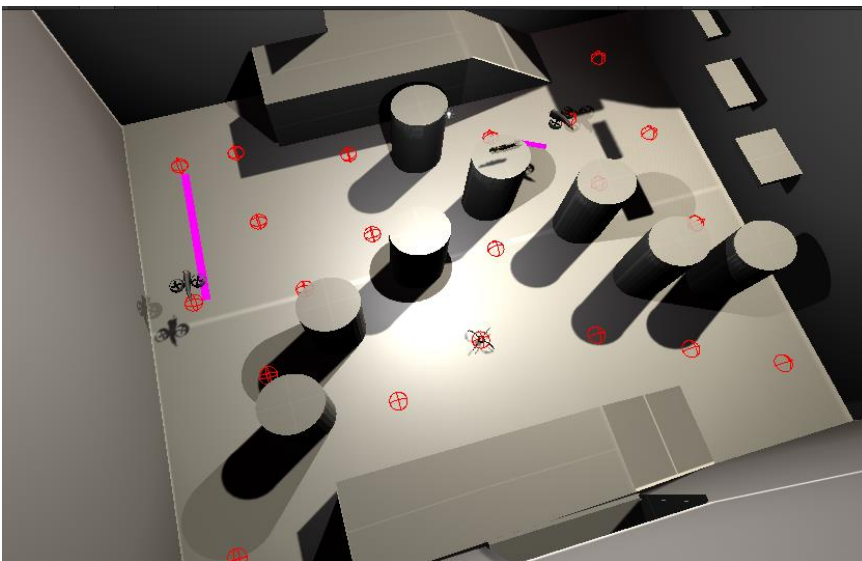
*Early output of the initial patrol point method.*



The initial method did not work as expected. It did ensure patrol points randomly spawned within the box however the issue comes with how the room is made. Some rooms have pillars or walls in the middle of the room and the patrol point occasionally spawn inside them causing the enemy to move into the walls infinitely as it cannot get close enough so that the patrol points get destroyed. Another issue was that Spawning patrol points at different heights would cause issues with enemy types that do not fly.

To simplify patrolling and have enemies roam albeit not naturally is to create set patrol points in the room prefab, retrieve them and add them to an array. This makes it much simpler and allows the enemy to roam much more reliably without any issues. To ensure that flying enemy types give the illusion of flying, I change the NavMeshAgent base offset so that they levitate quite high while still being able to patrol the set points. To try and add more randomness, I made the enemy select randomly select a patrol point to move towards after arriving at its previous patrol point. Upon arriving at the destination, the enemy waits for few seconds. This waiting feature was implemented following tutorial provided by Table Flip Games [23]

*Final version of patrol state. (Patrol points in red for debugging)*



As for the other states, they worked as expected and fit the game quite well. In addition to the states found in the tutorial, I added another state called Attack. This is different to the Shoot state found in the tutorial and is used for close range attacks to allow for different enemy types. This state upon player getting really close and activating it, the state calls the TakeDamage() function from the [Player](#) script causing the player to take damage.

#### 5.2.5.2 MINI BOSSES

The mini bosses that spawn in “Boss rooms” are implemented very similarly to regular enemies, inheriting many of its states while having its own additional Enrage state. The Enrage state is activated



when the boss's health has reached 40% of it is maximum health or less. This is done with an easy if statement and simple mathematical calculation. Once this has been satisfied, the "Enrage" function is called which sets the properties of the boss to a higher value than default.

After implementing the mini boss, I decided to experiment with the attack state as I felt that it should offer more upon getting hit. I added hitboxes at the point of contact that get activated upon the "Attack" function being called. The Player takes damage upon the hitbox entering the players box collider. This was implemented only on the boss to give a knockback effect which did not work as intended and instead of knocking the player back it propelled the player upwards. Despite knockback not working, I left the attack state as it is (With hitboxes) since it can be further developed in the future and hopefully the knockback feature will work as intended.

---

### 5.2.6 LOOT

Loot in game contains power ups or weapons and is found upon defeating enemies or spawned throughout the level. Enemies have been configured to spawn powerups only.

In order to create an effective and good-looking loot drop system when an enemy is defeated I followed a tutorial as mentioned in the methods section to get a stylized loot effect so when loot is dropped, it propels itself upwards and falls down. The tutorial was only used to get the drop motion as the tutorial does not spawn loot upon an object being destroyed but upon activation. So I used what I have learned from this tutorial to get the motion and then worked on the actual spawning of the loot upon enemy death.

After following the tutorial for spawning loot, I tried calling the CreateLoot function in [SpawnLoot](#) script upon enemy death hoping it would work but it did not. The reason was because the script searches for the position to spawn the loot and that position is stored as a game object which is a child of the Enemy object. So when the enemy dies and destroys itself, it destroys that position.

To solve this, I had to change the way the [Enemy](#) script worked. I disabled the mesh when enemy health has reached 0 while keeping the object alive. In the image below you can see that once enemyHealth has reached 0, the Mesh object (1<sup>st</sup> Child of the Enemy object) gets disabled, and the spawn loot function gets called. I also freeze the enemy object so that it cannot move when spawning loot resulting in the loot all spawning relatively close to each other and at the position the enemy died.



```

if (enemyHealth <= 0)
{
    sound.PlaySound("explosion");

    //Instantiate explosion VFX
    GameObject impactVFXObject = Instantiate(deathExplosion, transform.position, transform.rotation);
    Destroy(impactVFXObject, 1.7f);

    hasDied = true; // Set hasDied to true so object cannot call or execute any functions
    transform.GetChild(0).gameObject.SetActive(false); // Gets the child of the object which in this case is the Enemy mesh and DISABLE IT
    transform.GetComponent<BoxCollider>().enabled = false;
    GetComponent<Rigidbody>().constraints = RigidbodyConstraints.FreezeAll;
    GetComponent<SpawnLoot>().setSpawnLoot(true); // Call function to spawn loot.
    Invoke("DestroyObject", 2);
}

```

In game this would result in the enemy model not being visible upon enemy death however the actual enemy object is still in the game, but their colliders have been disabled. This allows the loot system to spawn the loot at the object's position. After 2 seconds, I then destroy the enemy object.

Another issue I had after following the tutorial was that objects would constantly move when hitting the ground which was not seen at the end of the tutorial video. So I decided to fix this by adding the `OnCollisionEnter` function in the [LootDropMotion](#) script and checked if the object the loot has collided with is the ground. If the colliding object is the ground then remove any velocity previously applied to the object by using the statement "`rb.velocity = Vector3.zero;`" as seen in the image below.

```

if (hitGround == true)
{
    Debug.Log("Hit the ground");

    //Enable and disable R8 properties so that Unity
    rb.useGravity = true;
    rb.isKinematic = false;
    rb.velocity = Vector3.zero;
    this.enabled = false;
}
}
@ Unity Message | 0 references
private void OnCollisionEnter(Collision collision)
{
    if (collision.gameObject.layer == 9)
    {
        hitGround = true;
    }
}

```

This worked as intended however I wanted to optimize the loot system for the future by adding a timer so that they are destroyed after specified time has passed.

I tried doing a timer in the [LootDropMotion](#) script however this timer did not work due to the [LootDropMotion](#) script disabling itself upon hitting the ground to remove any forces added to object. I then created another script called [LootTimer](#) which has an `OnCollisionEnter` function and a timer.

```

public class LootTimer : MonoBehaviour
{
    public float lifeDuration = 5f;
    private float lifeTimer;
    private bool hitGround = false;
    // Start is called before the first frame update
    @ Unity Message | 0 references
    void Start()
    {
        lifeTimer = lifeDuration + Random.Range(5f, 9f);
    }

    // Update is called once per frame
    @ Unity Message | 0 references
    void Update()
    {
        //Check if loot has hit ground
        if (hitGround)
        {
            lifeTimer -= Time.deltaTime;

            // if lifeTimer less than 0 destroy the game object
            if (lifeTimer <= 0)
            {
                Debug.Log(gameObject + "Destroyed");
                DestroyObject(gameObject);
            }
        }
    }

    @ Unity Message | 0 references
    private void OnCollisionEnter(Collision collision)
    {
        // If loot collides with object with layer 9 (Ground)
        if (collision.gameObject.layer == 9)
        {
            hitGround = true;
        }
    }
}

```

This script is purely just for destroying the object and the timer starts upon object detecting collision with the ground. After timer has reached 0, object destroys itself successfully.

Once the loot system was finished I tweaked how the magnetic effect produced following the tutorial mentioned in the methods section. I added Overlap sphere seen for the enemy states but for this script, it activates the magnetic force to move the powerup towards the player.

```

playerInMagnetRange = Physics.CheckSphere(transform.position, magnetRange, Player);

if (playerInMagnetRange)
{
    // Adds force to object and moves it towards the player object
    GetComponent<Rigidbody>().AddForce((player.transform.position - transform.position) * forceFactor * Time.smoothDeltaTime);
}

```

After implementing all the stylized effects and spawning of the loot, a script is added to each powerup, called [PowerUps](#). This handles the collision detection between the powerup and player and updates the player's properties or gun properties upon a successful collision.

---

## 5.2.7 UI

### 5.2.7.1 HEADS UP DISPLAY (HUD)

The HUD is implemented as a Canvas object allowing images and text to be displayed. The health bar is tied to the players health through the [HealthBar](#) script attached and fills the bar depending on the amount of health the player has left which results in the health bar working perfectly.

As for the ammo count, it is represented through text and the values passed in is taken from the Gun's bullets remaining and magazine capacity variable. This is done by attaching the [AmmoCount](#) script to each weapon slot then getting the child object which is the gun and calling the getter functions to

retrieve the bullets remaining and magazine capacity. The script then updates the text on the HUD with the values it has retrieved.

#### 5.2.7.2 MENUS

---

The majority of the main menu has been imported from the Unity Asset store in which I have tweaked to suit the project. The asset comes with the basic New Game, Continue, Settings (Which does not work in the final project build) and Quit button. I have changed the Continue option to VR Mode. I then changed the flow of the menu so when New Game has been selected, it shows a difficulty panel so the player can select either normal or hard mode, when VR mode is selected, it loads the scene for VR. Depending on the mode selected, a difficulty multiplier variable is set and is set to not destroy upon changing scenes. This variable is used to modify the enemy's properties to increase it when hard mode is selected.

The Pause menu which is available in-game is mapped to the "Escape" key and allows the player to pause or resume the game, opens a new panel displaying controls/instructions or quit to main menu. This pause menu is also created as a Canvas and each button responds to mouse click. All images and texts in pause menu have been provided with Unity Engine.

---

#### 5.2.8 VR MODE

Upon starting development for VR mode, I had to refresh my knowledge on VR development. For the hands interactions with objects I followed the tutorial provided by Valem [24] which allows the player to grab objects, call functions from a script attached to that object when an input is found, jump, move the player around and fetch the inputs all using the OpenXR plug-in. Each weapon and power up have been adapted for VR mode for example, guns will only shoot if the trigger of the right controller is pressed, and a ray emits from the gun's muzzle position to help players see where they are currently aiming. Also the sizes have been modified to suit the VR perspective. 1 change I made to grab interactions was setting the object that has been grabbed a child of the hand. This makes it easier to reference the object and was done within the [XR Grab Interactable](#) script attached to each grabbable object. I also added a Boolean that is set to true if the object has been selected (gripped in hand) or false if the object has been let go of (Let go of grip) which allows me to check whether a weapon is being held or not.

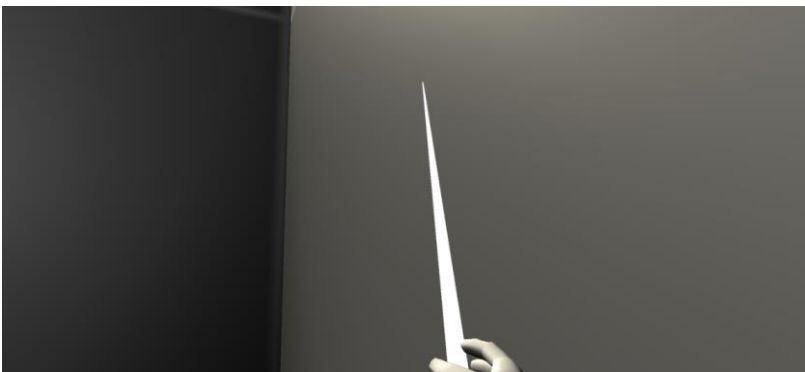
Once the basics of VR were produced I then added my own VR implementations to adapt to the game for example, normal mode allows the player to switch between weapons and reload them. In VR, there is no scroll wheel or various keys so to add these features I had created a holster which rotates as the camera rotates. The rotation was achieved following the tutorial by My. Pineapple Studio [25] but the logic was produced by me.

The holsters have box colliders that have been position by the side and back of the player (Sword being only object equip able at the back holster). The [Holster](#) script also checks if that weapon is a gun or sword to specify if the holster can hold that weapon. Once that is done, I use the `OnTriggerEnter` function and after a specified amount of time has passed while the weapon is in the box collider the weapon is deemed as attachable. To represent that the gun is attachable and has been held long enough in the holster, I have added an outline that changes colour. This outline effect was taken from the Unity Asset store and is provided by Chris Nolet for free. The weapon will only position itself to the holster's position only if the player lets go of the grip button while the weapon is within the box collider meaning I the Boolean I added in the [XR Grab Interactable](#) script is false.

The reload function has been created similarly to the holsters with the outline included to represent it being in the reload state but instead of the gun taking position of the box collider for reloading, it calls the `Reload` function of the gun.

As mentioned in Methods section, I had to implement a way to relieve the player from motion sickness and in order to do that I implemented teleportation movement provided by Valem [\[26\]](#)

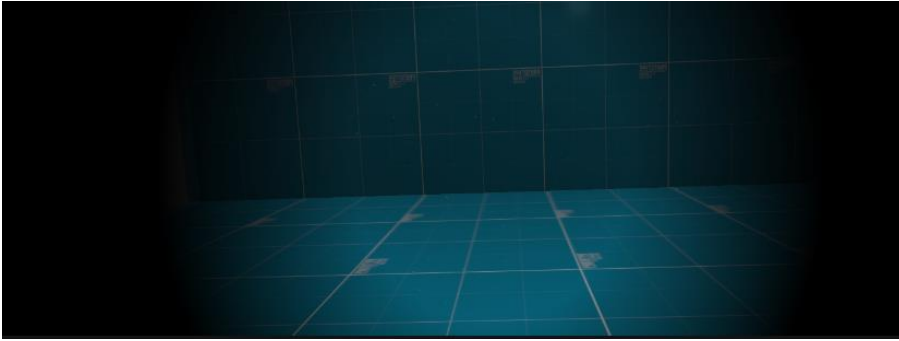
The teleportation system was as easy as adding the teleportation provider component to the XR Rig game object, adding a ray interactor to specify where you want to teleport to and specifying what the player can teleport too. After implementing this there were many issues with how the teleportation works with the rooms. Since the rooms have all been created as 1 whole mesh it was hard to separate from what can be teleported to and what cannot be which resulted in the player being able to teleport to the wall and getting stuck inside the wall.



*Image of player being able to teleport to wall (An area that can be teleported to colours the ray in White)*

The teleportation feature was scrapped in the final project and was replaced by a tunnelling system. Due to time constraints, this tunnelling effect was imported from the Unity Asset store and was provided for free by SIGTRAP. This relieves motion sickness by darkening the sides of the view when moving which has been researched to help with motion sickness. I added to this my own code to

toggle this effect on and off in the pause menu.



*An image of the effect in play.*

The heart of VR is the motion controls plus tracking, and I wanted to demonstrate this. The perfect feature to showcase them was the sword combat. This was easily done by enabling and disabling the box collider trigger of the blade when the velocity of the controller movement is above a certain threshold and calling enemy take damage function upon touching an enemy. This allows players to swing freely like they are actually sword fighting and the results turned out better than expected allowing players to reach out and swing at an enemy and successfully damage them.

---

#### 5.2.9 EVALUATION

After receiving results from the survey, it is evident that the project does indeed change the level structure each playthrough however due to limited number of rooms, enemies, and items available, each playthrough does not feel fresh.. This can be remedied over time by adding more or by following the same method as Strafe – adding joining 2 rooms together but not through doors, by walls. Due to the time constraints, I was not able to create various unique rooms or enemies.

Another downfall of the algorithm pointed by me after testing was that sometimes level generation took too long. For a game with little content, the game is expected to load quite fast but due to the level generation constantly resetting to avoid overlapping, the loading time is significantly increased.

Detailed feedback was actually targeted towards bugs and how the game feels with some bugs being solved in the final build and others not. While the gameplay all worked fine, it was reported that the looking around with the mouse felt bad and with no option to configure it, it was hard to get used to. This can be easily fixed by having an option to input a number to specify the speed at which the camera rotates. The equipping of guns felt quite peculiar as in FPS games, they have a universal button to equip weapons but in this project I had to map each weapon slot to their own key due to bugs.

For VR mode, the feedback received was quite positive. Majority of the features worked in VR and I am quite proud with what has been achieved. The gun holsters had 1 problem where the range at which you can grab the gun is quite large so trying to grab a gun from the floor with a gun already holstered, the game thinks you are trying to grab the holstered gun. There were problems with the sword combat

not working on smaller enemies due to it not reaching, flow of the game for example when starting the game it would have been nice to be able to navigate main menu in VR too. Other gripes were to do with how the game looked in VR, the guns were too small but were the right size when in hand. Most of these could have been polished but due to time constraints they have not been.

The project has very few bugs that occur when playing. Some notable bugs outlined from testers was that the menu does not operate after death and requires the player to Alt+Tab out to another application and switch back to the game. The cause of this issue is unknown as the issue does not occur every time. Another very rare visual bug is rooms overlapping. This is very minimal as only the corners of a room can be seen clipping into another, the game can still be completed if this bug were to occur.

## CHAPTER 6: CONCLUSIONS AND DISCUSSION

### 6.1 OVERVIEW

Overall, the project does succeed in implementing a working PCG algorithm which also includes a method to populate rooms with objects albeit not randomly. The use of the class diagram was helpful yet restrictive resulting in me not following it 100% throughout as there were many changes due to unpredicted bugs and features that I did not think about when producing the diagram. This shows that a UML diagram may not be so useful when creating games as

The primary objective was to give an alternative solution to hand crafting levels while offering more replayability.

In my opinion, I feel like the objective has been 50% achieved. The reason why I believe this is because while the algorithm does generate new level layouts every playthrough while spawning random objects and enemies in each, it lacks variety which is mainly due to time constraints. With such limited content, the game does not feel new every playthrough or interesting enough to notice that everything is different. As such, the project could have achieved the primary objective fully if more time were given.

With the sub objective – VR Mode, this was fully achieved even though how well it feels may be subjective. The VR mode definitely allows players to experience the game in another perspective and does add some value to the game while retaining all the core implementations.

For core implementation goals, almost all of them have been achieved with few bugs and a few features being left out due to time constraints and insignificance.

Overall, the project has met most of the objectives and can definitely be improved in the future. Hopefully, an algorithm to procedurally generate and spawn items can be explored and implemented to add more variety.

---

## 6.2 LITERACY REVIEW – RELEVANCE

If there was an opportunity to redo the project there are things that I would have done differently beforehand. Not to say that the results unsatisfactory but there are many things that I felt like could have been better.

Firstly, researching the different ways of PCG was definitely helpful to implement the final PCG in the project however the project could have better attempted to understand Binary Space Partitioning as in the project, the results outputted for BSP was not fully explored and understood which may have ultimately resulting in a better level structure than Room based algorithm.

This would require other algorithms that handle procedurally spawning objects within a room which would enhance the algorithm produced.

The project benefited a lot from researching into existing PCG algorithms to imitate the algorithm used in the game “Strafe”. It has been helpful for evaluating how well the PCG is and what further implementations can be added to improve the algorithm.

For VR, researching into other VR games of the same genre would have been helpful as it would inform me of the actions needed to do certain things such as Reloading or holding multiple guns. Both the reloading function and holding multiple guns in the final project has been fully improvised by myself which may or may not be the most immersive way.

In general, I wanted to explore the use of PCG and try my attempt at creating a VR game which is in the form as a separate mode and for both objectives I feel like I have achieved something to be proud of, especially given the time constraints and lack of knowledge in certain areas.

**Appendix A. Project Definition Document****Project Definition Document****Cover Sheet**

**Course:** BSc Computer Science with Games Technology

**Project Title:** FPS game with procedurally generated levels in Unity

**Student Name:** Kevin La

**Student Email Address:** [kevin.la@city.ac.uk](mailto:kevin.la@city.ac.uk)

**Consultant name:** Eddie Edwards

**Consultant Email Address:** [philip.edwards@city.ac.uk](mailto:philip.edwards@city.ac.uk)

**Clients:** N/A

**Project Origin:** The project idea was thought up by me.

**Arrangements for proprietary interests:** Due to the project being created in Unity Personal edition, all content made with Unity is fully owned by myself and requires no further licensing documentation. Throughout the project, I may be required to use other online resources available and free assets sourced externally to achieve my objectives within the time frame and of course the owner of those assets/resources will be credited.

**Other promises:** I aim to add Virtual Reality Support to the game to further enhance the user's experience and give more value to the game.

**Project Definition Document:** Proposal

**Problem to be solved**

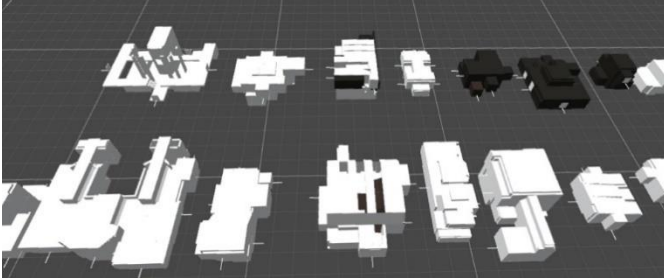
First person shooters have been a popular gaming genre across the world as it focuses on a centred perspective of shooting a gun or melee combat. Many of these FPS games are made with the mindset of having the player really feel like they are in the game as the playable character. They do this by making each character animation such as reloading a gun or swinging a sword true to real life as well as the camera perspective and many other factors (Sound, Gameplay feel). The problem with many FPS games is that despite looking visually appealing and feeling accurate to real life, they lack replayability. Many FPS games are story driven and follow a linear structure meaning that the player has very limited freedom. The game wants you to follow what the story has in store for the player and because of this, completing the game or level once will make subsequent playthroughs boring.

There are examples of FPS games that are non-linear, giving the player more freedom and directing the story based on narrative choices. Games like the Fallout series (Initial release, 1997 to latest, 2018) is a great example of an FPS open world game. Fallout allows the player to explore whatever they desire, do what they want and progress at their own pace. This gives their character an identity as each in-game choice and interaction with NPC's will affect the story, thus making each playthrough different.

In conclusion, having a lot of freedom and shaping the story adds replayability however those games usually require a higher production as it is required to think of a story, create models for the open world and much more, not to mention that Non-Linear story games are best experienced at a slow



pace to absorb everything. Procedural level generation could be an alternative which can be used to give a “*mindless shoot-em up*” feel to an FPS while adding replayability. Strafe is a great example of procedurally generated levels and they achieved this by making “*large collection of rooms that fit together like a puzzle and randomize every play.*” (Thom, 2014). On top of that, the game is fast pace which makes sense as there is no story, and the emphasis of the game is not appearance or narrative design but the addictive gameplay and randomness each play through.



The image above shows a collection of rooms that are pre-made and will be randomly put together to create a level.

The goal of this project is to explore combining the procedurally generated levels with a fast-paced game similar to Strafe to see if all these genres combined adds more replayability while keeping the fun factor with a lower production value as opposed to an open world FPS game like Fallout. On top of that, I will add VR support as an extra feature to further increase the replayability and add a new way of playing.

### Project Objectives

The main objective of this project shall allow the player to play the game at its basic functionality, complete a randomly generated level by reaching the requirements of the level while being filled with objects and enemy AI. After completion, the user can go to the next stage with a completely different level to the first playthrough. The main goal for the user is to reach the highest stage where each stage gets progressively harder. There are many other objectives that I wish to achieve to add more value and fun factor to the game.

- Basic Functionality
  - Character Movement
  - Camera setup
    - First person camera/POV
  - Character combat
    - Projectile weapons
    - Melee attack
  - User interface and HUD
    - Main menu screens.
    - Option screens.
    - HUD Design
  - Procedural Content Generation
    - Creation of predefined rooms.
    - Filling rooms with objects/meshes randomly.
    - End Goal – Climb the tower and get the furthest before dying or time running out.
  - Enemy AI
    - Enemy movement
      - Flying enemies, grounded enemies, Bosses.
    - Enemy Spawn points
- Full functionality
  - Loot system

- Defeated enemies drop usable items for player.
- NPCs
  - NPCs to assist or guide the player.
- Improved content generation
  - Hand crafted puzzles and adding them into predefined rooms.
- Saving/Loading
  - Save and Load game state.
- Mini bosses
  - Bosses randomly spawning in rooms or after a certain requirement is met.
  - Boss AI implementation
- Polishing and implementing extra features
  - Implement VR
    - Add VR controls/camera view.
    - Option to adjust VR settings.
  - Difficulty option
    - Easy, Normal, Hard mode.
  - Levelling system
    - Allow player to level up by earning exp from defeated enemies and learn new skills.
  - Advanced AI
    - Enemy health/damage scaling with Player level.
    - Improved AI

Many of these sub-objectives rely on 3D models and due to time constraints, I decided to use royalty free 3D models for my game. These models will be credited accordingly.

To evaluate these objectives and see if they are working as intended, they will be tested after each section is completed by myself and/or other users. This will be in a form of playing a build of the game with a section of the game functionality implemented and answering online surveys to get feedback on certain aspects of the game. I will then compare their results to the expected results and then decide whether this objective has been met or needs more work.

### **Project Beneficiaries**

The main beneficiary of this project will be myself. Reason being is that, as an aspiring game developer/VR Developer, I will learn how to implement procedural generation while improving gaining experience with VR development and many other areas like AI.

Other possible beneficiaries may include Unity users and other game developers as once the game is fully completed, it may be published in stores or GitHub for free for others to use and tinker with.

### **Work Plan**

For this project, I plan to use Agile development which is the concept of delivering incrementally. As I have sectioned the development process into 3, it would make sense for me to go with Agile development. This will allow me to release an incrementally improved game each iteration and by the end of the last iteration, it will be fully functional and testable by users. (Raluca, 2016

<https://marionettestudio.com/agile-game-development-quick-overview/#:~:text=What%20is%20Agile%20Game%20Development,in%20small%20periods%20of%20time.>)

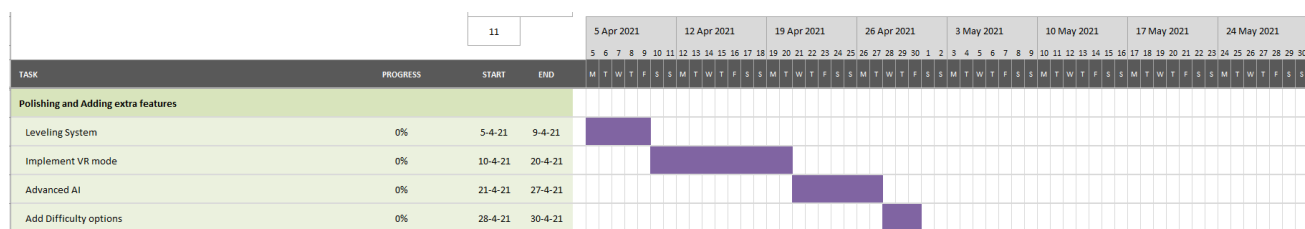
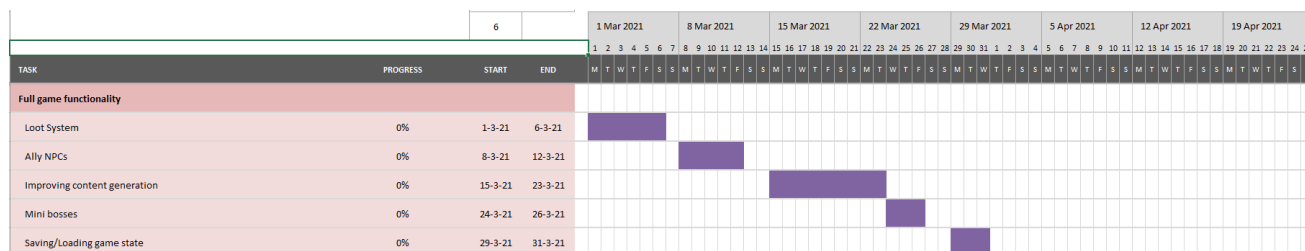
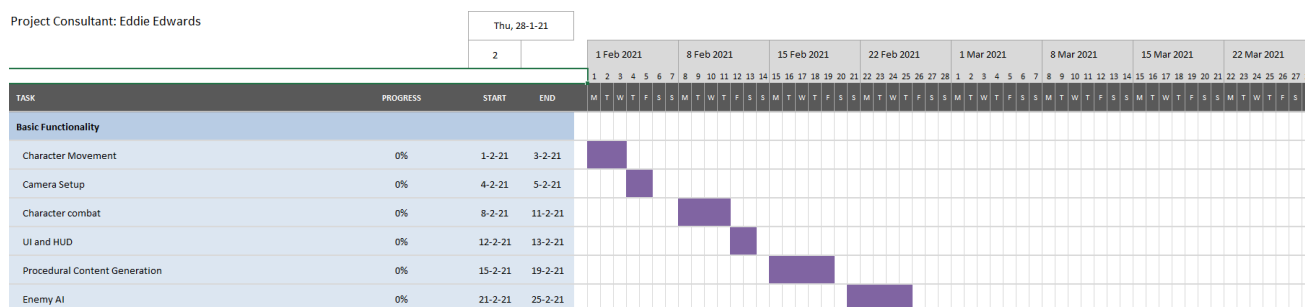
I have created a Gantt Chart to track progress and assign deadlines for each objective in each section of the project.

(Images shown below which are cut up into 3 images due to size issue)

## FPS Procedural Content Generation

Project Lead: Kevin La

Project Consultant: Eddie Edwards



The game I have in mind will be developed using Unity 2019.4 with Visual Studio 2019 being the IDE. As it is being made in Unity, the main language used will be C#. I decided on these tools as I have more experience with C# than C++ and VR support is much easier to deal with in Unity due to their plugin with OpenVR which allows me to develop games on the HTC Vive pro. Unity also has much more documentation and support when it comes to integrating VR controls in a game. For source control, I will use GitHub to manage the project code which allows me to go back to versions of code in case of bugs or failures.

Other tools may be Blender to create or manipulate 3d assets or Photoshop to design the UI and HUD.

## Project Risks

### Risk assessment

*Risk assessment form created in Excel.*

Objective	Likelihood 0 = No likelihood 5 = Very likely	Severity 1 = little impact, 10 = Major Impact	Risk Factor 0 - 5 = Low risk, 6 - 10 = Medium risk, 11 - 15 = High risk	Risk	Mitigation - what can be done about it?
Character Movement	1	10	1	Little to no risk as I have experience in implementing movement. Only risk may be movement may not be as fluid or fast paced as games like Doom or Strafe	Look at documentation online or past projects made in Unity.
Camera Setup	0	10	0	Experienced in this area, especially due to how easy Unity has made implementing a FPS camera view.	Look at documentation online or past projects made in Unity. Can always research if camera needs improvement or tweaks.
Character Combat	2	10	6	Inexperienced with creating an FPS shooter as I have mainly created games with melee type weapons so coding physics for bullets will be new to me.	Look at Unity Learn section on website. Refer to Unity's prebuild FPS project to see how it is implemented.
UI and HUD	0	5	0	I have decent amount of experience creating HUDs and UI for games and Unity makes it pretty easy to implement these.	Sort out Design of HUD and UI as soon as possible when starting this objective so that it can be implemented without any issues.
Procedural Content Generation	4	10	13	No experience on working with Procedural content generation. May take longer than expected to implement.	Revise and study this topic asap to understand the concept. Do tests as soon as the topic is fully understood.
Enemy AI	3	8	10	AI may not be as advanced as I wish. Have little experience in implementing Enemy AI in previous courses.	Take what I learned from previous courses and apply here while also using what I learn from researching.
Loot System	3	6	6	May take a while to implement an inventory for the player as I've never done it before. However having loot spawn randomly is easy and can be done in a short amount of time.	Research inventory system and have it update when picking up randomly spawned loot.
NPCs	2	4	5	Have some experience in creating NPCs such as NPCs that follow the play and assist in defeating enemies however this may not be as advanced as I wish.	Research different types of NPCs in similar genres and learn how to implement similar NPCs in Unity.
Improved Content Generation - Hand crafted puzzles	2	3	4	I've created puzzles in a different engine before (Unreal engine 4) however the only risk I see here is trying to successfully add these puzzles in a predefined room.	Start on this as soon as possible in order to make time for testing this and ensuring everything works according to plan. Optional but beneficial - Research different types of puzzles to add variety.
Saving/Loading	1	2	3	Saving and Loading a game state is a common feature in games so finding resource on this is easy. A small risk I can see is that there may be memory leaks if this feature isn't implemented correctly causing game to crash or saved files to not load.	Research tutorials on how to Save a state and load it. Then implement it ASAP giving time to test this feature.
Mini Bosses	3	7	10	Similar to implementing Enemy AI however this will be slightly more complex as a boss is meant to be much harder with different attack patterns to give a challenge to the player. This may lead to implementing this take longer than expected.	Research how to implement different states for an enemy and implement our own states for the Boss.
Implement VR	4	8	12	VR is something I've touched on but not fully created a project in. I have some knowledge on it however the difficult part is having some of the functionality into VR for example, navigating through menus will be different in VR, especially the inventory system. May have many bugs and could take a long time to implement and polished.	Start research inventory systems in VR and see how it's done. Have a refresh of how to implement Camera and player controls and ensure that the version of Unity supports the latest OpenVR plugin so that I can implement VR into the project.
Difficulty Options	0	2	0	Adding difficulty options is something I'm knowledgeable as I had to implement this in previous projects for other modules.	Implement this ASAP so there is no overlap and allows me to work on more important things earlier.
Leveling System	3	6	10	Inexperienced implementing a leveling system which can lead to feature being trashed or taking longer than expected.	Find resources on this topic ASAP.
Advanced AI	4	5	10	Inexperienced with AI in general so this feature may not be implemented or will take longer than expected.	Refer to Advanced Games tech lecture to see if there's any good documentation or resources. Also research other external online resources to improve AI.

Project Definition document: Research Ethics Checklist

## Research Ethics Review Form: BSc, MSc and MA Projects

### Computer Science Research Ethics Committee (CSREC)

<http://www.city.ac.uk/departments-computer-science/research-ethics>

Undergraduate and postgraduate students undertaking their final project in the Department of Computer Science are required to consider the ethics of their project work and to ensure that it complies with research ethics guidelines. In some cases, a project will need approval from an ethics

committee before it can proceed. Usually, but not always, this will be because the student is involving other people (“participants”) in the project.

In order to ensure that appropriate consideration is given to ethical issues, all students must complete this form and attach it to their project proposal document. There are two parts:

**PART A: Ethics Checklist.** All students must complete this part. The checklist identifies whether the project requires ethical approval and, if so, where to apply for approval.

**PART B: Ethics Proportionate Review Form.** Students who have answered “no” to all questions in A1, A2 and A3 and “yes” to question 4 in A4 in the ethics checklist must complete this part. The project supervisor has delegated authority to provide approval in such cases that are considered to involve MINIMAL risk. The approval may be **provisional** – *identifying the planned research as likely to involve MINIMAL RISK*. In such cases you must additionally seek **full approval** from the supervisor as the project progresses and details are established. **Full approval** must be acquired in writing, before beginning the planned research.

<b>A.1 If you answer YES to any of the questions in this block, you must apply to an appropriate external ethics committee for approval and log this approval as an External Application through Research Ethics Online - <a href="https://ethics.city.ac.uk/">https://ethics.city.ac.uk/</a></b>		<i>Delete as appropriate</i>
1.1	Does your research require approval from the National Research Ethics Service (NRES)?  <i>e.g. because you are recruiting current NHS patients or staff?</i>  <i>If you are unsure try - <a href="https://www.hra.nhs.uk/approvals-amendments/what-approvals-do-i-need/">https://www.hra.nhs.uk/approvals-amendments/what-approvals-do-i-need/</a></i>	<b>NO</b>
1.2	Will you recruit participants who fall under the auspices of the Mental Capacity Act?  <i>Such research needs to be approved by an external ethics committee such as NRES or the Social Care Research Ethics Committee - <a href="http://www.scie.org.uk/research/ethics-committee/">http://www.scie.org.uk/research/ethics-committee/</a></i>	<b>NO</b>
1.3	Will you recruit any participants who are currently under the auspices of the Criminal Justice System, for example, but not limited to, people on remand, prisoners and those on probation?  <i>Such research needs to be authorised by the ethics approval system of the National Offender Management Service.</i>	<b>NO</b>
<b>A.2 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee, you must apply for approval from the Senate Research Ethics Committee (SREC) through Research Ethics Online - <a href="https://ethics.city.ac.uk/">https://ethics.city.ac.uk/</a></b>		<i>Delete as appropriate</i>
2.1	Does your research involve participants who are unable to give informed consent?  <i>For example, but not limited to, people who may have a degree of learning disability or mental health problem, that means they are unable to make an informed decision on their own behalf.</i>	<b>NO</b>
2.2	Is there a risk that your research might lead to disclosures from participants concerning their involvement in illegal activities?	<b>NO</b>
2.3	Is there a risk that obscene and or illegal material may need to be accessed for your research study (including online content and other material)?	<b>NO</b>

2.4	Does your project involve participants disclosing information about special category or sensitive subjects?  <i>For example, but not limited to: racial or ethnic origin; political opinions; religious beliefs; trade union membership; physical or mental health; sexual life; criminal offences and proceedings</i>	NO
2.5	Does your research involve you travelling to another country outside of the UK, where the Foreign & Commonwealth Office has issued a travel warning that affects the area in which you will study?  <i>Please check the latest guidance from the FCO - <a href="http://www.fco.gov.uk/en/">http://www.fco.gov.uk/en/</a></i>	NO
2.6	Does your research involve invasive or intrusive procedures?  <i>These may include, but are not limited to, electrical stimulation, heat, cold or bruising.</i>	NO
2.7	Does your research involve animals?	NO
2.8	Does your research involve the administration of drugs, placebos or other substances to study participants?	NO
<b>A.3 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee or the SREC, you must apply for approval from the Computer Science Research Ethics Committee (CSREC) through Research Ethics Online - <a href="https://ethics.city.ac.uk/">https://ethics.city.ac.uk/</a></b>  <b>Depending on the level of risk associated with your application, it may be referred to the Senate Research Ethics Committee.</b>		Delete as appropriate
3.1	Does your research involve participants who are under the age of 18?	NO
3.2	Does your research involve adults who are vulnerable because of their social, psychological or medical circumstances (vulnerable adults)?  <i>This includes adults with cognitive and / or learning disabilities, adults with physical disabilities and older people.</i>	NO
3.3	Are participants recruited because they are staff or students of City, University of London?  <i>For example, students studying on a particular course or module.</i>  <i>If yes, then approval is also required from the Head of Department or Programme Director.</i>	NO
3.4	Does your research involve intentional deception of participants?	NO
3.5	Does your research involve participants taking part without their informed consent?	NO
3.5	Is the risk posed to participants greater than that in normal working life?	NO
3.7	Is the risk posed to you, the researcher(s), greater than that in normal working life?	NO
<b>A.4 If you answer YES to the following question and your answers to all other questions in sections A1, A2 and A3 are NO, then your project is deemed to be of MINIMAL RISK.</b>  <b>If this is the case, then you can apply for approval through your supervisor under PROPORTIONATE REVIEW. You do so by completing PART B of this form.</b>		Delete as appropriate

If you have answered NO to all questions on this form, then your project does not require ethical approval. You should submit and retain this form as evidence of this.		
4	<p>Does your project involve human participants or their identifiable personal data?</p> <p><i>For example, as interviewees, respondents to a survey or participants in testing.</i></p>	YES

## PART B: Ethics Proportionate Review Form

If you answered YES to question 4 and NO to all other questions in sections A1, A2 and A3 in PART A of this form, then you may use PART B of this form to submit an application for a proportionate ethics review of your project. Your project supervisor has delegated authority to review and approve this application under proportionate review. You must receive final approval from your supervisor in writing before beginning the planned research.

However, if you cannot provide all the required attachments (see B.3) with your project proposal (e.g. because you have not yet written the consent forms, interview schedules etc), the approval from your supervisor will be **provisional**. You **must** submit the missing items to your supervisor for approval prior to commencing these parts of your project. Once again, you must receive written confirmation from your supervisor that any provisional approval has been superseded by with **full approval** of the planned activity as detailed in the full documents. **Failure to follow this procedure and demonstrate that final approval has been achieved may result in you failing the project module.**

Your supervisor may ask you to submit a full ethics application through Research Ethics Online, for instance if they are unable to approve your application, if the level of risks associated with your project change, or if you need an approval letter from the CSREC for an external organisation.

B.1 The following questions must be answered fully.		Delete as appropriate
All grey instructions must be removed.		
1.1.	Will you ensure that participants taking part in your project are fully informed about the purpose of the research?	YES
1.2	Will you ensure that participants taking part in your project are fully informed about the procedures affecting them or affecting any information collected about them, including information about how the data will be used, to whom it will be disclosed, and how long it will be kept?	YES
1.3	When people agree to participate in your project, will it be made clear to them that they may withdraw (i.e. not participate) at any time without any penalty?	YES
1.4	<p>Will consent be obtained from the participants in your project?</p> <p>Consent from participants will be necessary if you plan to involve them in your project or if you plan to use identifiable personal data from existing records. "Identifiable personal data" means data relating to a living person who might be identifiable if the record includes their name, username, student id, DNA, fingerprint, address, etc.</p> <p><i>If YES, you must attach drafts of the participant information sheet(s) and consent form(s) that you will use in section B.3 or, in the case of an existing dataset, provide details of how consent has been obtained.</i></p> <p><i>You must also retain the completed forms for subsequent inspection.</i></p> <p><i>Failure to provide the completed consent request forms will result in withdrawal of any earlier ethical approval of your project.</i></p>	YES



1.5	Have you made arrangements to ensure that material and/or private information obtained from or about the participating individuals will remain confidential?	YES
-----	--	-----

B.2 If the answer to the following question (B2) is YES, you must provide details		Delete as appropriate
2	Will the research be conducted in the participant's home or other non-University location?  <i>If YES, you must provide details of how your safety will be ensured.</i>	NO

B.3 Attachments			
ALL of the following documents MUST be provided to supervisors if applicable.  All must be considered prior to final approval by supervisors.  A written record of final approval must be provided and retained.		YES	NO
Details on how safety will be assured in any non-University location, including risk assessment if required (see B2)			✓
Details of arrangements to ensure that material and/or private information obtained from or about the participating individuals will remain confidential (see B1.5)  <i>Any personal data must be acquired, stored and made accessible in ways that are GDPR compliant.</i>			✓
Full protocol for any workshops or interviews**			✓
Participant information sheet(s)**		✓	
Consent form(s)**			✓
Questionnaire(s)**  <i>sharing a Qualtrics survey with your supervisor is recommended.</i>		✓	
Topic guide(s) for interviews and focus groups**			✓
Permission from external organisations or Head of Department**  <i>e.g. for recruitment of participants</i>			✓

**\*\*If these items are not available at the time of submitting your project proposal, then *provisional approval* can still be given, under the condition that you must submit the final versions of all items to your supervisor for approval at a later date. All such items *must* be seen and approved by your supervisor before the activity for which they are needed begins. Written evidence of *final approval* of your planned activity must be acquired from your supervisor before you commence.**

## Changes

If your plans change and any aspects of your research that are documented in the approval process change as a consequence, then any approval acquired is invalid. If issues addressed in Part A (the checklist) are affected, then you must complete the approval process again and establish the kind of approval that is required. If issues addressed in Part B are affected, then you must forward updated documentation to your supervisor and have received written confirmation of approval of the revised activity before proceeding.

### **Templates for Consent and Information**

You must use the templates provided by the University as the basis for your participant information sheets and consent forms. You **must** adapt them according to the needs of your project before you submit them for consideration.

Participant Information Sheets, Consent Forms and Protocols must be consistent. Please ensure that this is the case prior to seeking approval. Failure to do so will slow down the approval process.

We strongly recommend using Qualtrics to produce digital information sheets and consent forms.

### **Further Information**

<http://www.city.ac.uk/departments-computer-science/research-ethics>

<https://www.city.ac.uk/research/ethics/how-to-apply/participant-recruitment>

<https://www.city.ac.uk/research/ethics>

### **Reference**

<sup>1</sup> (Thom, 2014, <https://strafedevblog.com/post/93056717598/how-we-are-handling-level-generation-in-strafe>).

<sup>2</sup> (Raluca, 2016, <https://marionettestudio.com/agile-game-development-quick-overview/#:~:text=What%20is%20Agile%20Game%20Development,in%20small%20periods%20of%20time.>)

### **Appendix B. Participation information sheet.**

Link:

<https://drive.google.com/file/d/1hlvd9J6SK5xmy4ewVi5lr8G515qzk29i/view?usp=sharing>

### **PARTICIPANT INFORMATION DOCUMENT**

#### **REC reference number, date and version of information sheet**

Reference Number: 0001

Date: 22<sup>nd</sup> April 2021

Version: 1.0

#### **Title of study**

3D FPS Game with procedurally generated content and VR Support.

#### **Name of principal researcher**

Kevin La

Email address: [Kevin.La@city.ac.uk](mailto:Kevin.La@city.ac.uk)

### **Introductory paragraph**

I would like to invite you to participate in an **anonymous** research study as part of my Final Project at City, University of London. Before taking the survey it is important for me and for you to ensure that you know why this survey is being conducted and what is involved. You will be given a copy of this sheet to refer back to.

### **Purpose of the study?**

Procedural Content generation (PCG) is a technique commonly used in games but can be used in other types of projects such as movies. The idea is that the content is generated automatically 1 way or another. The main objective of this study is to see whether PCG technique added to a game adds more replayability value. The sub objective of the study is to see if adding VR support also adds a new way to experience the game which will hopefully increase the value and replayability of the game since it opens a new possibility to play.

The PCG algorithm implemented in the project is mainly used to generate the level structure using custom made room, each playthrough will have a different layout with different enemies, items and more.

### **Why have I been invited to take part?**

You have been selected due to your knowledge in games. This will result in answers that give me a good understanding of whether a game feature works well, what can be improved and what feature will make the game better, as a gamer you will know what to look out for when playing the game.

### **Do I have to take part?**

Participation is optional. You can decide to participate or not participate at all in this study. If at any stage you feel uncomfortable or worried, you may opt out of the project without any consequences. It is entirely up to you whether or not to take part. A reminder that at any time of participating in this project, I or any tools that you have used to support the study's does in no way gather any information about you. All participation stages will be done **anonymously**.

### **What will happen if I take part?**

Should you proceed and take part in the study, you should expect the following actions to happen:

Read the participation sheet to fully understand the process and reasons of participation.

Will be tasked to play test the game. VR mode or Non-VR mode.

Will be asked to fill out an Anonymous survey based on the game you have played, giving feedback on each feature in the game.

### **What do I have to do if I take part?**

When taking part, you will have to play test the game given to you and fill in the survey as you play or after you played.

### **What are the possible disadvantages or risks of taking part?**

Since optimization was not an importance in this project, the game may or may not run on the participant's machine. The project is solely designed for windows operating system so any MAC users or Linux users will need to find other means to get access to a windows machine to run the game.

### **What are the possible benefits of taking part?**

There is no real benefit for taking part however taking part will greatly benefit the project as well as future game developers if they require PCG technique implemented in their game.

### **What should I do if I want to take part?**

If you want to take part in the study, you will be required to head to the link that you will be given by myself. The links provided will direct you to the anonymous survey in which you will be able to answer whether features work or not. You will also be required to download and extract the project that will be given to you by myself in which you will be able to run the game and play test.

### **Data privacy statement**

City, University of London is the sponsor and the data controller of this study based in the United Kingdom. This means that we are responsible for looking after your information and using it properly. The legal basis under which your data will be processed is City's public task.

Your right to access, change or move your information are limited, as we need to manage your information in a specific way in order for the research to be reliable and accurate. To safeguard your rights, we will use the minimum personal-identifiable information possible (for further information please see <https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/lawful-basis-for-processing/public-task/>).

Although the survey will be done Anonymously and no identifiable data about the participant is recorded, the only people who have access to the results is myself and the person marking the project.

You can find out more about how City handles data by visiting <https://www.city.ac.uk/about/governance/legal>. If you are concerned about how we have processed your personal data, you can contact the Information Commissioner's Office (IOC) <https://ico.org.uk/>.

### **What will happen to the results?**

The result of the survey is completely anonymous as the survey links given have been set to Anonymous mode. The results of the survey may be used to support my points in my report. The results will be used for nothing else apart from the project.

### **Who has reviewed the study?**

This study has been approved by City, University of London, Department of Computer Science Research Ethics Committee.

### **What if there is a problem?**

If you have any problems, concerns, or questions about this study, you should ask to speak to a member of the research team. If you remain unhappy and wish to complain formally, you can do this through City's complaints procedure. To complain about the study, you need to phone 020 7040 3040. You can then ask to speak to the Secretary to Senate Research Ethics Committee and inform

them that the name of the project is 3D FPS Game with procedurally generated content and VR Support.

You can also write to the Secretary at:

Anna Ramberg  
Research Integrity Manager

City, University of London, Northampton Square  
London, EC1V 0HB

Email: [Anna.Ramberg.1@city.ac.uk](mailto:Anna.Ramberg.1@city.ac.uk)

**Thank you for taking the time to read this information sheet.**

## **Appendix C Non-VR Qualtrics Questionnaire**


### Anonymous Survey Link

A reusable link that can be pasted into emails or onto a website,  
and is unable to track identifying information of respondents.

[https://cityunilondon.eu.qualtrics.com/jfe/form/SV\\_dilDfiw96k1NqU6](https://cityunilondon.eu.qualtrics.com/jfe/form/SV_dilDfiw96k1NqU6)

Customize Link

[https://cityunilondon.eu.qualtrics.com/jfe/form/SV\\_dilDfiw96k1NqU6](https://cityunilondon.eu.qualtrics.com/jfe/form/SV_dilDfiw96k1NqU6)




\*This questionnaire does not collect any personal information  
and will be completed in Anonymous mode\*

Before starting, have you read the participation information  
sheet?

☐ I have read the participation information sheet.


☐ I have not read the participation information sheet.





Captcha

☐ I'm not a robot

  
reCAPTCHA  
[Privacy](#) - [Terms](#)



Can you move the character around using W,A,S,D keys and look around using your mouse?

- ☐ Yes, can do both
- ☐ Can only move around
- ☐ Can only look around
- ☐ None of the above

Does the camera show a first person perspective?

- ☐ Yes
- ☐ No

Can you jump using the space bar?

- ☐ Yes
- ☐ No

Can you pick up weapons? (Using E to equip on weapon slot 1 and Q to equip on secondary weapon and M to equip on melee slot 1)

☐ Yes

☐ No

If you selected Yes in the previous question, can you switch weapons using the mouse scroll wheel?

☐ Yes

☐ No

Can you drop the weapon currently in hand? (Using the G Key)

☐ Yes

☐ No

Can you shoot the gun or attack with the sword?

☐ Yes

☐ No

After the gun has no bullets remaining in the clip, can you reload the gun and shoot after reloading has finished?

☐ Yes

☐ No

Do the enemies you encounter patrol the rooms, chase you, shoot/attack? (Depending on the type of enemy)

☐ All of the above

☐ Only Patrols

☐ Only Chases

☐ Only Shoots/Attack

☐ None of the above

Are you able to defeat the enemies with your weapons?

☐ Yes

☐ No

Do you, the player receive damage from enemy attacks? (Close attack, Enemy laser beam and more)

☐ Yes

☐ No

With each playthrough of the game, does the level layout/structure change? (Meaning no playthrough should be the same)

☐ Yes

☐ No

Do you see any clipping or rooms overlapping each other?

☐ Yes, A lot of overlapping

☐ Rarely

☐ Not at all

With each room, do you see objects (Power ups, weapons, enemies) spawned in specific locations? (Apart from the boss circular room)

☐ Yes

☐ No

With each playthrough, do you notice that the objects spawned at specific locations are randomized? (May be hard to tell due to limited amount of available objects in the game)

☐ Yes

☐ No

With the spawned loot in each room, can you collect them (Power ups) or equip them (Weapons)?

☐ Yes

☐ No



Do you also see Power ups spawning and falling down to the ground upon defeating an enemy?

☐ Yes

☐ No

Do the health orbs (Red orbs) slowly gravitate towards you as you are close to it?

☐ Yes

☐ No

Which of the power ups (Green speed orbs, Red health orbs, ammo pack) update the player as expected?

☐ All of them

☐ Speed orbs

☐ Health orbs

☐ Ammo pack

☐ None

Was you able to navigate through the main menu to start the game in the desired mode and difficulty?

☐ Yes

☐ No

Does the HUD in-game work as expected? (Does the health deplete when getting hit or ammo count decrease upon shooting)

☐ HUD works fine

☐ Only the health bar works

☐ Only the ammo count works

☐ None of them work

Are you able to pause in-game using the ESC key?

☐ Yes

☐ No





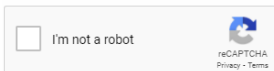
\*This questionnaire does not collect any personal information and will be completed in Anonymous mode\*

Before starting, have you read the participation information sheet?

- ☐ I have read the participation information sheet.
- ☐ I have not read the participation information sheet.



Captcha



Can you move the character around using the touchpad on the Vive wand and does the camera accurately follow your head movements?

☐ Both work fine

☐ Can only move around

☐ Camera follows my head movement

☐ None of the above

With the right controller touch pad, are you able to rotate the camera each time to press Left or Right on the touch pad?

☐ Yes

☐ No

Does a tunnel vision effect appear when moving your character in any direction?

☐ Yes

☐ No

Can you make the character jump by pressing down the left touchpad on the Vive wand?

☐ Yes

☐ No

Can you pick up weapons? (Using the grip button on the controllers when near the weapon)

☐ Yes

☐ No

For items far away, can you click & hold the touch pad on the right controller (Which should display a line from your hand) and point to a weapon and grab it with the grip button?

☐ Yes

☐ No

Are you able to shoot with the guns using the Right trigger?

☐ Yes

☐ No

Are you able to holster the gun by holding the gun and putting it on your sides?

☐ Yes

☐ No

Can you grab the guns from the holster and use it as normal?

☐ Yes

☐ No

Are you able to put the sword on your back by letting go of the grip button after positioning your hand (The hand holding the sword) behind your neck?

☐ Yes

☐ No

After the gun has no bullets remaining, can you reload the gun by holding the gun around chest level and shoot after reload has finished?

☐ Yes

☐ No

When using the sword, can you swing the weapon and does it inflict damage on enemies? (Basically can you defeat an enemy with your sword?)

☐ Yes

☐ No

Can you defeat enemies with your gun?

☐ Yes

☐ No

Do the enemies you encounter patrol the rooms, chase you, shoot/attack? (Depending on the type of enemy)

☐ All of the above

☐ Only Patrols

☐ Only Chases

☐ Only Shoots/Attack

☐ None of the above

Does the character you control receive damage from enemy attacks or enemy projectiles?

☐ Yes

☐ No

With each playthrough of the game, does the level layout/structure change?

☐ Yes

☐ No

Do you see any clipping or rooms overlapping each other?

☐ Yes, A lot of overlapping

☐ Rarely

☐ Not at all

With each room, do you see items (Power ups, weapons) spawned in specific locations?

☐ Yes

☐ No

With each playthrough, do you notice that the items spawned at specific locations are randomized?

☐ Yes

☐ No

With the spawned loot in each room, can you collect them (Power ups) or equip them (Weapons)?

☐ Yes

☐ No

Do you also see Power ups spawning and falling down to the ground upon defeating an enemy?

☐ Yes

☐ No

Do the health orbs (Red orbs) slowly gravitate towards you as you are close to it?

☐ Yes

☐ No

Which of the power ups (Green speed orbs, Red health orbs, ammo pack) update the player as expected?

☐ All of them

☐ Speed orbs

☐ Health orbs

☐ Ammo pack

☐ None

Are you able to toggle the tunnel vision on/off in the pause menu?

☐ Yes

☐ No

☐ I couldn't tell if it was toggled on or off.

Any other feedback to give?



End of survey.

Thank you for participating. Just a reminder that you have completed this survey using the Anonymous link meaning that we will be unable to track identifying information of respondents.

## Appendix E. Non VR Mode survey results

Recorded Date	Q1 - *This questionnaire does not collect any personal information and will be c...	Q3 - Can you move the character around using W,A,S,D keys and look around using...	Q26 - Does the camera show a first person perspective?	Q4 - Can you jump using the space bar?	Q5 - Can you pick up weapons? (Using E to equip on weapon slot 1 and Q to equip...	Q6 - If you selected Yes in the previous question, can you switch weapons using...	Q7 - Can you drop the weapon currently in hand? (Using the G Key)	Q8 - Can you shoot the gun or attack with the sword?
Apr 26, 2021 2:12 PM	I have read the participation information sheet.	Yes, can do both	Yes	Yes	Yes	Yes	Yes	Yes
Apr 27, 2021 8:09 PM	I have read the participation information sheet.	Yes, can do both	Yes	Yes	Yes	Yes	Yes	Yes
Apr 26, 2021 2:03 PM	I have read the participation information sheet.	Yes, can do both	Yes	Yes	Yes	Yes	Yes	Yes
Apr 26, 2021 2:01 PM	I have read the participation information sheet.	Yes, can do both	Yes	Yes	Yes	Yes	Yes	Yes
Apr 26, 2021 2:02 PM	I have read the participation information sheet.	Yes, can do both	Yes	Yes	Yes	Yes	Yes	Yes

Recorded Date	Q9 - After the gun has no bullets remaining in the clip, can you reload the gun...	Q10 - Do the enemies you encounter patrol the rooms, chase you, shoot/attack? (De...	Q11 - Are you able to defeat the enemies with your weapons?	Q12 - Do you, the player receive damage from enemy attacks? (Close attack, Enemy...	Q13 - With each playthrough of the game, does the level layout/structure change?...	Q14 - Do you see any clipping or rooms overlapping each other?	Q15 - With each room, do you see objects (Power ups, weapons, enemies) spawned in...
Apr 26, 2021 2:12 PM	Yes	All of the above	Yes	Yes	Yes	Rarely	Yes
Apr 27, 2021 8:09 PM	Yes	All of the above	Yes	Yes	Yes	Not at all	Yes
Apr 26, 2021 2:03 PM	No	All of the above	Yes	Yes	Yes	Not at all	Yes
Apr 26, 2021 2:01 PM	Yes	All of the above	Yes	Yes	Yes	Not at all	Yes
Apr 26, 2021 2:02 PM	Yes	All of the above	Yes	Yes	Yes	Rarely	Yes



Recorded Date	Q16 - With each playthrough, do you notice that the objects spawned at specific l...	Q17 - With the spawned loot in each room, can you collect them (Power ups) or equi...	Q18 - Do you also see Power ups spawning and falling down to the ground upon defe...	Q19 - Do the health orbs (Red orbs) slowly gravitate towards you as you are close...	Q20 - Which of the power ups (Green speed orbs, Red health orbs, ammo pack) updat...	Q21 - Was you able to navigate through the main menu to start the game in the des...	Q22 - Does the HUD in-game work as expected? (Does the health deplete when gettin...
Apr 26, 2021 2:12 PM	No	Yes	Yes	Yes	All of them	No	HUD works fine
Apr 27, 2021 8:09 PM	Yes	Yes	Yes	Yes	All of them	Yes	HUD works fine
Apr 26, 2021 2:03 PM	Yes	Yes	Yes	Yes	All of them	Yes	HUD works fine
Apr 26, 2021 2:01 PM	Yes	Yes	Yes	Yes	All of them	Yes	HUD works fine
Apr 26, 2021 2:02 PM	Yes	Yes	Yes	Yes	Speed orbs Health orbs	Yes	HUD works fine

Recorded Date	Q23 - Are you able to pause in-game using the ESC key?	Q24 - If you selected yes in the previous question, does pressing the buttons in...	Q25 - Any other feedback to give? (E.g. about what you would like to see in the f...
Apr 26, 2021 2:12 PM	Yes	Yes	Feel like the button to equip weapons should be compiled into 1. Also, the main menu was not navigable after dying. As for the room generator, it did randomly generate but with just a limited amount of rooms in the game, it didn't feel very different each playthrough.
Apr 27, 2021 8:09 PM	Yes	Yes	Main menu after dying mouse disappears, settings not implemented
Apr 26, 2021 2:03 PM	Yes	Yes	Some rooms didn't have walls. The energy gun sometimes didn't get ammo
Apr 26, 2021 2:01 PM	Yes	Yes	
Apr 26, 2021 2:02 PM	Yes	Yes	

## Appendix F. VR Mode survey results

Recorded Date	Q1 - *This questionnaire does not collect any personal information and will be c...	Q3 - Can you move the character around using the touchpad on the Vive wand and d...	Q4 - With the right controller touch pad, are you able to rotate the camera each...	Q5 - Does a tunnel vision effect appear when moving your character in any direct...
May 2, 2021 8:51 PM	I have read the participation information sheet.	Both work fine	Yes	Yes
May 2, 2021 8:47 PM	I have read the participation information sheet.	Both work fine	Yes	Yes
May 2, 2021 8:43 PM	I have read the participation information sheet.	Both work fine	Yes	Yes

Recorded Date	Q6 - Can you make the character jump by pressing down the left touchpad on the V...	Q7 - Can you pick up weapons? (Using the grip button on the controllers when nea...	Q8 - For items far away, can you click & hold the touch pad on the right control...	Q9 - Are you able to shoot with the guns using the Right trigger?
May 2, 2021 8:51 PM	Yes	Yes	Yes	Yes
May 2, 2021 8:47 PM	Yes	Yes	Yes	Yes
May 2, 2021 8:43 PM	Yes	Yes	Yes	Yes

Recorded Date	Q10 - Are you able to holster the gun by holding the gun and putting it on your s...	Q11 - Can you grab the guns from the holster and use it as normal?	Q12 - Are you able to put the sword on your back by letting go of the grip button...	Q13 - After the gun has no bullets remaining, can you reload the gun by holding t...	Q14 - When using the sword, can you swing the weapon and does it inflict damage o...
May 2, 2021 8:51 PM	Yes	Yes	Yes	Yes	Yes
May 2, 2021 8:47 PM	Yes	Yes	Yes	Yes	Yes
May 2, 2021 8:43 PM	Yes	Yes	Yes	Yes	Yes

Recorded Date	Q15 - Can you defeat enemies with your gun?	Q16 - Do the enemies you encounter patrol the rooms, chase you, shoot/attack? (De...	Q17 - Does the character you control receive damage from enemy attacks or enemy p...	Q18 - With each playthrough of the game, does the level layout/structure change?	Q19 - Do you see any clipping or rooms overlapping each other?	Q20 - With each room, do you see items (Power ups, weapons) spawned in specific l...
May 2, 2021 8:51 PM	Yes	All of the above	Yes	Yes	Not at all	Yes
May 2, 2021 8:47 PM	Yes	All of the above	Yes	Yes	Rarely	Yes
May 2, 2021 8:43 PM	Yes	All of the above	Yes	Yes	Not at all	Yes

Recorded Date	Q21 - With each playthrough, do you notice that the items spawned at specific loc...	Q23 - Do you also see Power ups spawning and falling down to the ground upon defe...	Q22 - With the spawned loot in each room, can you collect them (Power ups) or equi...	Q24 - Do the health orbs (Red orbs) slowly gravitate towards you as you are close...	Q25 - Which of the power ups (Green speed orbs, Red health orbs, ammo pack) updat...
May 2, 2021 8:51 PM	Yes	Yes	Yes	Yes	All of them
May 2, 2021 8:47 PM	Yes	Yes	Yes	Yes	All of them
May 2, 2021 8:43 PM	Yes	Yes	Yes	Yes	All of them

Recorded Date	Q26 - Does the HUD in-game work as expected? (Does the health deplete when gettin...	Q27 - Are you able to pause in-game using the menu button? (Circle button above L...	Q28 - If you selected yes in the previous question, does pressing the trigger on...	Q29 - Are you able to toggle the tunnel vision on/off in the pause menu?	Q30 - Any other feedback to give?
May 2, 2021 8:51 PM	HUD works fine	Yes	Yes	Yes	When gun was attached to the holster, it was hard to pickup new guns from the floor. It kept thinking I wanted to equip the gun from the holster. Also the size of objects in the game was very strange. Guns were small when on the floor.
May 2, 2021 8:47 PM	HUD works fine	Yes	Yes	Yes	I saw 1 instance of overlapping. The end room corner was visible within another room. There is no indication that the sword has hit, I kept swinging until the enemy exploded.
May 2, 2021 8:43 PM	HUD works fine	Yes	Yes	Yes	Having the game start with a main menu that is VR friendly would be nice. Sword combat did not work well with small spider. Was only able to defeat drone with the sword. But overall, the controls worked and with a little more tuning it may be a standalone game for VR that can be further developed.

## Appendix G. User and Installation Guide

Link: <https://drive.google.com/file/d/1wnEZWq5CizDnHmQ7BNZ6UrSMaxD-IOYo/view?usp=sharing>

### Installation Guide

Upon downloading the project folder using the google drive link ([https://drive.google.com/file/d/1S2LKiV4Mwt7T0Ug\\_1-PzGzGWv7gTDce0/view?usp=sharing](https://drive.google.com/file/d/1S2LKiV4Mwt7T0Ug_1-PzGzGWv7gTDce0/view?usp=sharing)), find the executable called "FPS Final Project.exe" and you will be greeted with the main menu.

For VR Mode, Ensure that you have Steam VR installed and all the HTC Vive trackers, headset and controllers are turned on and working. Upon running the executable the project will launch SteamVR automatically which will allow you to select VR Mode.

### Gameplay guide

#### Main Menu

When you run the executable, you will be put in the main menu in which you can select the mode you wish to play. If you have selected Normal mode you will be given the option to select difficulty.

As of now, the Settings menu does not work and is only used as a place holder for when settings is implemented.

#### Player controls NON-VR

Keys/Controls	Action
W,A,S,D (Alt. Arrow keys)	Move the character in the direction pressed.
Mouse movement	Look around
Spacebar	Jump
E	Equip gun to slot 1
Q	Equip gun to slot 2
M	Equip sword to melee slot 1
Left mice click	Shoot gun / Attack with sword
R	Reload the gun
ESC	Pause game

#### Player controls VR

Keys/Controls	Action
Left controller touchpad	Move the character in the direction pressed.
Right controller touchpad left click/right click	Rotates the camera 45 degrees left or right
Left controller touchpad click	Jump
Grip button on any controller	Hold weapon
Right controller touchpad click and hold	Long range grab

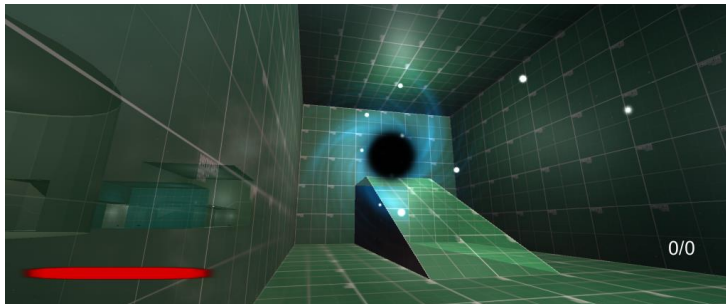
Right controller trigger (when weapon is held on the right controller)	Shoot gun / Attack with sword
Hold gun to chest	Reload the gun
Hold gun to sides	Holster the gun
Hold sword behind back	Holster the sword
Right Primary button (Button above touchpad)	Pause
Right controller trigger (WHILE PAUSED)	Clicks on pause buttons

### Power ups

While you travel through the map you will encounter items such as power ups that you can collect to give yourself a boost. These powerups include Speed orbs (Green orbs), Health orbs (Red orbs) and ammo pack. To collect them, simply touch them. Powerups may spawn within rooms and will be guaranteed to spawn upon defeating an enemy.

### End Goal

Reach the end portal located in a green room.



### **Appendix H. Unity Project and building instructions**

Link: [https://drive.google.com/file/d/1-D1ynofCldEtLKCsI4K\\_h47bnDWu1UMI/view?usp=sharing](https://drive.google.com/file/d/1-D1ynofCldEtLKCsI4K_h47bnDWu1UMI/view?usp=sharing)

The Windows executable and source code were both submitted on the cloud ( Google Drive ) and can be accessed via the link:

Windows Executable: [https://drive.google.com/file/d/1S2LKiv4Mwt7T0Ug\\_1-PzGzGWv7gTDce0/view?usp=sharing](https://drive.google.com/file/d/1S2LKiv4Mwt7T0Ug_1-PzGzGWv7gTDce0/view?usp=sharing)

Unity Project: [https://drive.google.com/file/d/11WCOBreXwiVqw\\_qKvxcTSDft-Jp-0zTP/view?usp=sharing](https://drive.google.com/file/d/11WCOBreXwiVqw_qKvxcTSDft-Jp-0zTP/view?usp=sharing)

The link is also found in the text file submitted in Other Appendices

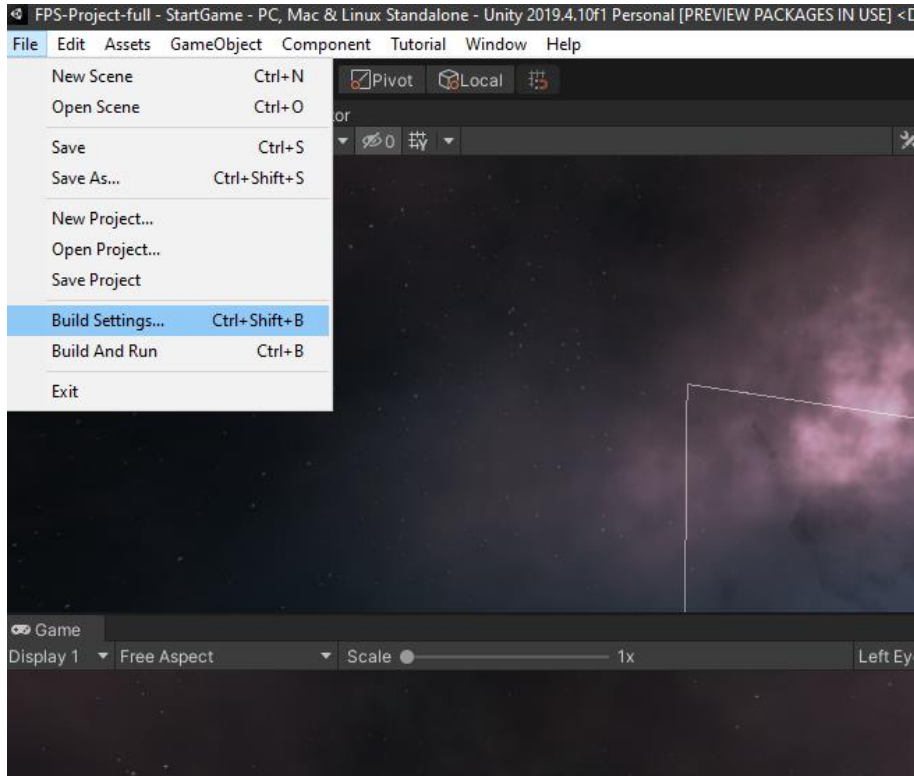
The Project folder contains the Unity project with all the source code and assets for the Unity 2019.4.10f1 editor version. If you wish to run the project through Unity Editor, you will need to perform the following steps:

- Download and open Unity Hub and log into your Unity account.
- Download and install the version 2019.4.10f1 – you can get the build from <https://unity3d.com/get-unity/download/archive>

- From Unity Hub, click on “Add” and select the Final project folder that was extracted from the zip file..

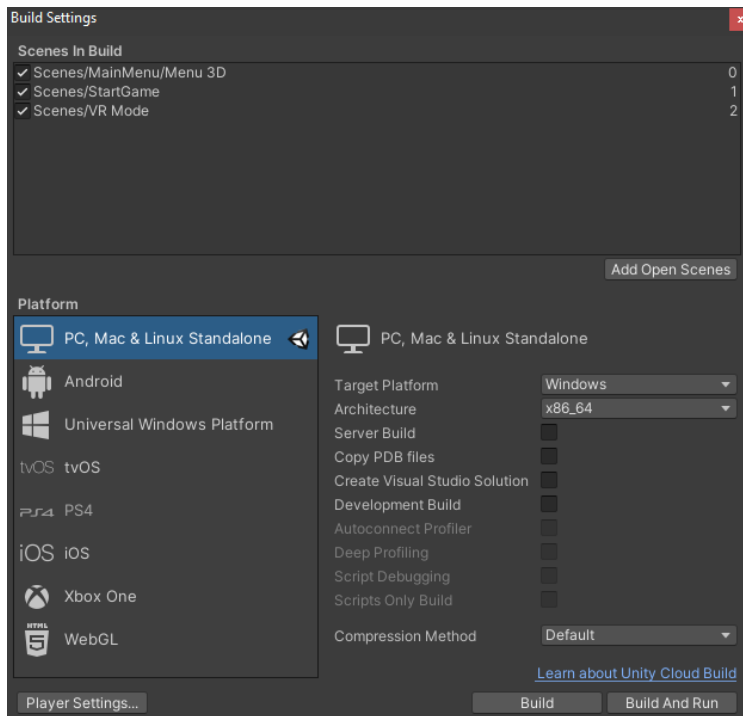
### Building the project

To re-build the project, open it in Unity and navigate to Build Settings by selecting File top left.



Alternatively you can press Control+Shift+B.

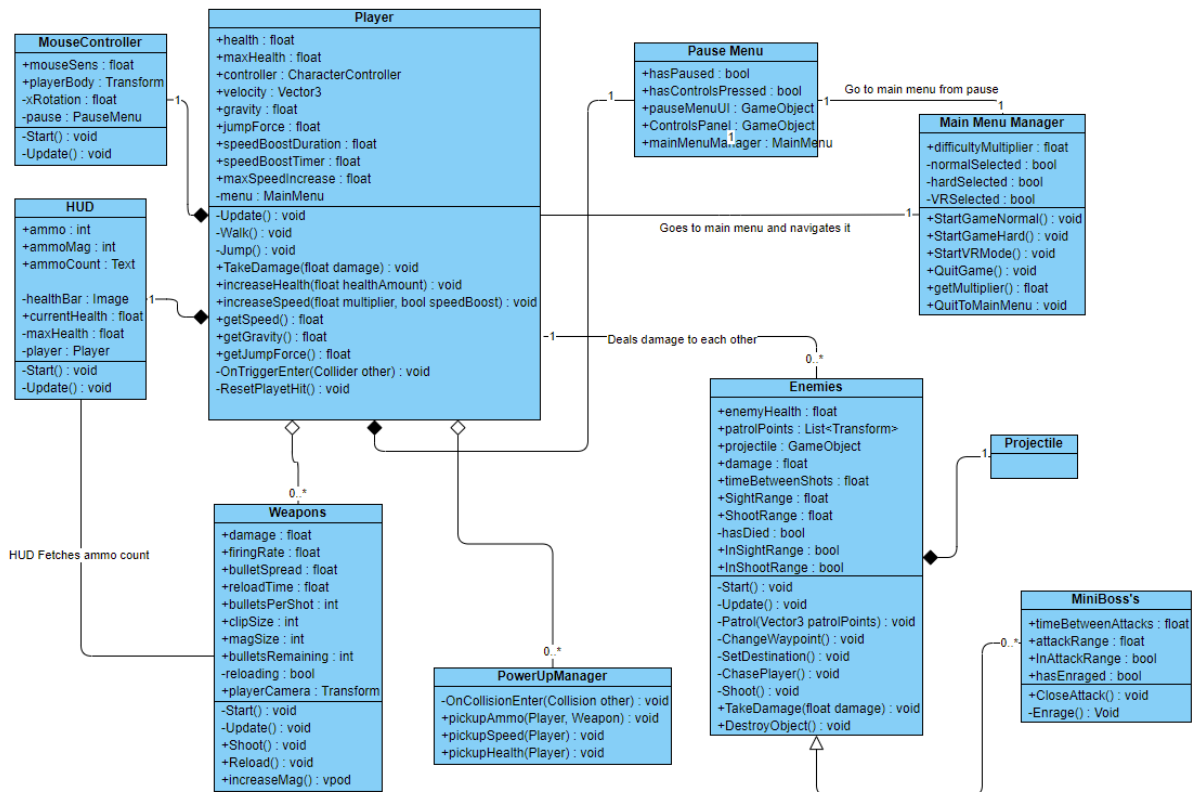
Make sure all MainMenu, StartGame and VR Mode scenes are added to the Scenes in Build tab in the order seen in the image.



Select Target platform and Architecture and select Build And Run. It will give you a prompt to select a folder where you want to store the Windows executable.

Please note that the project is not intended for any platform other than PC and has not been optimized or test in any other platform.

### **Appendix I. UML Class Diagram (Created in Visual paradigm online)**



## REFERENCES

- [1] Baver, K. and Baver, K. (2021) *This is the Way: How Innovative Technology Immersed Us in the World of The Mandalorian* | *StarWars.com, StarWars.com*. Available at: <https://www.starwars.com/news/the-mandalorian-stagecraft-feature> (Accessed: 3 May 2021).
- [2] Packt>, P. (2021) *The Drawbacks of procedural generation*, Packt>. Available at: [https://subscription.packtpub.com/book/game\\_development/9781785886713/1/ch01lvl1sec14/the-drawbacks-of-procedural-generation](https://subscription.packtpub.com/book/game_development/9781785886713/1/ch01lvl1sec14/the-drawbacks-of-procedural-generation) (Accessed: 3 May 2021).
- [3] Fingas, J. (2015) *How minecraft worlds are made*, Engadget.com. Available at: <https://www.engadget.com/2015-03-04-how-minecraft-worlds-are-made.html> (Accessed: 3 May 2021).
- [4] 9fingergames, 9. (2016) *Guns. guns? GUNS! news - GoToHell, Indie DB*. Available at: <https://www.indiedb.com/games/gotohell/news/guns-guns-guns5> (Accessed: 3 May 2021).
- [5] N/A, N. (2017) *Dungeon Generation - Procedural Content Generation Wiki*, Pcg.wikidot.com. Available at: <http://pcg.wikidot.com/pcg-algorithm:dungeon-generation> (Accessed: 3 May 2021).
- [6] Johnson, L., N.Yannakakis, G. and Togelius, J. (2010) *Cellular automata for real-time generation of infinite cave levels* | *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, *DL.acm.org*. Available at: <https://dl.acm.org/doi/abs/10.1145/1814256.1814266> (Accessed: 3 May 2021).
- [7] Hely, T. (2013) *How to Use BSP Trees to Generate Game Maps*, Game Development Envato Tuts+. Available at: <https://gamedevelopment.tutsplus.com/tutorials/how-to-use-bsp-trees-to-generate->

game-maps--gamedev-12268 (Accessed: 3 May 2021). Brackey, B. (2017) *Ammo & Reloading - Unity Tutorial*. Available at: <https://www.youtube.com/watch?v=kAx5g9V5bcM&t=39s> (Accessed: 3 May 2021).

[8] Fuad, M. and Mahendran, A. (2020) *Unity XR platform updates* *Unity XR 플랫폼 업데이트* *Unity XR プラットフォームの最新情報* - *Unity Technologies Blog*, *Unity Technologies Blog*. Available at: <https://blogs.unity3d.com/2020/01/24/unity-xr-platform-updates/> (Accessed: 3 May 2021).

[9] Sunny Valley Studio, S. (2019) *Procedural dungeon in Unity 3D Tutorial P1 - Theory*. Available at: <https://www.youtube.com/watch?v=VnqN0v95jtU&list=PLcRSafycjWFFEPbSSjGMNY-goOZTuBPMW> (Accessed: 3 May 2021).

[10] Lague, S. (2015) *[Unity] Procedural Cave Generation (E01. Cellular Automata)*. Available at: <https://www.youtube.com/watch?v=v7yyZZjF1z4> (Accessed: 3 May 2021).

[11] N/A, N. (2014) *How we are handling level generation in STRAFE*, *strafegame*. Available at: <https://strafedevblog.com/post/93056717598/how-we-are-handling-level-generation-in-strafe> (Accessed: 3 May 2021).

[12] Project-Shasta, P. (2018) *[Unity] Random Level Generator #00 - Project overview*. Available at: [https://www.youtube.com/watch?v=C4ZqrhCP0Bg&list=PLvMpomwW7ZQH3jHDyFP\\_hUS8560E4SdKM](https://www.youtube.com/watch?v=C4ZqrhCP0Bg&list=PLvMpomwW7ZQH3jHDyFP_hUS8560E4SdKM) (Accessed: 3 May 2021).

[13] Brackeys, B. (2019) *FIRST PERSON MOVEMENT in Unity - FPS Controller*. Available at: [https://www.youtube.com/watch?v=\\_QajrabyTjc](https://www.youtube.com/watch?v=_QajrabyTjc) (Accessed: 3 May 2021).

[14] Dave/GameDevelopment, D. (2020) *FULL PICK UP & DROP SYSTEM for WEAPONS or ITEMS || Unity3d Tutorial*. Available at: <https://youtu.be/8kKLUsn7tcg> (Accessed: 3 May 2021).

[15] Brackeys, B. (2017) *Weapon Switching - Unity Tutorial*. Available at: [https://www.youtube.com/watch?v=Dn\\_BUIVdAPg&t=54s](https://www.youtube.com/watch?v=Dn_BUIVdAPg&t=54s) (Accessed: 3 May 2021).

[16] Dave/GameDevelopment, D. (2020) *How to make ALL kinds of GUNS with just ONE script! (Unity3d tutorial)*. Available at: Dave/GameDevelopment, D. (2020) *FULL PICK UP & DROP SYSTEM for WEAPONS or ITEMS || Unity3d Tutorial*. Available at: <https://youtu.be/8kKLUsn7tcg> (Accessed: 3 May 2021). (Accessed: 3 May 2021).

[17] Dave/GameDevelopment, D. (2020) *FULL 3D ENEMY AI in 6 MINUTES! || Unity Tutorial*. Available at: <https://www.youtube.com/watch?v=UjKSFOlXesw> (Accessed: 3 May 2021).

[18] MakingMagic, M. (2016) *Unity 3d Magnetic field tutorial*. Available at: <https://www.youtube.com/watch?v=Xhp7nciaACU> (Accessed: 3 May 2021).

[19] OneWheelStudio, O. (2019) *How to Create A Stylized Loot Effect in Unity?*. Available at: <https://www.youtube.com/watch?v=lqGIBMe-Pqw> (Accessed: 3 May 2021).

[20] Wayra Codes, W. (2020) *HEALTH BAR CREATION in UNITY [Easy]*. Available at: <https://www.youtube.com/watch?v=NE5cAlCRgzo> (Accessed: 3 May 2021).

[21] Valem, V. (2020) *Introduction to VR in Unity - PART 2 : INPUT and HAND PRESENCE*. Available at: [https://www.youtube.com/watch?v=VdT0zMcgTQ&list=RDCMUCPJlesN59MzHPPCp0Lg8sLw&start\\_radio=1&t=729](https://www.youtube.com/watch?v=VdT0zMcgTQ&list=RDCMUCPJlesN59MzHPPCp0Lg8sLw&start_radio=1&t=729) (Accessed: 3 May 2021).

[22] Technologies, U. (2021) *Unity - Scripting API: CharacterController.Move*, Docs.unity3d.com. Available at: <https://docs.unity3d.com/ScriptReference/CharacterController.Move.html> (Accessed: 3 May 2021).

[23] Table Flip Games, T. (2019) *Building Simple AI Patrols | Unity AI Pathfinding (Part 3) | Table Flip Games*. Available at: [https://www.youtube.com/watch?v=5q4JHuJAACQ&list=PL8lV\\_joQZ5sfqiNwoJcokJlcrqplW8uSs&index=4](https://www.youtube.com/watch?v=5q4JHuJAACQ&list=PL8lV_joQZ5sfqiNwoJcokJlcrqplW8uSs&index=4) (Accessed: 3 May 2021).

[24] Valem, V. (2020) *Introduction to VR in Unity - PART 5 : GRAB INTERACTION*. Available at: <https://www.youtube.com/watch?v=FMu7hKUX3Oo&t=815s> (Accessed: 3 May 2021).

[25] Mr.Pineapple Studio, M. (2020) *How To: Gun Holster in VR. Unity VR Beginner tutorial, Oculus and any other headset!*. Available at: <https://www.youtube.com/watch?v=FFM2oyLUysk&t=5s> (Accessed: 3 May 2021).

[26] Valem, V. (2020) *Introduction to VR in Unity - PART 3 : TELEPORTATION*. Available at: <https://www.youtube.com/watch?v=fZXKGJYri1Y&t=394s> (Accessed: 3 May 2021).

---

#### UNITY ASSETS USED

Sigtrap (2019) *VR Tunnelling Pro* Available at: <https://assetstore.unity.com/packages/tools/camera/vr-tunnelling-pro-106782> (Accessed: 3 May 2021)

Nolet, C. (2018) *Quick Outline*, Unity Asset Store. Available at: <https://assetstore.unity.com/packages/tools/particles-effects/quick-outline-115488> (Accessed: 3 May 2021).

BYTES, P. (2017) *Starfield Skybox*, Unity Asset Store. Available at: <https://assetstore.unity.com/packages/2d/textures-materials/sky/starfield-skybox-92717> (Accessed: 3 May 2021).

Ciathyza, C. (2018) *Gridbox Prototype Materials*, Unity Asset Store. Available at: <https://assetstore.unity.com/packages/2d/textures-materials/gridbox-prototype-materials-129127> (Accessed: 3 May 2021).

N/A, M. (2020) *Medium Mech Striker*, Unity Asset Store. Available at: <https://assetstore.unity.com/packages/3d/characters/robots/medium-mech-striker-124342> (Accessed: 3 May 2021).

Dmytro, D. (2020) *Unity Asset Store*. Available at: <https://assetstore.unity.com/packages/3d/characters/robots/spider-orange-181154> (Accessed: 3 May 2021).

Technologies, U. (2021) *Unity Samples: UI*, Unity Asset Store. Available at: Dmytro, D. (2020) *Unity Asset Store*. Available at: <https://assetstore.unity.com/packages/3d/characters/robots/spider-orange-181154> (Accessed: 3 May 2021). (Accessed: 3 May 2021).

Studio, H. (2020) *Glowing orbs pack*, Unity Asset Store. Available at: <https://assetstore.unity.com/packages/vfx/particles/spells/glowing-orbs-pack-87648> (Accessed: 3 May 2021). \*\* Purchased item \*\*



GameDev, A. (2019) *LowPoly Sci-Fi Crates Free*, *Unity Asset Store*. Available at: <https://assetstore.unity.com/packages/3d/props/lowpoly-sci-fi-crates-free-146016> (Accessed: 3 May 2021).

Unterguggenberger, J. (2021) *Volumetric Lines*, *Unity Asset Store*. Available at: <https://assetstore.unity.com/packages/tools/particles-effects/volumetric-lines-29160> (Accessed: 3 May 2021).

Asylum, P. (2014) *SciFi Enemies and Vehicles*, *Unity Asset Store*. Available at: <https://assetstore.unity.com/packages/3d/characters/robots/scifi-enemies-and-vehicles-15159> (Accessed: 3 May 2021).

Bykov, A. (2019) *Several Swords With Scabbards*, *Unity Asset Store*. Available at: <https://assetstore.unity.com/packages/3d/props/weapons/several-swords-with-scabbards-121182> (Accessed: 3 May 2021).

Adequate, A. (2015) *Shotgun*, *Unity Asset Store*. Available at: <https://assetstore.unity.com/packages/3d/props/guns/shotgun-26685> (Accessed: 3 May 2021).

Technologies, U. (2020) *Unity Particle Pack 5.x*, *Unity Asset Store*. Available at: <https://assetstore.unity.com/packages/essentials/asset-packs/unity-particle-pack-5-x-73777> (Accessed: 3 May 2021).

*Sci Fi Weapons | Dev Assets* (2021) *Devassets.com*. Available at: <https://devassets.com/assets/sci-fi-weapons/> (Accessed: 3 May 2021).