Project Terms of Reference

Midnight Rising

Contents

1. The Vision of the Project

To develop a top-down wave-based zombie survival game using Unity Engine.

1.1. The Scope of the Project

This project will be addressing the consolidation of our learning so far on our degree into a single game. It will test our knowledge in games programming, games design, project management, and software development practices as well as develop our skills in Unity and C#.

1.2. A Statement of Overall Purpose of the System

Each of the five members in the group project will be responsible for implementing their own individual sub-system. Once completed all these subsystems will be combined into the final game. Collectively the group will be developing a top-down shooter game in which the player has to survive waves of enemy zombies. Many aspects of game development such as gameplay, progression, combat, UI, and AI will be included in the project. A fully functioning, playable, and bug-free game will be produced by the end of the project. The final product will also be demonstrated after production is finished.

1.3. Main Functional Areas of the System

1.3.1. Gameplay Programming - Andrew Alford

Gameplay programming is the job of encapsulating all game logic code required to make the game playable. This includes elements such as interaction/control with a character; movement around the game world; objectives of the game; win/lose conditions of the game; and any other requirements necessary to make the game functional. For this project, the gameplay programmer will be responsible for implementing all these features as well as some game specific functionality for the top-down survival game genre. This includes spawning waves of dynamically spawning enemies which increase in difficulty over time as well as tracking the players in-game progression to see if they can unlock new areas of the map. Responsibilities of the gameplay programmer will be expanded upon in section 2.9.1. Aside from their own tasks, it is also the gameplay programmers' job to work closely with all other team members to ensure their components are correctly incorporated into the game.

1.3.2. HUD & UI - Alexandru Daniel Pascal

HUD and UI design and implementation are jobs regarding the process of creating functionalities for different sorts of 2D art that will aid users to visualise vital information about the current state of game and much more. Those functionalities are related to the player's stats (like health, stamina, current character in control, weapons in use, possibly enemy's health etc.) and main menu control (which would include the option to save/load the game as well as options for different sorts of resolution, skill tree access and quit game). Moreover, when accessing the main menu, the game will pause the current progress. It is important to notice that the work which involves HUD and UI is closely related to other team member's jobs like characters and AI development or Skill Tree functionalities which makes the overall development of the project to be strongly dependent on each other's work.

1.3.3. Characters & AI - Carl Pendleton

Artificial intelligence will be used in the game to define how the computer-controlled characters behave. In this game, at least two-character types will be present – zombie enemies and friendly player characters. The character types will have different behaviours but will both implement common artificial intelligence techniques such as decision trees (sometimes called behaviour trees) and pathfinding which is usually derived from the A* algorithm. Giving characters these pre-defined behaviours gives them the appearance that they are thinking for themselves. Hence, helping to immerse the player in the game. When designing the characters in the game and deciding what pre-defined behaviours they will have, existing video games featuring similar AI will be researched. The research will be used to identify what AI behaviour is expected in today's video games and the behaviours that players expect to see. For example, in zombie games, players will expect the zombies to attack on site of the player. In the Unity game engine, AI functionality is already provided but can be extended. If the functionality is to be extended, research into techniques such as A* pathfinding for example will have to be conducted.

1.3.4. Weapons & Pick-ups - Alex Trench

Weapons and pick-ups are both a gameplay system in which the use can use weapons; usually guns, to damage enemy player or AI. This includes implementing the designs of weapons as well as the functionality. There will be an array of weapons all with different damage and fire types, such as the shotgun which shoots in a cone or an RPG which cause damage to surrounding areas on explosion. Weapons will also have a levelling system like a skill tree, so when you get certain amounts of xp using a certain weapon its stats are increased. Pick-ups are a system to give temporary boost or special abilities to the player to create dynamic gameplay, some pickups might give the player unlimited ammunition or temporary god mode, while some might change to the current weapon the player is using to a special weapon. Pickups usually drop from dead enemies at a random rate.

1.3.5. Skill Trees & Progression - Haoming Yuan

Skill tree is a feature in some of the games, that allow the player to custom configurations of a character's abilities. A skill tree starts with one or several base skills for different classes. After base skills, the skills split into either more specialized skills or just higher-level skills. For example, in our game can be something like HP. It will increase when player learn the HP skill in the skill tree and each point on the HP skill will increase certain amount of HP and with unlocking the HP I skill at lv10 it might allow the player to learn the HP II skill. It will increase more HP for the character.

The player's skill tree will affect subsequent game development, for different skill tree build will lead to different ways of playing. Also, in the development of the game we will consider the intensity balance between skills.

1.4. Third-Parties

There are no anticipated third parties involved in this project.

1.5. Code of Conduct

A code of conduct has been established and signed by all members of the group. The document covers many important aspects of project management such as how we will work together in a respectful and professional manner; how issues and conflicts within the group are addressed (if any occur); and how project planning will be conducted. The full code of conduct has been included under appendix A1.

1.6. Ethical Approval

Before any project can begin it is University policy that said project must first gain ethical approval. Our project was signed off as “low-risk” meaning that do not intend to involve any third parties in the development of the project, and any analysis we do will only be on secondary data which has previously been published. The ethics form has been included under appendix A2.

2. Team System Specification – Requirement Capture and Analysis

2.1. Linking Subsystems Together

To create the final game, all the individual subsystems must be integrated together. Throughout the project, version control will be used and with all group members having previous experience with GitHub (GitHub for Unity, 2019), that is the version control that will be used. During the development phase, each group member will have their own branch in the GitHub repository which will allow members to work on files specific to their individual subsystem without affecting the work of other group members. There will be multiple times during the project where subsystems will have to be integrated into the "master" branch of the repository in order to test the game. An example of this could be integrating the AI and the weapons system so that the health and damage system can be tested. It is expected that there will be the iterative process of working on a subsystem in a separate branch, merge the subsystem and test, then again working on the subsystem in a separate branch to fix any issues find during the testing.

Unity (Unity Pro, 2019) has integrated the GitHub platform so the developer can now utilize the tools provided by GitHub without having to use a separate program. There is also a feature which allows the developer to lock files to prevent unauthorized access. So, during development it is expected that each group member will lock the files specific to their subsystem to prevent accidental modification.

2.2. Packaging the Game

A requirement of the module is that all projects must be runnable on University machines. This is so they can demonstrate to teaching staff at the end of development. To demonstrate a project, a runnable file of the game should be created. Packaging the game should be considered feasible as Unity has previously published an in-depth tutorial on the topic (Unity Manual, 2018). The tutorial walks through selecting which platform to make the build for and how the build process works. To publish the game for demonstration on a University machine, the game must be built for a Windows 10 PC environment.

2.3. Making the Game Playable

Time is a limiting factor on the development of the project. Compared to fully polished games such as Geometry Wars 3 (Lucid Games, 2014), a game which had multiple years of development time by a large experienced team, we are only a team of 5 and there is just 12 weeks to plan, design, implement, test, and publish the game. To ensure that the game is playable by the end of the project each proposed subsystem has been designed to implement a major component of the game. This splits the development evenly between all 5 group members. It is the responsibility of each group member to make sure the essential features of their subsystems are implemented and that they have been tested (by following the testing strategy proposed in section 4.3 of this document) so that the game is playable in preparation for the demonstration at the end of the project.

2.4. Sound Effects & Music

Sound Is often the most underappreciated part of games design, games rely heavily on their sound to immerse the player in the universe it's trying to craft. Sound can be used to manipulate the user into feeling a certain way, for example in horror games music might slowly ramp up in tempo the further you walk down a corridor, this builds tension and anticipation. One of the other many keys uses for sounds is telegraphing attacks, this technique is used in many boss fights as a well-timed sound effect for an attack can let the player know what is coming. Games such as the run and gun boss fighting game Cuphead (Studio MDHR, 2017) use tactics like this flawlessly.

Sound design however is not an essential aspect of this project, in the requirements we decided this element of the game is a “should”, as we deem it far more important for this prototype game that we have a fully functional game without sound than focusing on sound design before the functionality of the game Is perfect. Some elements of the game may have sounds while others remain silent, for the weapons system for example it's an important part of the game feel to have reactive sounds, while it might not be necessary in the current build to add sound effects to the enemy's, or at the beginning and end of rounds.

2.5. Level Design

Fiel and Scattergood describe level design as the process of “turning an idea into a game” (2005, p.25). In the context of this project level design can be summarised as the task of creating the playable area for the proposed games subsystems to be demonstrated in. Having a playable area is essential. At a minimum this will be a small basic “demo world” made specifically for demonstrating the subsystems. If time permits a fully polished level could potentially be implemented. The level would be expected to be of high quality using professional assets and utilising an appropriate layout. To achieve this, responsibility would be split evenly between all 5 group members by having each group member implement their own distinct section of the level. Making sections of the level unique helps to make the game world more visually appealing, easier to navigate, and more interesting to play in. Potential level segments inspired by our chosen genre (a zombie survival game) could include:

- Town Centre
- Harbour
- Graveyard
- Forrest
- Courtyard
- Marketplace

Depending on the genre of the game, levels can be used with multiple aims, some levels might be made as a sort of learning level, introducing a new mechanic into the game. While others might serve as an extreme test of a player's abilities. With Sandbox games however its level must be extremely repayable, and as such usually do not hold players hands when teaching mechanics, it assumes the player will learn as they replay the game.

2.6. Requirements Specification of Common Elements

As a result of the requirements capture and analysis, table 1.7.3. specifies the requirements for the shared group elements of the product (outlined in section 1.2). These requirements outline what the game needs to be runnable and presentable for the demonstration at the end of the project. Requirements to help polish out the game have also been included, such as implementing sound design and level design; however, since these requirements are not essential for the game to be playable, they have been allocated a lower priority.

	Requirement	Priority	Justification
1.1	The game must be designed as a group showing the linking between all common elements of the game	High	To result in a working product at the end of the project
1.2	The game must be runnable on a university machine	High	So that it can be demonstrate as required by the assessment
1.3	The game must be playable and bug-free	High	It is important that the game is fully tested to ensure the demonstration runs smoothly
1.4	The game should have sound effects and music	Medium	This isn't directly related to any single subsystem, but it would help polish the game
1.5	The game could have a fleshed-out environment and playable level	Low	This isn't directly related to anyone's subsystem, but it would help polish the game. A basic demo world could be used instead

Table 2.1. - Requirements Specification of Common Elements

3. Specification of Main Functional Subcomponents

3.1. Specification of the Gameplay Programming Sub-Component

This section of the project Terms of Reference aims to analyse and capture requirements for the gameplay subsystem. Reviews of relevant literature will be conducted, and analysis of existing products from the top-down shooter genre, such as Lara Croft & the Temple of Osiris (Crystal Dynamics, 2014), and other survival game genres like Halo 3 ODST's Firefight mode (Bungie, 2009) will take place. Zegal states that segmentation of gameplay design can be broken down into several manageable components (Zegal, et al., 2008). Following this idea, the specification of this subcomponent will be split into several sections each describing a core gameplay component. These gameplay components will then be broken down into the individual units required for the game. Ultimately this research will be summarised in section 7.1 of this document through a set of gameplay requirements derived from research needed to make this subsystem successful.

Figure 3.1.1. - Lara Croft and the Temple of Osiris (Crystal Dynamics, 2014)

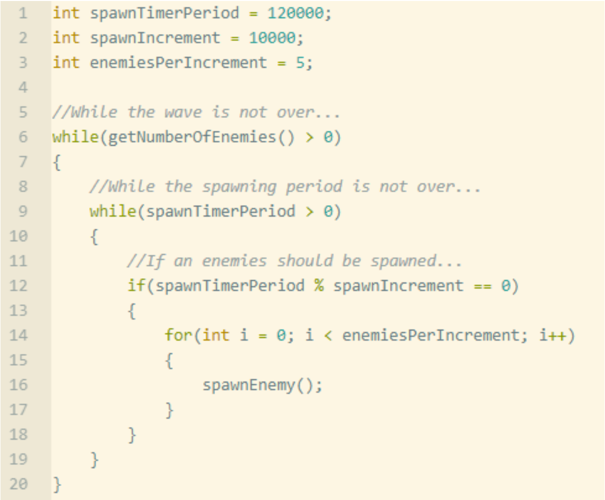
Main Mechanics

Waves & Difficulty

Wave based survival is arguably the core gameplay mechanic of the game. The is no win condition in a survival game, it is the players goal to try and survive for as long as possible. As oppose to having a pre-set difficulty, the difficulty of the survival increases of time. The longer a game lasts, the harder it gets as each wave is more difficult than the last. Initially waves are very easy, only spawning low numbers of weak enemies. As waves progress tougher enemies will spawn in greater numbers. This can be seen in games such as Call of Duty's zombies' mode (Infinity Ward et al., 2018) and Halo 3 ODST's firefight mode (figure 3.1.2) where only the weakest enemy type spawns during the first few rounds of combat.

Figure 3.1.2. - Halo 3 ODST's firefight mode (Bungie, 2009)

After analysing existing games, it is proposed that each wave of gameplay will have a set amount of time where enemies can spawn (the "the spawning period"). During this period, enemies are spawned at incrementally. Once the spawning period is over it is up to the player to destroy all the enemies that have been spawned to end the wave. For example, if the spawning period lasted 2 minutes with 5 enemies spawning every 10 seconds, a total 60 enemies would spawn in the wave for the player to combat. Variables would be used to increment the values depending on which wave the player was on. This will allow for different numbers of enemies to be spawned each wave all at different intervals. Figure 3.1.3. demonstrates some pseudo code to further explain this mechanic.



```
int spawnTimerPeriod = 120000;
int spawnIncrement = 10000;
int enemiesPerIncrement = 5;

//While the wave is not over...
while(getNumberOfEnemies() > 0)
{
    //While the spawning period is not over...
    while(spawnTimerPeriod > 0)
    {
        //If an enemies should be spawned...
        if(spawnTimerPeriod % spawnIncrement == 0)
        {
            for(int i = 0; i < enemiesPerIncrement; i++)
            {
                spawnEnemy();
            }
        }
    }
}
```

Figure 3.1.3. - Pseudo code for spawning enemies during a wave

In Gears of War 3’s Horde mode (Epic Games, 2011), boss waves occur at specific points in a game. These boss waves introduce one very tough and often unique enemy for the players to take out. An issue with survival games is that they tend to be repetitive. Special waves, such as boss waves, would help to break up gameplay providing the player with a different challenge during regular gameplay. A “wave-type” could potentially be implemented to help break-up gameplay.

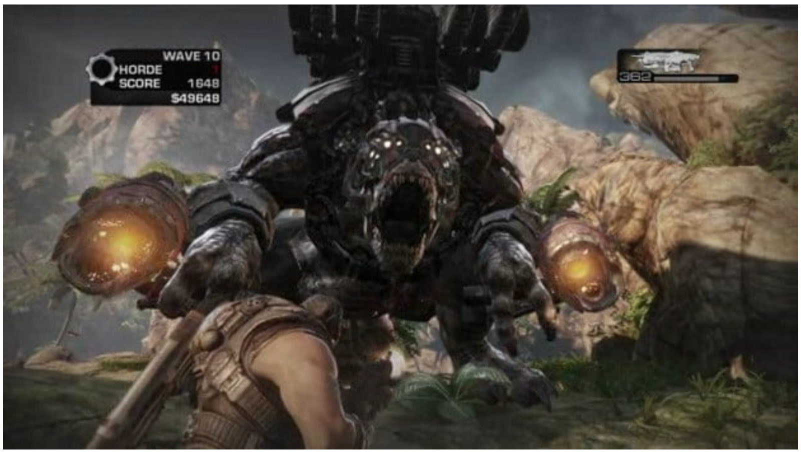


Figure 3.1.4. - Boss battles in Gears of War 3’s Horde mode (Epic Games, 2011)

Whilst some games such as Halo Spartan Strike (343 Industries, 2015) offer non-stop action, games such as Call of Duty’s Zombies franchise offer cool-down periods between rounds. It can be argued that these rounds slow the gameplay down, however they give the player some rest and allow them to prepare for the next wave. The main game should implement cool-down periods between waves, however a more difficult game mode without cool-downs could be included as an additional game type to suit different play styles (see Barrage Mode).

Enemy Spawning

The previously defined spawning algorithm (figure 3.1.3) establishes how frequently enemies spawn and how many of them spawn per wave, but the limits of this system are not addressed. To have successful enemy spawning, thorough testing will need to take place ensure this algorithm does not have a negative impact on the game. Only so many entities can be rendered on the screen before it starts to affect the games performance. What is the maximum number of enemies which can be spawned before this occurs? Once the maximum number of enemies is reached progression of the game will also need to be considered. Currently it is proposed that stats of enemies will just be buffed from this point forward to continue to challenge the player, however this could change later in implementation as a result of testing feedback.

It also needs to be considered where enemies will spawn. Triple-A games such as Call of Duty’s Zombies mode, which has enemies climb through windows or jump over fences, or Halo 3 ODST’s firefight mode, which has enemies land down in spaceships, both offer extravagant methods of enabling enemies to enter the area of play. For a smaller scale top-down shooter game, it would not matter how enemies spawned only that it is done in an appropriate style for the game. Enemies could hop over fences or crawl out of the ground, like they do in Call of Duty, or even just spawn off screen and navigate their way to the player. It is important to give context as to where the enemies are coming from. It would look unprofessional to have them simple appear out of thin air. The area of play for the proposed games is intended to be quite large as it is split into 5 different subsections. If enemies spawn in subsections the player is not in it may take them a while to reach the player, resulting in the slowing down of gameplay. The spawning algorithm to be used in the game must consider which area of the level the player is in and only spawn enemies nearby the player.

Player Progression through Direct Gameplay

Many games continually reward the player as they play to keep them invested in the game they are playing. Examples of this include gaining experience to level up your character; finding treasure chests which are filled with powerful items; or changing the state of the game world as a result of player actions. Koepp suggests that this type of gameplay releases dopamine in the players brain, a chemical responsible for motivation and reward behaviour (Koepp et al., 1998). In games such as Call of Duty's zombies' mode, the player progresses by unlocking different parts of the map to access more of the playable level. With the proposed level design in section 2.5, similar functionality could be implemented. At the start of the game the player will spawn in a closed off section of the level. As they progress through the waves of the game, the more areas they can unlock.

Typically, in-game currencies are often used to unlock items and areas of the game. For example, Call of Duty's Zombies mode includes points which you can exchange to open new areas of the map or gain access to more powerful weapons. Players earn points through killing enemy zombies, with points stacking up depending on how they play (e.g. headshots, multiple kills at once, etc.). The proposed top-down survival game should take the same approach. It may also be beneficial to award bonuses to players at the end of each wave depending on how well they played. Bonuses may include total number of kills, round bonus (the higher the round the higher the bonus), and survival (how much damage they received). These bonus points could be awarded to the player during the cool down periods between waves.

Player Movement & Controls

Player Actions

Although other subcomponents are responsible for dictating how characters move around and fight enemies, it is the responsibility of the gameplay programmer to program the players interactions with the game world and how the player interacts with the character on screen. The proposed player action would be:

- Movement (forwards, backwards, left, and right)
- Look (360 degrees)
- Fire weapon
- Throw grenade
- Swap characters
- Interaction with the game world
- Pausing the game

Peripherals

Requirement 1.2 specifies that the game must be demonstrated on a university machine meaning it must be runnable on a Window's 10 PC. This will leave the default input peripherals to be a keyboard and mouse, however the nature of "twin-stick" shooters are that they are designed to be played with a controller with input sticks. Due to this, it must be a priority that players have the option to play with either the default keyboard & mouse, or a controller if they prefer. Allowing players to choose their preferred peripheral helps to increase the accessibility of the game. Microsoft has a useful tutorial on for integrating their Xbox One controller with the Unity engine, so this requirement should be feasible (Microsoft Developers Blog, 2014).

After researching existing popular top-down twin-stick shooter games such as Halo Spartan Strike and Lara Croft & the Temple of Osiris, tables 3.1.1 and 3.1.2 have been created to demonstrate the proposed default control schemes for both the Xbox One controller and the mouse & keyboard peripherals for the game. This default layout uses a similar layout to those seen in the popular games researched. Having general control schemes across games helps give the player a sense of familiarity with the game. This helps make the game more accessible as players do not need to relearn the control scheme when switching between similar games.

Action	**Input**
Fire weapon	‘Left-Click’ Mouse
Interact	‘F’ Key
Look	‘Mouse Track-Ball'
Move backwards	‘S’ Key
Move forwards	'W’ Key
Move left	'A' key
Move right	'D' key
Pause the game	‘Esc’ Key
Swap to the next character	‘E’ key
Swap to the previous character	‘Q’ Key
Swap to specific character	Keys '1' - '4'
Throw grenade	‘G’ key or ‘Right-Click’ Mouse

Table 3.1.1. - Default control layout for keyboard and mouse peripherals

Action	**Input**
Fire weapon	Right Trigger
Interact	‘A’ Button
Look	Right Stick
Move backwards	Left Stick or D-Pad ‘down’
Move forwards	Left Stick or D-Pad ‘up’
Move left	Left Stick or D-Pad ‘left’
Move right	Left Stick or D-Pad ‘right’
Pause the game	Start Button
Swap to the next character	Right Bumper
Swap to the previous character	Left Bumper
Throw grenade	Left Trigger

Table 3.1.2. - Default control layout for Xbox One Controller

The default control scheme will not be the only control scheme available. Several preset control schemes will be made available and there should also be the ability to create custom control schemes by binding keys, buttons, and joysticks to in-game actions. It is important to provide many different input options as all players are different. A player with disabilities such as missing limbs or limited mobility should not be excluded from playing the game. Many different control schemes should be available to avoid this social issue. It should also be noted that it will also be playable with Microsoft’s newly released adaptive controller (a unique controller designed to provide accessibility for all) since it receives all the same inputs as the Xbox One controller (Microsoft, 2018).

Figure 3.1.5. - Xbox Adaptive Controller overview (Microsoft, 2018)

Game Modes

Different game modes have the potential to be “out of scope” for this project as they would be separate from the main game. However, they enable extra challenges for the player and more options for players with different play styles making the game more accessible.

Time-Attack Mode

Time-Attack is a popular mode throughout all video game genres. In time-attack the concept of time is a resource and is used to challenge players to see how quickly they can complete the game. Famous examples include Sonic the Hedgehog (Sega, 1991) and GoldenEye 007 (Rare, 1997), both games which reward players for revisiting levels and completing them in target times. Adapting a time-attack game mode to the top-down survival genre could be done making time a resource that needs to be collected in order to progress through that level. For example, in this game mode the player would have a 60 second timer, once the timer reaches 0 the game ends. Extra time is awarded to the player for progressing through the game, examples of this may include killing enemies or successfully completing waves. The better the player is at the game, the further they can progress. The goal of the time-attack game mode is to see how far players can progress through the game before time runs out.

Barrage Mode

In the military a barrage is a large concentrated attack on specific area. As previously mentioned, a more difficult game mode could be implemented where players do not have the luxury of “cool-down” periods between combat waves. This would be barrage mode. It would be available for players who have mastered the main game mode and want more of a challenge. Barrage mode is intended to put the players in a situation where they are constantly under pressure and must have complete focus to be successful. Halo’s firefight mode allows the players to select modifiers to make enemies stronger in the game, which is balanced by awarding the player with more experience points for their in-game achievements. Traits, such as giving enemies double health, faster speeds, and stronger attacks, could potentially be introduced into barrage mode. Awarding players with more experience for their actions on harder difficulties would also be an alternative way to help level up your character quicker. Arguments could be made that this game mode could be unethical as the player would be under an extreme amount of pressure; however, this game mode adds a new level of challenge for players and is completely optional. As a precaution they player will be fully informed of the game types contents before they are able to play it, so they will be aware of what the game mode entails.

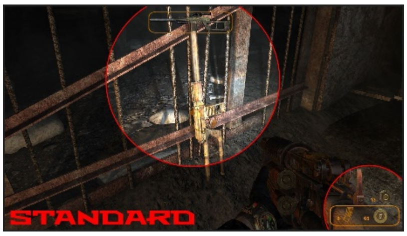
3.2. Specification of the HUD & UI Sub-Component

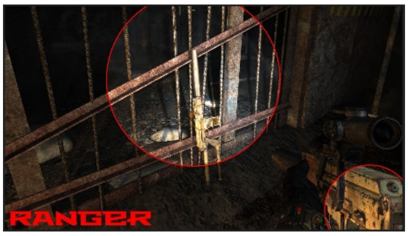
The main goal of this section is to conduct an analysis regarding the current subsystem, “HUD and UI”. In order to achieve this goal, research of literature reviews must be done beforehand. Aspects like methods that are currently used in the industry which captures the attention of players and controls their behaviour through 2D art will be investigated. As a follow-up of this section, the requirements will be provided within section 7.2.

HUD

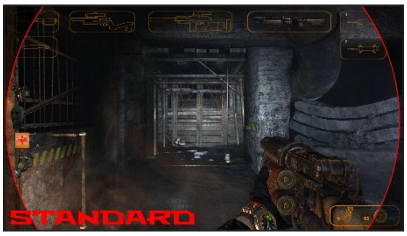
Even though the project is intended to work as a Top-Down zombie survival, this does not interfere with the way HUD is going to be designed and implemented in comparison with other types of games like FPS (First-Person-Shooters), TPS(Third-Person-Shooters), etc. By analysing titles that made use of enemies which are represented by hordes of zombies such as: Left 4 Dead 2, Metro 2033 series, the “Dead Ops Arcade” from inside Call of Duty: Black Ops, or even Dead Island: Epidemic which also has the closest resemblance with the current project’s idea will provide enough info on how to deliver a user-friendly HUD as well as a HUD which is not too bloated with 2D art that defocuses the player from its main target.

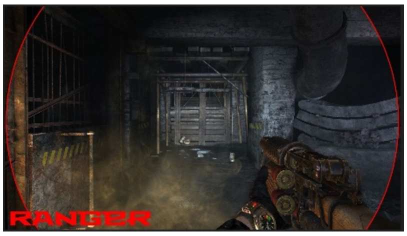
Some games like the Metro 2033 series have gone one step further and implemented a new type of difficulty (which is called Ranger) that gets rid of the entire HUD and allows the users to get lost in their mesmerising post-apocalyptic world of Moscow and its undergrounds. Some game visual snippets (Heads Up Display | Metro 2033, N.D.). are snowed below, and they clearly demonstrate how helpful the presence of the HUD can sometimes be.



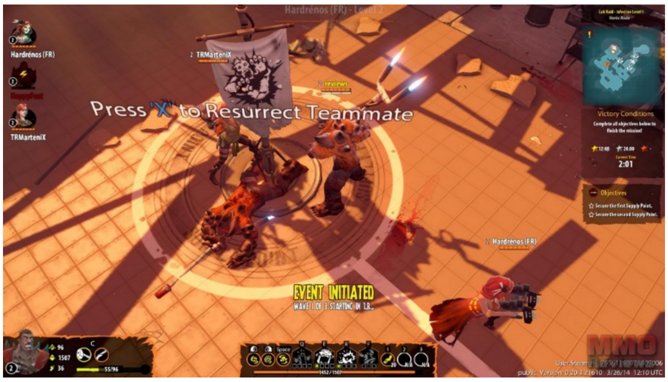


In the first set of two pictures, the weapon is easily noticeable but sometimes items are hidden around the map which makes it harder for the casual player to detect helpful items like ammos and bandages.



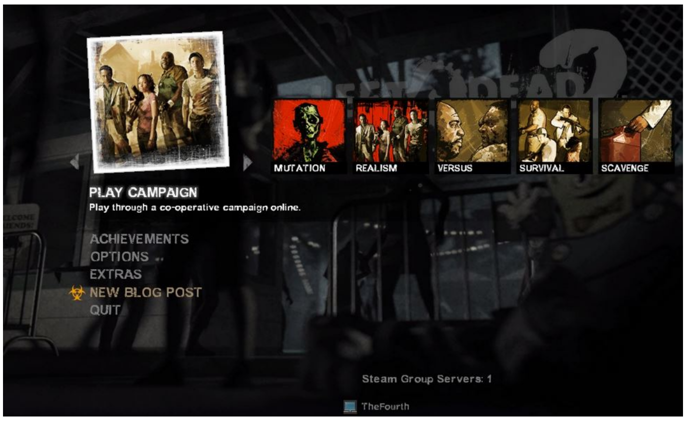


As I have previously mentioned, Dead Island: Epidemic (MMO Games, 2015) has been a game that unfortunately failed to pass the stage of OB (Open-Beta), but it still managed to deliver some features that are similar to the current project’s ideas like: non-intrusive HUD full of useful information about other allies from the team which sits in top left side of the screen, easy to access skills as well as skill-tree on the bottom of the screen, descriptive mini-map and live information about objectives that is accessible on the top right of the screen, etc. Another important feature of the game is regarding its game mechanics. The way it is played is as a Top-Down Third-Person Camera zombie game and the PvE (Player-versus-Environment) part of the game integrates some of the functionalities presented in this document. The HUD of the game becomes more descriptive (sometimes too descriptive as it can be seen in the picture below where dead allies hold a message above their body that is somewhat intrusive and can potentially distract team members from the actual objective of eliminating the surrounding threats) and serves for a different purpose when the game is played in its PvP (Player-versus-Player) mode but as long as this functionality of integrating online features is not a high importance requirement of the current project plan, it remains under 'Future Work and Recommendations'.



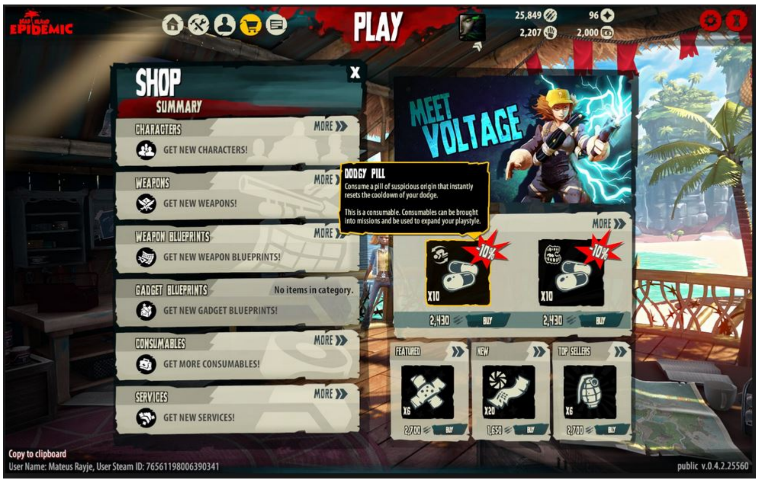
UI

UI plays a major part for what 2D Art stands for. It is the first thing players interact with and they constantly use it in order to change difficulties, settings, game modes, save current progress or load previously saved game data, etc. This means that the overall aspect of the Main Menu should be visually pleasant and closely related to the game's idea. Most of the games that are released nowadays make use of main menus with dynamic backgrounds like the one which is showcased inside Left 4 Dead 2 where players see pre-rendered scenes with the infected lurking in the dark that sets the base idea of the game.



Play through a co-operative campaign online.

Moreover, in another game that I have previously discussed, Dead Island: Epidemic, the Main Menu may feel overwhelming for a person who has not played it before when you compare it to the title above, L4D2. The actual useful features that allows players to quit, change settings, check user profile are minimised on the top part of the screen whereas lots of information regarding the shop takes most of the space. When looking back over to the Main Menu from inside L4D2, it is easily noticeable why information should not be pushed on a single screen but divided in sub sections that are easily interchangeable.



DODGY PILL

Consume a pill of suspicious origin that instantly resets the cooldown of your dodge.

This is a consumable. Consumables can be brought into missions and be used to expand your playstyle.

User Name: Mateus Rayje, User Steam ID: 76561198006390341

Summary

Having a wide variety of games that made use of Top-Down Third Person functionalities like the ones that were previously mentioned, it supplies enough valuable information for the design and implementation stages of the HUD and UI to benefit from choosing the constructive features that were used in the above titles. After having analysed each individual segment of relevant titles, a set of proprietary requirements will be written in more details as well as classified under high, medium or low priority depending on the overall needs of the product's idea, in the section 7.2.

3.3. Specification of the Characters & AI Sub-Component

The aim of this section is to analyse and capture requirements for the "Characters & AI" subsystem. In order to capture requirements, the artificial intelligence in existing video games will be explored. The project is to develop a top-down zombie survival shooter, so it is important to explore games with similar themes and/or features to the project vision. For example, games which feature zombie AI opponents, friendly AI characters or player character switching functionality. The research conducted in this section will lead to subsystem requirements that will be specified in section 7.3.

Enemy Artificial Intelligence

Left 4 Dead 2 (Valve Corporation, 2009) is a first-person survival horror shooter game and in the campaign mode of the game, the player must fight against hordes of zombie enemies referred to as "infected" and reach safehouses which act as checkpoints. The game features "infected" which are the typical, regular zombies you would expect to find in a zombie themed game, and "special infected" which are unique characters with distinct AI behaviours.

The "infected" opponents have AI behaviour that players really expect from a zombie character. The behaviour being that when the player is in sight, the zombie will run towards the player then melee attack the player when in proximity. Another AAA title, Call *of Duty: Black Ops* (Activision, 2010), features the min-game *Dead Ops Arcade* that closely resembles the project vision as it is a top-down shooter with the objective of surviving waves of zombie opponents. Like in *Left 4 Dead 2*, the zombie opponents move towards the player then attack when close enough. However, in *Dead Ops Arcade*, as soon as the enemies spawn, they are aware of the players position so there is no way for the player to become hidden or evade the enemies. In *Left 4 Dead 2*, the zombie behaviour is more complex as zombies can roam the surroundings and exhibit behaviours such as sitting down or vomiting for example. So, the zombies appear to have no knowledge of the player's position until they are alerted.

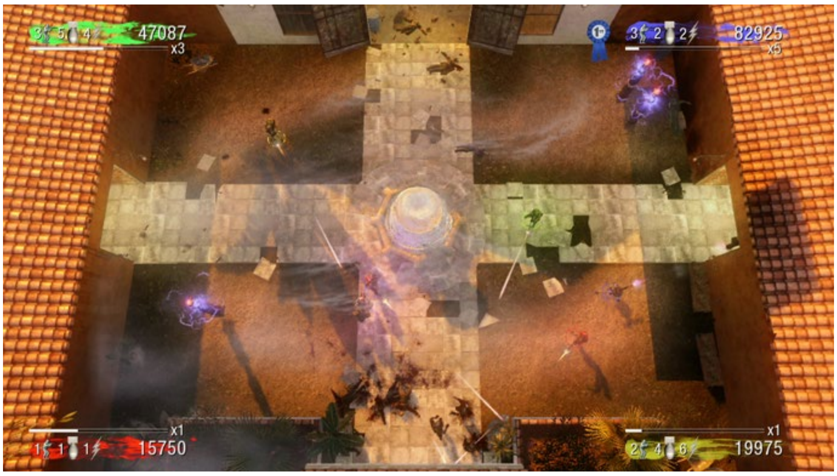


Figure 3.3.1 - "Dead Ops Arcade" in Call of Duty: Black Ops (Activision, 2010)

In *Left 4 Dead 2*, when the player changes the difficulty of the level, the behaviour of the AI opponents remains unchanged but the damage that they inflict to the player increases instead. So regardless of the selected difficulty, the zombies will always run towards the player and attempt to attack. But in *Dead Ops Arcade*, although there are no difficulty options, in the early rounds of the level the zombies stagger and walk towards the player, hence making it easier for the player as they are given more time to make decisions and are less likely to be surrounded by a horde of enemies. As the player progresses through the rounds, the speed of the zombies steadily increases and eventually plateaus so despite having no difficulty setting, the difficulty gradually increases as the AI behaviour changes.

As previously mentioned, *Left 4 Dead 2* has multiple enemy types with unique behaviours and a similar idea could be implemented in the project. Having multiple enemy types will give the player more challenges and will require them to make more decisions such as which enemy type will be prioritized for example. To implement this in the project more work would be required, and it will be another feature that will require balancing along with weapons to make the game fun yet challenging. A feature like this is not a priority requirement but more of an optional one which certainly has the potential to improve the player's experience and hence improve the game.

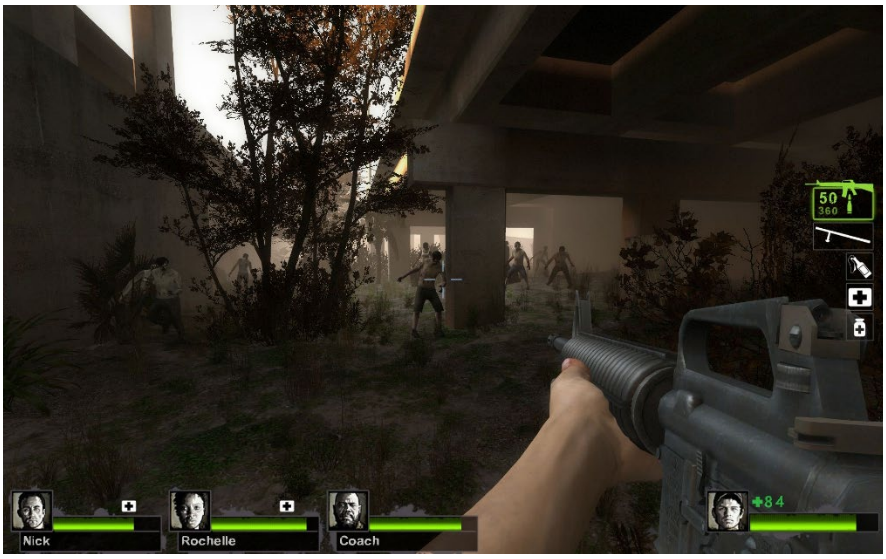


Figure 3.3.2 - Left 4 Dead 2 (Valve Corporation, 2009)

Friendly Artificial Intelligence

In single-player shooter video games, AI is essential to provide both friendly characters and opponents with pre-defined behaviour. The use of friendly AI can include helping to guide the player through the environment, aiding with storytelling and further immersing the player in the game, or to simply help the player defeat opponents. *Left 4 Dead 2* provides the player with three friendly AI characters and collectively form what are referred to as "the survivors" in-game. At the start of each campaign, the player plus the three friendly AI begin the level together and the AI helps the player fight off zombies, revive and heal the player as well as transfer consumables to the player. The AI behaviour is complex the ability to transfer items to the player for example isn't necessary for the project, but there are some fundamental behaviours that are. The first, fundamental behaviour is that the friendly AI will always try to stay close to the player to be able to protect them. The second is that they can kill enemy zombies meaning that they can protect the player.

Conflict: Desert Storm (Square Enix, 2002) is a third-person tactical shooter which also features friendly and enemy AI. Like *Left 4 Dead 2*, the player starts the campaign with three friendly AI characters. With this game being a tactical shooter, the player is given more explicit control of the positioning of their friendly AI counterparts. This is achieved by allowing the player to issue commands to all characters are even just one. For example, the "halt" command can be used to stop all characters or a specific character. The command functionality isn't required for the project but the fundamental behaviour of the AI targeting enemy players is. The main purpose of the AI here is to help the player defeat opponents by killing enemies and protecting the player. To do this, both titles will have their own system in which the AI makes decisions – presumably the system is some sort of behaviour tree. In the behaviour tree, there will be logic that decides which enemy to target. For example, the AI may select the nearest enemy to themselves, the nearest enemy to the player or there may be a more complex system in place that evaluates all the enemies and decides which one is of the biggest threat.

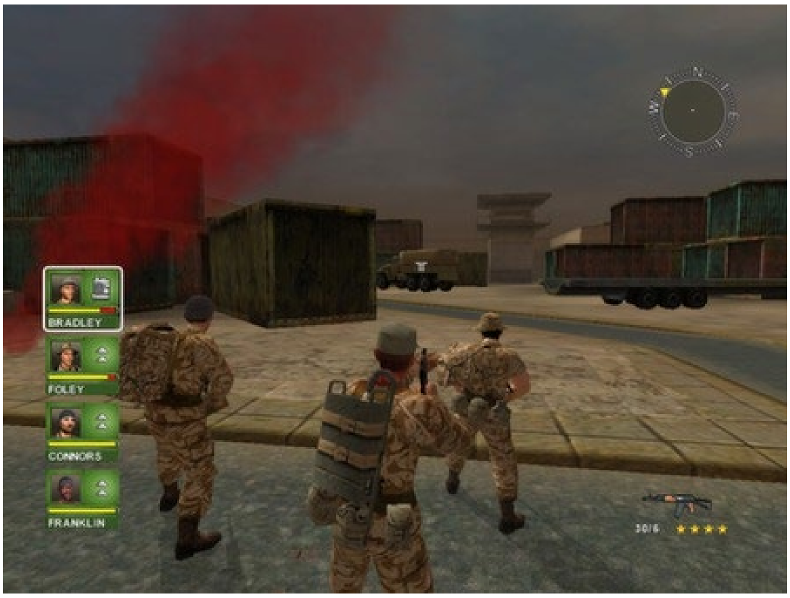


Figure 3.3.3 - Conflict: Desert Storm (Square Enix, 2002)

Character Switching

The ability to switch between playable characters during a level is absent in *Dead Ops Arcade* and *Left 4 Dead 2* but not in *Conflict: Desert Storm*. In this title, the player can select any of the four characters to play as during the level – assuming all the characters are alive. This allows the player to take control of any character and move them to a specific position. When the player relinquishes control of a character, the AI then takes control of the character. This means that in the game, the player could take control of the character with the sniper and position him, then switch to another character and let the AI now controlling the sniper character target enemies but remain in the position that they player positioned him in. Since the game is a tactical shooter this functionality is required but in a top-down shooter like the project, it would be more of an optional feature.

Summary

There are many existing video games that feature both friendly and enemy AI as well as zombie opponents. So, during the development phase there are many sources to draw inspiration from, but the titles mentioned above are the main sources as they closely resemble selected features that are intended to be implemented in the project. To implement any artificial intelligence behaviour in the game, the AI tools and techniques supplied by the Unity (Unity Pro, 2019) engine will have to be utilized. The decision making of all AI entities will be controlled using behaviour trees, and AI movement will use the navigation system which itself implements the A* pathfinding algorithm. (Unity Manual, 2018). The must should and could level requirements constructed as a result from this research will be stated in section 7.3.

3.4. Specification of the Weapons & Pick-ups Sub-Component

Weapon systems are often the focal point of games, especial of games in the wave-based survival genre, in order to capture requirements, or this weapons system similar systems in existing video games will be explored. The project is to develop a top-down zombie survival shooter, so it is important to explore games with similar themes and/or features to the project vision. There are a few games that our project vision draws huge amounts of influence from, these being “Dead ops Arcade”, a zombie survival minigame from Call of Duty: Black Ops (Activision, 2010) and Halo: Spartan Strike (343 industries, 2015).

Dead Ops Arcade:

As its name might suggest, Dead Ops arcade (Activision, 2010) Is a homage to old arcade games such as “Contra”, both being run and gun shooters with similar gameplay. The weapons system in Dead ops Arcade conforms closely to a typical arcade shooter, being that you spawn into the game with one basic weapon, usually without an ammo capacity to aid with the “run and gun” mechanics, and throughout the game the players either pick up new more powerful weapons, which unlike the base weapon usually have some sort of charge or ammo capacity. Dead ops use a power up system which meshes with the weapons system as some of the power ups give player better weapons, such as a shotgun or rocket launcher. As however this game mode is a “Easter egg” for a larger game this game mode isn't highly developed, nor meant to be played as a stand-alone game, where mechanics such as levelling, and weapon selection might be preferable.

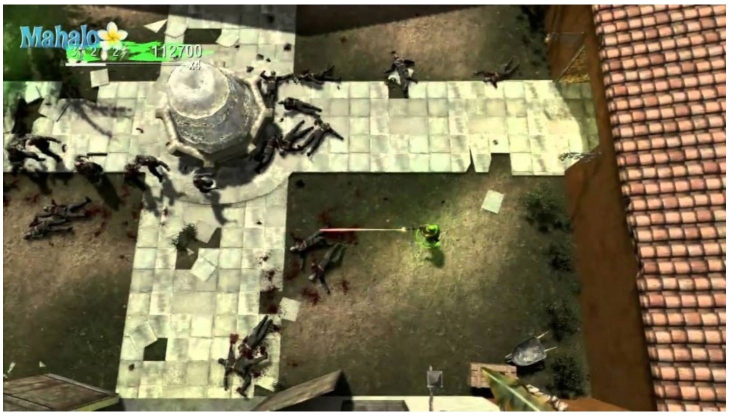


Figure 3.4.1 - Dead Ops Arcade (Activision, 2010)

Halo: Spartan Strike & Halo: Spartan Assault:

These two Halo games break from the franchises usual formula as these are top down shooter game made for Windows and windows phones, as this is a standalone game it has different features than that of Dead Ops Arcade, in these games' weapons are selected before the start of the mission and the player is stuck to those weapons for that mission, Along with shooting, this game implements a system found in other halo titles, the ability. Each player can select from one of a few abilities, which can be used at any time bar a short cooldown, abilities can be something such as a short sprint, or a bonus temporary shield.

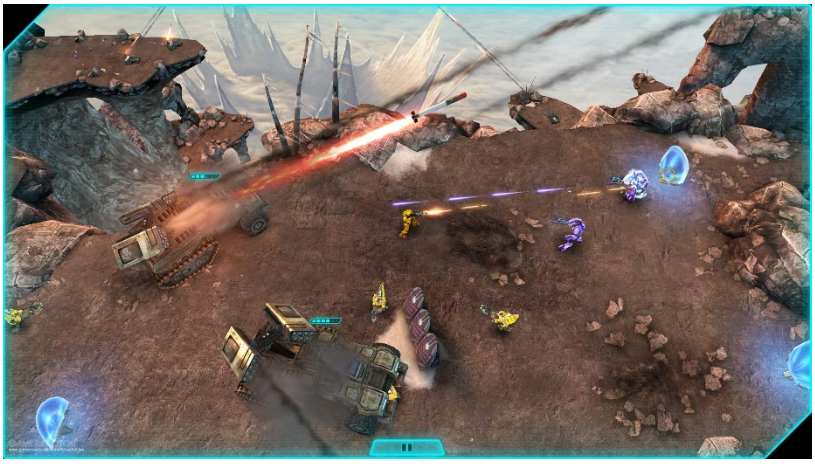


Figure 3.4.2 -Halo: Spartan Assault (343 Industries, 2015)

Weapons:

Weapons are one of the most important aspects of a survival game. Games such as Left for Dead (Valve, 2008) and Killing floor (Tripwire Interactive, 2009) require their weapons to feel responsive and satisfying to use, what makes weapons satisfying to use however Is sometimes hard to decipher. I am sure most people will say the first three games in the Halo franchise have a great feel about the weapons; but why? Halo revolutionized first person shooters on consoles by adding a mechanic called aim assist, while first person shooters excelled on mouse and keyboard it was always notoriously difficult to aim on consoles due to the lack of diversity offered by the thumb sticks, therefore for Halo: Combat Evolved (Bungie, 2001) helped console players by adding a leeway to their aim, meaning if they were aiming close to an enemy it “snapped” to them creating a soft lock on, increasing responsiveness and accuracy, a turning point for console shooters.

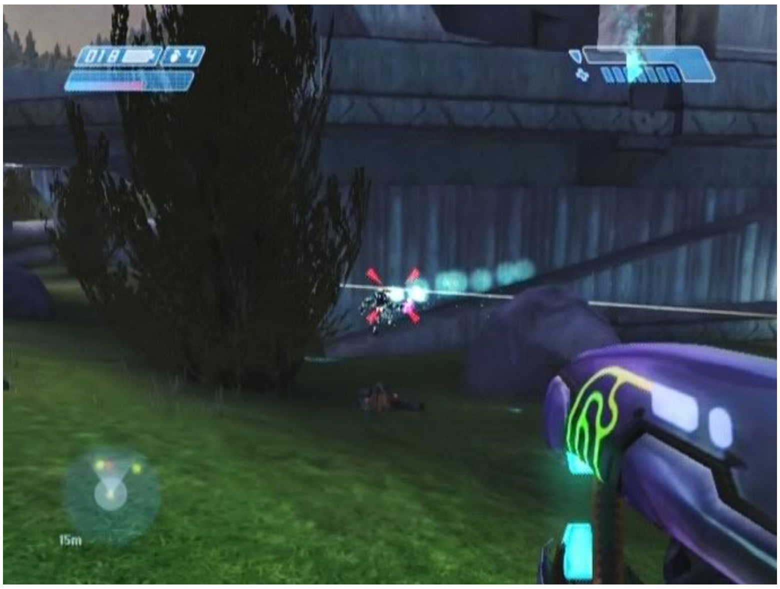


Figure 3.4.3 -Halo: Spartan Assault (Bungie, 2001)

The most important factor that makes weapons feel good is visually feedback, when the play shoots an enemy knowing the enemy has been hurt by the shot is key feedback, whether that be a blood spurt or a hitmarker. Some Arcade style games have a hit bubble pot up above where the damage is done, this is seen in the RPG game Borderlands, this adds to the punch of hitting an enemy as you can visually see how much damage you have done.

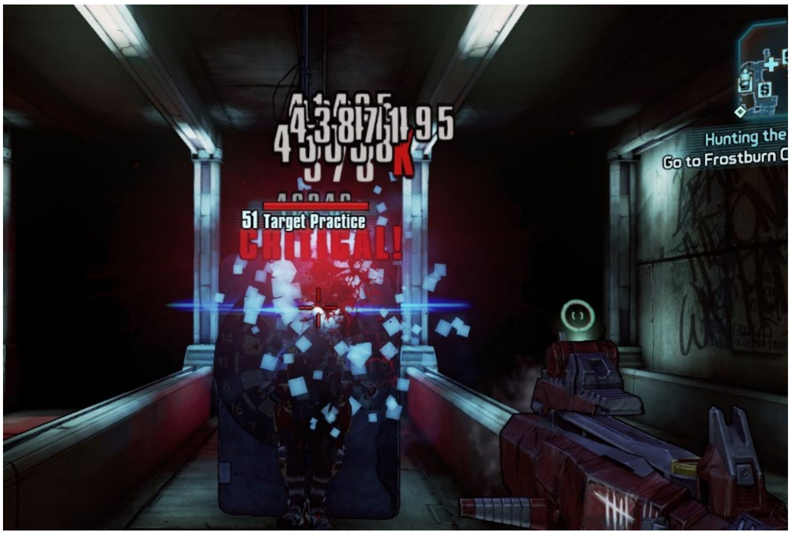


Figure 3.4.4 -Borderlands (Gearbox Software, 2009)

In this project there will be multiple weapon choices for each of the following weapon types:

Assault Rifle's
Shotgun's
Rocket Launcher/ Explosive's
LMG's
Sub-Machine Gun's

Each will have both visual and functional differences; a large array of weapons may help facilities more styles of play any play may choose to go down. For example, one assault rifle night has lower rate of fire but more on hit damage, making it more effective in the earlier round when there are less enemies. While other weapons from the same class will be locked behind the skills and progression tree implemented by another group member. The weapons in this project will all work a very similar way to "dead ops Arcade", being that the weapons always shoot in the direction to player is facing with the player facing where the mouse cursor is located, this functionality will be implemented by the gameplay programmer. I will be implementing the abilities to fire weapons, damage enemy's and Pick-up power ups

Pick-ups:

As discussed earlier, similar styles of games such as Dead ops Arcade uses power ups to add different dimensions to the game, as well as allowing the user some much needed assistance the further into the levels you go. Power ups can affect what weapon the player has equipped, replacing the base weapons with a much more powerful special weapon, which has a time limit. Some potential power ups are listed below.

God Mode

Extra Life's

Double Damage

Double Points

Ray-gun

RPG

Clear Map of enemy's

Weapon Management:

There have been many ways games accommodate their weapons systems, come typical first-person shooters such as Call of Duty employ a simple system in which players have one primary weapon; usually an assault rifle, sub machine gun or sniper rifle and one secondary weapon; usually a pistol. This system works well with their class creation system, allowing players to have up to 10 different assortments of primary and secondary weapons for the player to switch between in game. While other games use a "weapon wheel", the wheel is a large interface that allows for the selection of many more weapons then a simple "switching". This system makes more since in role playing games such as red dead redemption, where the player has more items than just weapons, with bandages and lassos taking up weapon slots.

This projects system will be more arcade style like that of the contemporary "Dead ops Arcade", reducing the amount of choice the player has helps reduce the complexity of the game and helps balance the style of the game. The way I imagine the weapons to work in this project is like most other Call of Duty zombie maps, where the plyer spawn into the game with a pistol, and then they must "buy" weapons laying around the map with points earned in game by killing enemies. However, if testing proves that this method harms gameplay, an alternative system such as that in Halo: Spartan Strike could be implemented, where by players select two starting weapons and stick with those for the game, excluding certain power ups which change weapons temporarily.

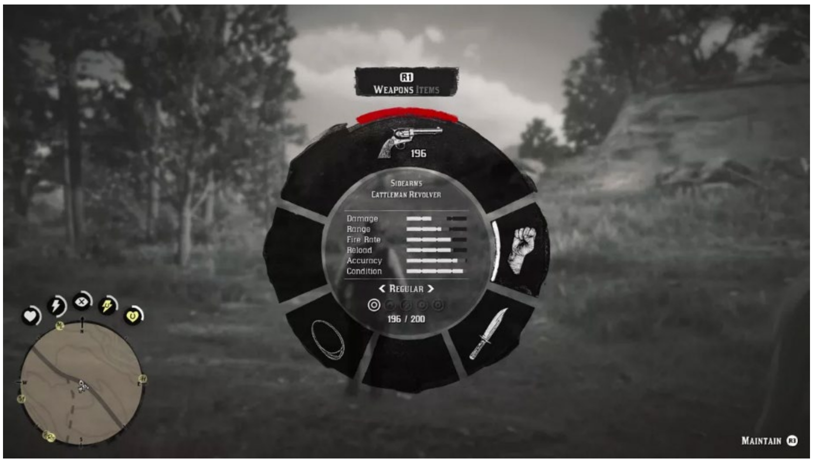


Figure 3.4.5 -Red dead redemption 2(Rockstar, 2018)

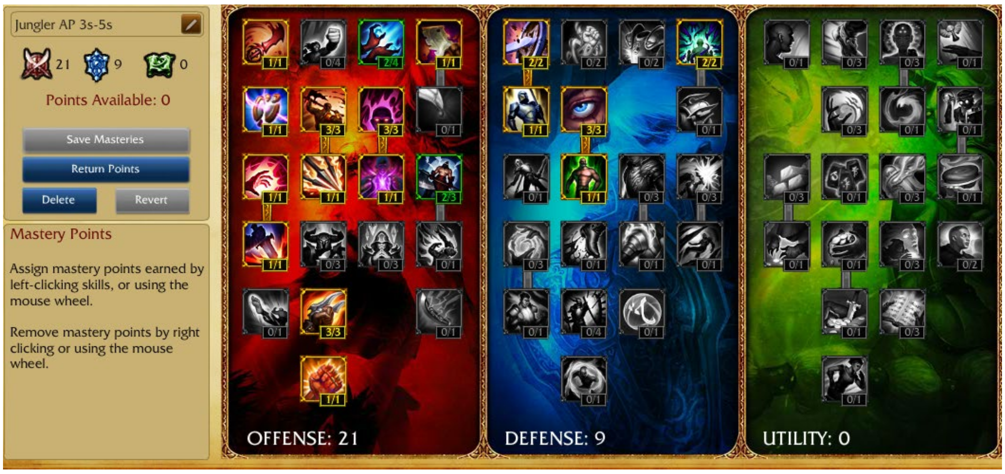
3.5. Specification of the Skill Trees & Progression Sub-Component

The purpose of this section is to analyse and capture requirements for the subsystem "Skill Trees & Progression”. In order to do this, research and learn how they implement these features in the existing gaming in the game industry. The research conducted in this section will lead to subsystem requirements that will be specified in section 7.4.

Skill Tree

The skill tree is an indispensable feature in many games today. This feature exists to increase the fun of the game, the freedom of the game, and the balance of each level to the player. Skill trees are potent balancing points designers can use to throttle advancement in a game, while giving that extended play time more meaning. Adding different skills or different skill build can give players more choice. The best skill trees are the ones that encourage player to level and allow player to play the game in the way they enjoy it most. This personalization also satisfies the common needs of the game and has a powerful and diverse skill tree system that allows you to turn the content of a single game into an endless sandbox of gameplay discovery.

league of legends one of the most popular game in the world. It has a mastery skill tree allow the player to have a set of different stats, it can be pre-set before the game start. It can change the different skill build according to the different champion the player plays. So that give the player a lot of choice, can create a lot of different new ways of playing the game. Also, in the skill Tree it has different tier of skill some of them are require maxing the level of low tier skill to unlock the higher tier skill in the skill tree and usually high tier skill give more stats compare to the low tier skill.



Mastery Points

Assign mastery points earned by left-clicking skills, or using the mouse wheel.

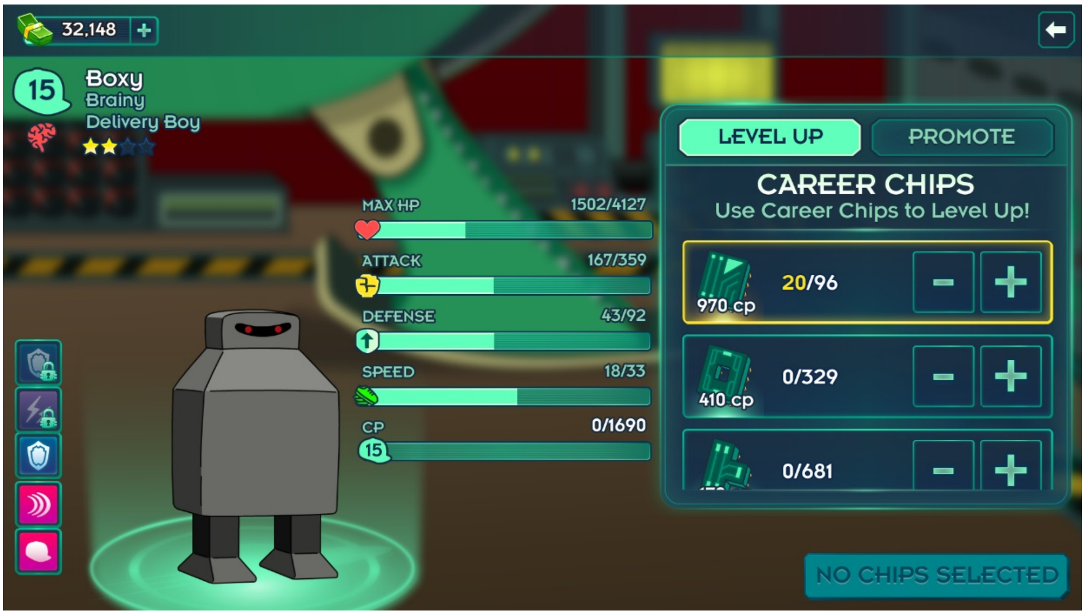
Remove mastery points by right clicking or using the mouse wheel.

Xp & progression system

Progression is one of the important game design elements that guides users to mastery and accomplishment. It can be represented in games and game applications in many ways, and progress bars and levels are the two most common ways to indicate progress.

One of the mechanisms used to make progress in many games is experience points or XP. It is especially common in many games today, and it is often used in game design. In our game It is going to be a wave zombie survival game. The enemies will be getting stronger and stronger over time. So, after the player kill the enemies it will get Xp points to level up and every time the player level up will gain some skill point allow the player to upgrade their skill in the skill tree. So, the player strength can be improved as well by the time goes.

In the League of Legends, the level difference between champion and champion will give the player a distinct amount of advantage when they play against each other. Even if it is 1 level difference, the Xp system in different games will have different effects on the game play.



In our game the level up speed and the stats in the skill tree will be as balance as possible to this game. Not making the game either too easy or too hard for the player. Every time player level up will get some skill points and some more stats on the character.

4. The Project Tasks and Deliverables

4.1. How the Development Lifecycle will Work in the Project

Agile methodologies will be integrated into the development lifecycle of the project. In agile development, phases of the project can be revisited to improve upon specific areas of the game. For example, a section of the game may be tested, depending on the outcome of testing that section would be redesigned and reimplemented to improve upon the overall quality of the game.

Where applicable, Beck's eXtreme Programming (XP) methodology will be followed (Beck, 1999). XP proposes that a project leader is appointed to manage and coordinate team activities. This will ensure good communication between the team, helping to link together the subsystems defined in section 1.3.

The following subsections outline how the project fits into the chosen development lifecycle.

Planning

Initial concepts of the game and the requirements needed to make it a reality will be planned out early in the development lifecycle. Research will be conducted into how these requirements will be met and any potential legal, social, ethical, or professional concerns will be addressed. All of this will be encased within deliverable 1.1 the Project Terms of Reference (this document).

Design

XP encourages simple design. There is no need to design the game in large detail as designs can be revisited throughout the implementation and testing phases of the development lifecycle. A small game design document (deliverable 2.1) will be created to give an initial concept of what each element of the game will look and play like.

Implementation

Most of the project time will be focused on implementation. Here all group members will program their individual subsystems and link them together into the final product (deliverable 3.1).

Testing

Since an agile development methodology is being utilized, testing and implementation will occur simultaneously. This allows for bugs and errors to be caught early on in development, helping to create a high quality robust and reliable game. A full testing strategy will be specified in section 4.3. Evidence of testing will be submitted under the project deliverable 4.1

4.2. Resource List

PCs with peripherals

Each member of the group will require their own PC with all the required software installed. These PC will be used throughout the duration of the project.

Xbox One Controller - (for gameplay programming)

The gameplay programmer will require an Xbox One Controller to demonstrate requirement 2.1. With the controller they will be able to make the players character navigate the game world.

Adobe Creative Cloud 2019

The most to date version of Adobe Creative Cloud will be required. Programs such as Illustrator and Photoshop will be used to draw graphics and mock-ups for the game, Audition will be used to record and edit sound effects, and Acrobat will be used to manage the PDF documentation used for game design and project management.

Microsoft office Business 365

Document templates, UML, project plans, and any other forms of documentation will be created using the applications packaged with Microsoft Office.

Unity (Version 2018.3.6)

As outlined in section 1, the project will be developed in Unity. It is important that all users work on the same version of the engine (2018.3.6 - the newest release) for smooth and consistent development in case features and code changes between versions.

GitHub Desktop

GitHub Desktop will be used to manage the projects repository, allowing all group members to access work. It allows for version control and backs up our work online. Group members can also work on their own branches without affecting the rest of the repository.

4.3. Testing Strategy

To ensure the agreed project deliverables are of good quality, a test strategy needs to be put in place. The following section outlines the test strategy defined for this project.

Both Test-First Development (TFD) and Test-Driven Development (TDD) will be used for testing the product. TFD is when test cases are written prior to implementation. Having pre-defined test cases means that there will be less ambiguity regarding what needs to be implemented. This testing procedure will be used on a high-level to ensure both the group requirements (section 2.6) and the individual requirements (section 7) outlined in this document are met successfully. In TDD many very small tests are executed during implementation. Every time a test is passed, a new test is added. Continually testing the product in this manor allows for each component of the game to become more and more robust over time. The recursive nature of TDD means that it is adaptable for adoption in the agile methodology being utilised for development (see section 4.1).

Individual tests will be performed using both compliance-directed testing and fault-directed testing. Compliance-directed testing is the process of examining how closely components of the game conform to the requirements of the system. This type of testing will mostly be used alongside TFD for evaluating project requirements. Fault-directed testing is used to detect errors in the system. Tests aim to reveal errors by pushing the boundaries of the game to discover its limits. All potential interactions the play could have with the game should be considered.

Ensuring testing is performed in a professional manor, group members will not test their own subsystems of the game. Instead they will test out what other members have implemented. This removes any bias from testing, allowing for honest and reliable feedback when evaluating the project requirements.

Combing all the methods of testing covered will help to create a complete robust and high-quality product by the end of the development lifecycle.

4.4. Risk Analysis

A risk assessment form following the Modules standard risk assessment template has been included under appendix A3.

4.5. Project Plan

The gantt chart has been included under appendix A4.

4.6. Agreed Deliverables

Planning

Deliverable 1.1: Project Terms of Reference

This Terms of Reference document acts as evidence of project planning. It encases a code of conduct; a Gantt chart forming our project plan; a risk assessment form; as well as legal, social, ethical, and professional consideration.

Design

Deliverable 2.1: Games Design Document

A small game design document will be produced at this stage of the development lifecycle. It will detail what the final game will look like and how it will play, bringing together all common and individual elements of the project.

Implementation

Deliverable 3.1: End Game

As a result of implementation, the end game will be submitted and demonstrated on a University machine.

Testing

Deliverable 4.1: Testing Logs

Since we are using an agile methodology, testing and implementation will occur simultaneously. Throughout implementation, testing logs as the game is tested.

5. The Legal, Social, Ethical, and Professional Dimension

The United Kingdom enforces the PEGI rating system for the classifications video games, games rated for 12's and over often contain non-graphic violent to humans or animal characters, significant nudity or bad language. Games rated 16 and over usually include heavy depiction of violent or sexual activity which looks like real life. While 18+, the highest categorization for games are rated that way usually due to a 'gross' level of violence likely to cause the viewer to feel a sense of revulsion. We expect our game to contain a moderate depiction of violence, but the depiction will ultimately depend on the assets available to use in the game. These factors make the game more accessible to a wider audience while also causing less controversy. Although we cannot say for certain what the PEGI rating of our game would be until it is officially rated, we are confident in our prediction that it will be classified as a PEGI 16 due to the similarities between our game and similar games which are usually rated 16+.

Video games such as Grand Theft Auto are often criticized contributing to a desensitization of youth, through the depiction and glorification of excessive violence. Whether or not it can be said that games companies have an ethical responsibility to curve their depictions to protect vulnerable individuals can be debated, however due to the nature and potential content of the game, we are confident that no such social issues should arise.

During the development phase of the project, assets sourced from online will be used. These assets will include sounds, 3D models for example. Assets can be purchased and in turn you acquire the license to use the assets in your projects. However, for this project we will only be using assets that are free to use. We must ensure that the license associated with the asset grants us access to use the asset in the project. This is a legal issue that exists within the project and we must not infringe any copyrighted assets at any point during the project.

6. Costings

This section will estimate how much the development of this project will cost. It assumes we are an established business looking to work on the project full time for 12 weeks (3 months – 90 days / 60 working days).

6.1. Running Costs (Per Day)

Office Space

Our office is located at the Quayside, Newcastle Upon Tyne. We use 540sq ft of space valued at **£199 per calendar month per desk**. The total rent for the duration of the project will be **£2985**. Daily running costs of office space work out to **£32.45** (£5985 / 90 days rent).

Software Costings

Payment for the software specified in section 2.3 is needed for development on the game. This includes Microsoft Office 365 Business (**£7.60** per user per month), Unity Pro (**£96.69** per month), and Adobe Creative Cloud Business (**£50.57** per user per month). Multiplying these values by 5 (for each staff member), then by 3 (for each month of development) results in a **total of £1,163.45**. Dividing this sum by 90 (the number of days we are working on the project) results in a daily software running cost of **£12.93**.

Insurance

The company Digital Risks has quoted us with **£64.54 per month** for insurance covering up to £1,000,000 in public & product liability, £10,000,000 in employer's liability, £5,000 insurance for our hardware, £500,000 for professional indemnity, £250,000 for Cyber Security, and £100,000 for commercial legal protection. Insurance will cost the company **£2.15 per day**.

Total Running Costs (Per Day)

The total running cost per day will be **£47.53**.

6.2. On-Cost Wages for an employee per day

Employee wages

Each staff member will be earning the **living wage** which is **£9** per hour (Living Wage Foundation, 2019). Working 8 hours per day results in a wage of **£72** per day (**£360 per working week**).

Employee National Insurance

Since all our staff are under **National Insurance bracket A**, **12%** (Gov.uk, 2019) of each employee's weekly wage will be paid for the 2018-2019 tax year. This works out to each employee paying **£8.64** to National Insurance each day. Their remaining pay slip will be **£63.36**.

Employer National Insurance

The business must also pay towards the national insurance of each employee. This will be **13.8%** of each employee's daily pay (£72). For all 5 employees, the company will pay **£49.68** towards their national insurance each day.

Total On-cost wages

Adding the on-cost wages for each employee will result in a total of **£121.68** per day**.** For all 5 employees the total on-cost wages will be **£608.40** per day.

6.3. Day Rate

The day rate can be calculated using ((running costs per day / number of employees) + on-cost wages for each employee per day). The day rate for this project will be **£131.19**.

6.5. Costings Summary

Day Rate per Employee

The day rate per employee will be the total day rate divided by 5, which is **£26.24**.

Total Cost for Staff Time

The total cost for staff time will be **£36,504** for the duration of the project. ((total employee wages + total employer national insurance) * 60 working days).

Project-Specific Costs

- Microsoft Office Business: **£114** ((cost per month * 3) * number of employees))

- Unity Pro: **£290** (cost per month * 3)
- Adobe Creative Cloud Business: **£758.55** ((cost per month * 3) * number of employees))

Contingency Cost

The contingence cost will be **£2,951.69** (5% of the total project cost).

Total Cost for the Project

The total cost will be **£59,033.70** ((running costs per day + total on-cost wages for employees per day) * 90 days).

7. Subcomponent Specification – Requirement Specification

7.1. Requirements Specification of Gameplay Elements

	Requirement	Priority	Justification
2.1	Interaction and control for the player, allowing them to move around the environment. The player will be able to select between a mouse and keyboard or an Xbox One controller	High	Allows the player to interact with the level through their preferred peripheral, ultimately making the game more accessible
2.2	Zombies will spawn in incremental waves, which get tougher to fight as the game progresses	High	Gives the player something to do in the game and challenges them as the game progresses
2.3	Track the players state to determine different in-game events. (E.g., open different sections of the level as the player progresses, end the game when the player dies, etc.)	High	Allows the players actions to directly affect the game world (E.g., they make a bad move they die and the game ends)
2.4	Enemies should dynamically spawn nearby the player depending on where they are in the game world	Medium	Having enemies spawn nearby the player would help intensify the action of the game; however, if this is not feasible, enemies could be spawned into the world using traditional spawn points
2.5	Could have different game modes accessible from the main menu. (E.g., time-attack where the player must see how far they can progress in a set amount of time)	Low	The main game mode must be prioritised. If there is extra time at the end of development, this feature could be implemented.

Table 7.1. - Requirements Specification of the Gameplay Subsystem

7.2. Requirements Specification of HUD & UI Elements

No	Requirement	Priority	Justification
3.1	Accessible main menu that allows users to pause the game and do different actions inside of it	High	Main Menu is the most important aspect when it comes to the implementation of UI Elements and contains significant functionalities inside of it that allows the player to control, pause or even quit the game besides other possible implementations that are discussed below
3.2	The functionality to save the current state of the game to a local file and allow users to load previously passed checkpoints	High	Save/Load features have always been part of generations of games and it is a functionality that must be implemented inside the Main Menu
3.3	Visually dynamic character vital stats like health/stamina/ammo/etc.	High	Stats for the playable characters are important in order to determine whether or not is the case to switch to different characters or heal the ones that require aid. (in the case that the healing functionality is available) Ammunition supply is another important stat that helps players to easily visualise and determine quick moves in case they run out of it
3.4	Dynamic backgrounds for the main menu	Medium	A dynamic background would improve the overall appearance of the main menu and by analysing trending AAA titles and how they have implemented their main menus would provide enough background in order to design our own dynamic main menu
3.5	Visually dynamic enemy vital stats like health left/type of zombie/etc.	Low	As long as there is a possibility of having multiple types of zombies, a clear icon to represent their type would make sense and aid the player to distinguish zombies throughout their playtime. Vital stats like health will have a higher possibility of being implemented

Table 7.2. - Requirements Specification of the HUD & UI Subsystem

7.3. Requirements Specification of Character & AI Elements

	Requirement	Priority	Justification
4.1	Enemies must attack and deal damage to player and friendly AI characters	High	The aim of the game is to survive and without a threat the player simply has nothing to survive. Hence, enemies need to be able to kill the player and friendly AI characters.
4.2	Friendly AI characters must follow player around environment	High	If the friendly AI characters are near to the player, then they can assist the player in killing enemies. It also gives the player a sense that they are controlling a team of characters.
4.3	Friendly AI characters must attack and deal damage to enemies	High	The purpose of having friendly AI characters is to aid the player throughout the game, so they must be able to kill enemies. This essentially gives the player teammates in single player game.
4.4	Player should have the ability to switch control to friendly characters	Medium	Allowing the player to switch between multiple characters introduces more variety. The player can then come up with their own strategies (for example, switch to the shotgun character in close quarters areas of the environment).
4.5	Could have multiple enemy types	Low	Adding enemies with different behaviors will require the player to use different strategies and make different decisions throughout the game. Hence, providing more variety along with challenges.

Table 7.3. - Requirements Specification of the Characters & AI Subsystem

7.4. Requirements Specification of Weapons & Pick-ups Elements

	Requirement	Priority	Justification
5.1	The game must have an array of useable weapons, that are able to cause damage to opponents	High	A survival game would be incredibly difficult if there was nothing to kill the enemies.
5.2	The game must have a selection of power ups that the player can collect from the ground	High	Gives the play positive feedback for killing enemy's, increasing enjoyment of the product.
5.3	A leveling system for the weapons that increase the stats based on how much you have used it	Medium	Gives the player a sense of pride and accomplishment, as it gives the player something to show for all the time invested.
5.4	Pickups that alter the players current weapon	Medium	This will allow for more dynamism in the game, increasing the amount of replay ability.
5.5	Customizable weapon Attachments	Low	Makes the player’s character feel unique from others.

Table 7.4. - Requirements Specification of the Weapons & Pick-ups Subsystem

7.5. Requirements Specification of Skill Trees & Progression Elements

	Requirement	Priority	Justification
6.1	The balance of the skill tree	High	The survival game should have a certain amount of difficulty
6.2	Give different tier of skill in the skill tree	High	Allow the player to have more powerful stats to survive in the game
6.3	As the game time and difficulty increase, the player's level will also increase. Each time player level up will provide some skill points.	High	The skill point will allow the player to upgrade skill in the skill tree. It will be balance not letting the character have level up too fast or having too much skill point to upgrade skill. So, game will be too easy.
6.4	Each different character has a unique set of skill tree	Medium	Make character totally unique to each other
6.5	Each character has a Unique skill	Low	Make each character have unique skill to player with

Table 7.5. - Requirements Specification of the Skill Trees & Progression Subsystem

References

- 343 Industries (2015) *Halo: Spartan Strike* [Video game]. Microsoft Studios.
- Activision (2010) *Call of Duty: Black Ops* [Video game]. Activision.
- Adobe CC Business (2019) *Creative Cloud Packages*. Available at: https://www.adobe.com/uk/creativecloud/plans.html?promoid=NV3KR7S1&mv=other
- Agile Alliance (no date) *What is Extreme Programming (XP)?*. Available at: https://www.agilealliance.org/glossary/xp/
- Bungie (2009) *Halo 3: ODST* [Video game]. Microsoft Studios.
- Crystal Dynamics (2014) *Lara Croft and the Temple of Osiris* [Video game]. Square Enix.
- Cullen, M. (no date) *Basics of Sound Design for Video Games*.
- Digital Risks (2019) Available at: https://www.digitalrisks.co.uk/
- Ekman, I. (2005) *Meaningful Noise: Understanding Sound Effects in Computer Games*.
- Epic Games (2011) *Gears of War 3* [Video game]. Microsoft Studios.
- Feil, J. Scattergood, M. (2005) *Beginning game level design*. Thomson Course Technology
- Game Analytics (2017) *9 Sound Design Tips to Improve your Game's Audio*. Available at: https://gameanalytics.com/blog/9-sound-design-tips-to-improve-your-games-audio.html
- GitHub for Unity (2019) Available at: https://unity.github.com/
- Godey, A. Barbosa, E.F. (2010) *Game-Scrum: An Approach to Agile Game Development*.
- Heads Up Display | Metro 2033 (N.D.) Available at: https://metrovideogame.fandom.com/wiki/Heads-Up_Display (Accessed: 24 February 2019).
- IGN (2019) *PEGI Rating System*. Available at: https://uk.ign.com/wikis/content-ratings/PEGI
- Infinity Ward, Treyarch, & Sledgehammer Games (2018) *Call of Duty* [Video game series]. Activision.
- Living Wage Foundation (2019) *The Calculation*. Available at: https://www.livingwage.org.uk/calculation
- Lucid Games (2014) *Geometry Wars 3: Dimensions* [Video game]. Activision.
- Microsoft (2018) *Xbox One Adaptive Controller* [hardware].
- Microsoft Developers Blog (2014) *Adding Xbox controller support to your Unity 3D game*. Available at: https://blogs.msdn.microsoft.com/uk_faculty_connection/2014/12/02/adding-xbox-controller-support-and-input-to-your-unity3d-game/
- Microsoft Office Business (2019) Available at: https://www.microsoft.com/en-us/microsoft-365/business Gov.uk (2019) *National* Insurance *rates*. Available at: https://www.gov.uk/national-insurance-rates-letters
- MMO Games (2015) *Dead Island: Epidemic Preview*. Available at: https://www.mmogames.com/gamereviews/dead-island-epidemic-preview/ (Accessed: 24 February 2019).
- Rare (1997) *GoldenEye 007* [Video game]. Nintendo.
- Sega (1991) *Sonic the Hedgehog* [Video game].
- Square Enix (2002) Conflict: Desert Storm [Video game]. Square Enix.
- Studica (2015) *How to Setup GitHub with Unity: Step-by-Step Instructions*. Available at: https://www.studica.com/blog/how-to-setup-github-with-unity-step-by-step-instructions
- Unity Manual (2018) *Inner Workings of the Navigation System*. Available at: https://docs.unity3d.com/Manual/nav-InnerWorkings.html
- Unity Manual (2018) *Publishing Builds*. Available at: https://docs.unity3d.com/Manual/PublishingBuilds.html
- Unity Pro (2019) Available at: https://unity3d.com/
- Valve Corporation (2009) *Left 4 Dead 2* [Video game]. Valve Corporation.
- Venturelli, M. (2015) *Everything I Learned About Duel-Stick Shooter Controls*. Available at: http://www.gamasutra.com/blogs/MarkVenturelli/20150817/251387/Everything_I_Learned_About_DualStick_Shooter_Controls.php
- Zagal, P. Fernádez-Vara, C. Mateas, M. (2008) *Rounds, Levels, and Waves: The Early Evolution of Gameplay Segmentation*.

Appendices

A1 Code of Conduct

Code of Conduct

Midnight Rising

Module: KV6002 - Team Project & Professionalism

Northumbria University Newcastle

January 31, 2019

Revised: February 7, 2019

1. **Working Together**

1.1. All group members will be respectful of one another's beliefs and cultural backgrounds.

1.2. There will be no discrimination of sex, sexual orientation, nationality, colour, race, ethnic origin, religion, age, disability, or and other factor not specified in this document.

1.3. There will be no belittling the ideas and opinions of other group members. Every member should be able to feel included in the group.

1.4. All group members will be treated equally and with respect.

1.5. Do not interrupt other group members whilst they are speaking.

1.6. There will be no personal attacks on group members. Instead, provide constructive criticism on ideas if necessary.

1.7. If any conflicts arise between group members,they will be dealt with by talking it out with the rest of the team until a consensus is reached. If the issue persists then the project supervisor will be consulted to help resolve the matter.

1.8. There will be no collaboration between group members on subsystems as this is considered cheating. Team members should still support each other however, especially if other team members are struggling with the workload.

1.9. To maintain a high standard of integrity, good judgement will be used to deal with any ethical issues that arise which are not detailed in this document.

2. **Meetings & Communication**

2.1. Meetings will take place every Thursday from 2pm to 4pm, with potentially at least one other meeting each week. It is expected that all members will attend these meetings.

2.2. Preferably, the group will meet in the library study rooms. If there are no rooms available then the group will meet in the CIS Games Lab (102) instead.

2.3. Any group work must be completed during meetings.

2.4. All individual work must be completed outside of group meetings. If any group member is struggling then they must communicate this with rest of the group.

2.5. Facebook Messenger and email have been implemented for communication outside of group meetings, allowing for 24/7 communication between group members. This will be used if any issues arise that require a spontaneous meeting.

2.6. If a member of the group is unable to attend a meeting for any particular reason, they must ensure other members of the group are aware of this before the meeting takes place.

3. **Documentation & Code**

3.1. All documentation and code will be submitted to a private repository on GitHub. Group members are expected to use this repository when working on the project.

3.2. All documentation detailing the actions of specific team members will require that team member to sign-off that document via print & signature. E.g., task allocations in project supervisor meetings.

3.3. Any issues/bugs discovered in the code will be reported to GitHub's 'issues' system. The group member responsible for fixing the issue must be assigned to the it.

3.4. All group members will be expected to follow the agreed upon project style guide.

3.5. All group members will be expected to complete their assigned subsystem(s) in time for any project deadlines. If any group members are falling behind on the project, the project supervisor will be consulted to help resolve the issue.

4. **Task Allocation**

4.1. As a group, members will work on: game design; testing; sound effects and music; environment and level design; as well as all project management.

4.2. Andrew Alford will work on the Gameplay Programming aspects of the project. This will be expanded upon in the Terms of Reference document.

4.3. Alex Trench will work on the Weapons & pick-ups in the project. This will be expanded upon in the Terms of Reference document.

4.4. Carl Pendleton will work on the Characters & AI in the project. This will be expanded upon in the Terms of Reference document.

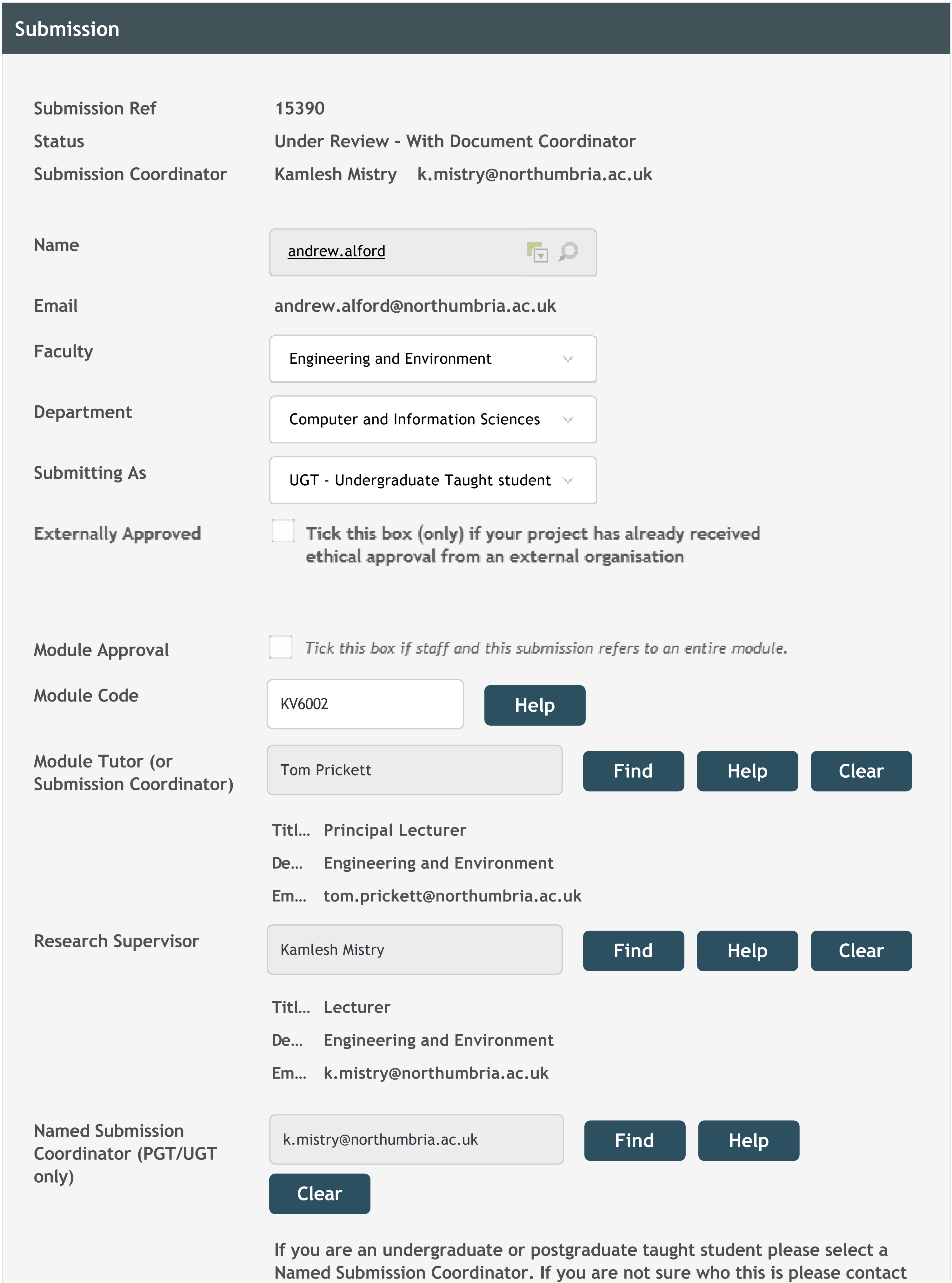
4.5. Alexandru-Daniel Pascal will work on the HUD & UI aspects of the project. This will be expanded upon in the Terms of Reference document.

4.6. Haoming Yuan will work on the Skill Tree & progression aspects of the project. This will be expanded upon in the Terms of Reference document.

Code of Conduct	Date & Time:
Student	Signature
Andrew Alford	AA
Alexandru-Daniel Pascal	
Carl Pendleton	C. Pendleton
Alex Trench	A. Trench
Haoming Yuan	袁浩铭

My Documents

A2 Ethical Approval



Submission

Submission Ref 15390

Status Under Review - With Document Coordinator

Submission Coordinator Kamlesh Mistry k.mistry@northumbria.ac.uk

Name andrew.alford

Email andrew.alford@northumbria.ac.uk

Faculty Engineering and Environment

Department Computer and Information Sciences

Submitting As UGT - Undergraduate Taught student

Externally Approved □ Tick this box (only) if your project has already received ethical approval from an external organisation

Module Approval □ *Tick this box if staff and this submission refers to an entire module.*

Module Code KV6002

Module Tutor (or Submission Coordinator) Tom Prickett

Titl... Principal Lecturer

De... Engineering and Environment

Em... tom.prickett@northumbria.ac.uk

Research Supervisor Kamlesh Mistry

Titl... Lecturer

De... Engineering and Environment

Em... k.mistry@northumbria.ac.uk

Named Submission Coordinator (PGT/UGT only) k.mistry@northumbria.ac.uk

If you are an undergraduate or postgraduate taught student please select a Named Submission Coordinator. If you are not sure who this is please contact your Module tutor or Supervisor as appropriate.

Ethical Risk Level	Low

Risk Level Conditions:

Your ethical risk is **low**. Your research should only consist of one or more of the following:
- Analysis of secondary data which has been previously published.
- Desk or lab-based research which does not involve collecting data from people (other than pilot data collected solely within the research team).

Your project proposal does not need to be reviewed by your Faculty Research Ethics Committee, however, you need to be ethically aware and ensure that you have not breached plagiarism or copyright regulations and have adequately referenced your material. It is recommended that you refer to Northumbria Research Ethics Policy.

Co-investigators

Add Edit Delete Save Refresh

NAME OF CO-INVESTIGATORS
Alex Trench
Alexandru-Daniel Pascal
Carl Pendleton
Haoming Yuan

G1: General Aims and Research Design (Mandatory)

Title

Title of your research project

Team Project & Professionalism - Top-down Zombies Game

Outline General Aims and Research Objectives

State your research aims/questions (maximum 500 words). This should provide the theoretical context within which the work is placed, and should include an evidence-based background, justification for the research, clearly stated hypotheses (if appropriate) and creative enquiry.

To develop a top-down wave-based zombie survival game using Unity Engine. This project will consolidate all the games programming experience we have received on this course as well as develop our skills in C# and Unity.

G2: Research Activities (Mandatory)

Please give a detailed description of your research activities

Please provide a description of the study design, methodology (e.g. quantitative, qualitative, practice based), the sampling strategy, methods of data collection (e.g. survey, interview, experiment, observation, participatory), and analysis. Do sensitive topics such as trauma, bereavement, drug use, child abuse, pornography, extremism or radicalisation inform the research? If so have these been fully addressed?

Objectives:

- Analysis

1. Create a Terms of Reference document to outline the scope, aims, and objectives of the project
2. Project planning - Gantt charts, Github projects/task allocation, group meetings
3. Research on making 3D top-down games in Unity will be conducted
   We will also need to get a good understanding on C# to begin synthesis

- Synthesis

4. Design the game
5. Implement the game
   This will be split into 5 different subsystems (1 per group member)
   5.1. Gameplay Programming
   5.2. Characters & AI
   5.3. Weapons & Pick-ups
   5.4. HUD & UI
   5.5. Skill Tree & Progression
6. Test the game (Done throughout synthesis)

- Evaluation & Conclusion

7. Each group member will write out their own evaluation report summarising the outcomes of the project

G3: Research Data Management Plan (Mandatory)

Anonymising Data (mandatory)

Describe the arrangements for anonymising data and if not appropriate explain why this is and how it is covered in the informed consent obtained.

N/A - There will be no user testing for this project

Storage Details (mandatory)

Describe the arrangements for the secure transport and storage of data collected and used during the study. You should explain what kind of storage you intend to use, e.g. cloud-based, portable hard drive, USB stick, and the protocols in place to keep the data secure.

If you have identified the requirement to collect 'Special category data', please specify any additional security arrangements you will use to keep this data secure.

N/A - There will be no user testing for this project

Retention and Disposal (mandatory)

☑ I confirm that I will comply with the University's data retention schedule and guidance.

Research Data Management link

Data Protection link

Records Retention Schedule link

G4: Research Project Timescale (Mandatory)

Proposed Start Date 14/02/2019

Proposed End Date 16/05/2019

G5: Additional Information

□ Externally Funded

External Funder

Please give details of your 'other' funder

Agresso Reference

□ Franchise Programme Organisation

Please give details of your franchise organisation

Type a value

□ NHS Involvement

Please give details of any NHS involvement

Type a value

□ Clinical Trial(s)

Please give details of any Clinical Trial(s)

Type a value

□ Medicinal Products

Please give details of any Medicinal Product(s)

G6: File Attachments

Additional files can be uploaded e.g. consent documentation, participant information sheet, etc.

Please note: It is best practice to combine all documents into one PDF (This avoids the reviewer having to op...

Go To Attachments

G7: Health and Safety (Mandatory)

☑ I confirm that I have read and understood the University's Health and Safety Policy.

☑ I confirm that I have read and understood the University's requirements for the mandatory completion of risk assessments in advance of any activity involving potential physical risk.

Please tick one of the boxes below...

□ There are PHYSICAL risks associated with the work and I have consulted the following approved risk assessments...

State Risk Assessment references and titles

Specific risk assessments, where required, have been produced, approved and submitted to the Risk Asse...

I will take the necessary action, adhere to any identified control measures, and consult with the central Health and Safety Team where necessary to manage the risks.

☑ I can confirm that there are no physical risks associated with this project and so no risk assessments are required.

G9: Electronic Signature (Mandatory)

☑ I confirm my supervisor has reviewed the contents of this document

☑ I confirm I have assessed the ethical risk level of my work correctly and answered the above sections as fully and accurately as possible.

Full Name andrew.alford

Date 14 February 2019 15:38:10

PDF Version

Create PDF

No items to display.

Review Comments, Conditions and Outcomes

Log of any Ethical Incidents

Log New Incident

INCIDEN...	CREATED DATE TIME	CREATOR NAME	COMPLAINANT DETAILS

No items to display.

Title and Objectives (see G1)

Add Save

Reviewer A: Reviewer B:

e.g. Are the research question and/or study aims clear?

DATE	ROLE	COMMENT

No items to display.

Proposed Methodology and Analysis (see G2)

Add Save

Reviewer A: Reviewer B:

e.g. Is the design appropriate to the research question?
Are the methods of data analysis appropriate to the research question?

DATE	ROLE	COMMENT

No items to display.

Sample and Recruitment (see M1)

Add Save

Reviewer A: Reviewer B:

e.g. Is the sampling approach appropriate to the design?
Is the sample sufficient and achievable?
Is the process of recruitment clearly explained?

Are participants receiving payments for taking part, and if so is the payment appropriate?
If the DBS is ticked, has the appropriate information been included?

DATE	ROLE	COMMENT

No items to display.

Consent (see M1)

Add Save

Reviewer A: Reviewer B:

e.g. Is the approach to consent seeking clear?
Is consent from parents/ carers/ guardians required?
Are all necessary recruitment and informed consent documentation included (e.g. letters of permission, letters of invitation)
Is the information sheet adequate to ensure informed consent?
Are the consent form(s) appropriate?

DATE	ROLE	COMMENT

No items to display.

Researcher and Participant Safety (see M1)

Add Save

Reviewer A: Reviewer B:

e.g. Is there any risk of physical harm for the researcher(s) or the participants and if so what attempts have been made to alleviate or minimise them?
Have Risk Assessments been referred to where appropriate?

DATE	ROLE	COMMENT

No items to display.

Research Activities (see G2-G8, M1-M5, H1-H5)

Add Save

Reviewer A: Reviewer B:

e.g. Are the research tasks described clearly?
Do sensitive topics such as trauma, bereavement, drug use, child abuse, pornography or extremism/ radicalism inform the research? If so have these been fully addressed? (and we can use this to amend the information on risk levels on the form)Is there any risk that the tasks may cause psychological harm and if so what attempts have been made to alleviate or minimise them?

DATE	ROLE	COMMENT

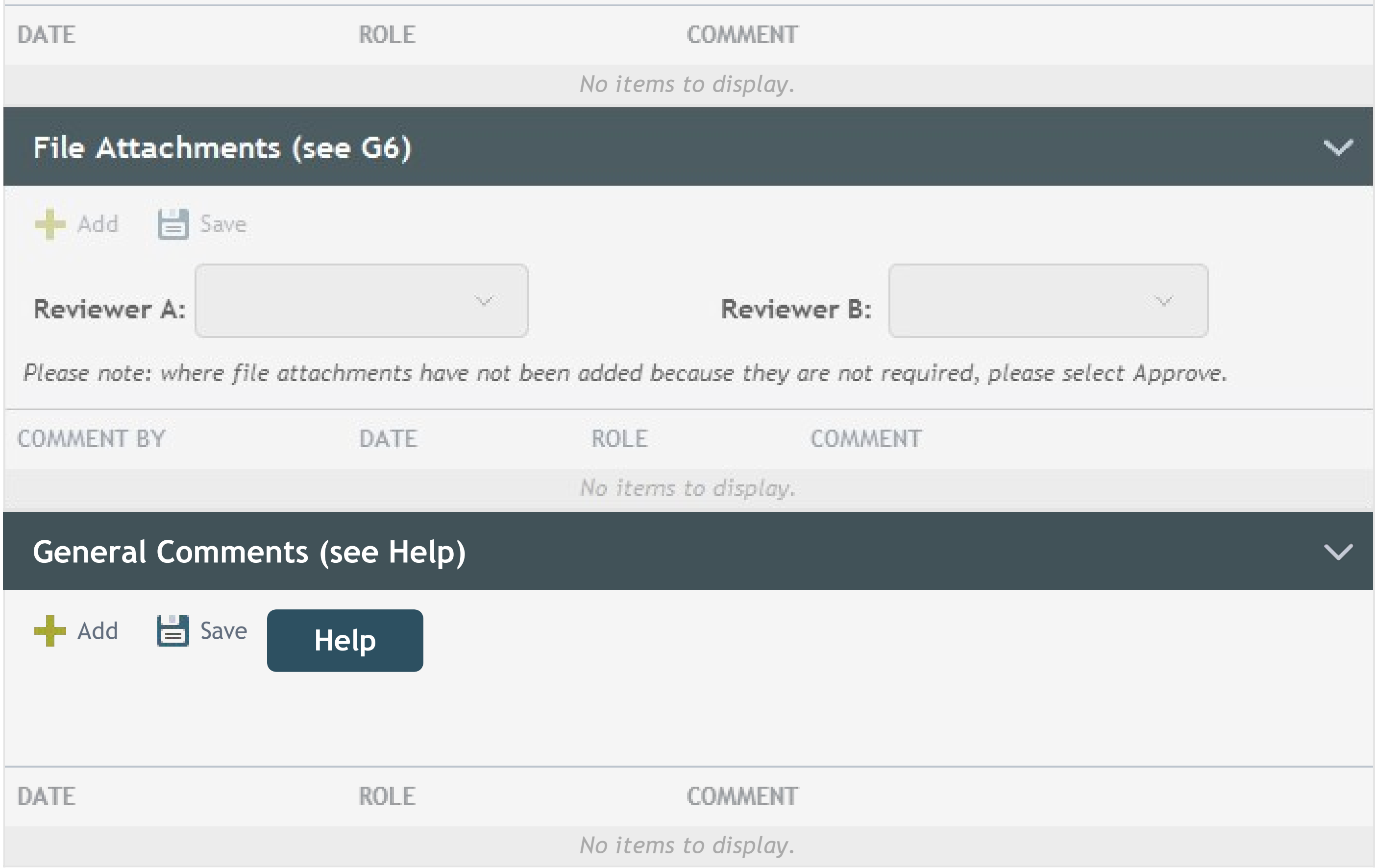
No items to display.

Data Management Plan (see G3)

Add Save

Reviewer A: Reviewer B:

e.g. Have sufficient steps been taken to ensure participant anonymity/confidentiality of data?
Are the arrangements for data storage and disposal clearly outlined?
Are these arrangements in line with University and/or the funding body requirements?



DATE	ROLE	COMMENT
No items to display.		

File Attachments (see G6)

Reviewer A:

Reviewer B:

Please note: where file attachments have not been added because they are not required, please select Approve.

COMMENT BY	DATE	ROLE	COMMENT
No items to display.			

General Comments (see Help)

DATE	ROLE	COMMENT
No items to display.		

A3 Risk Assessment Form

Risk types – F (Financial), T (Technology), P (People), E (Environmental), S (Security)

No.	Risk Description	Risk Type	Likelihood (1-10)	Impact (1-10)	Risk Value (1-100)	Risk Monitoring/Control Flag	Risk Management Strategy	Risk Owner
1	Team members stop showing up to meetings	P	5	2	10	Ensure members have read the Code of Conduct document and that all members can attend on proposed date and time	Group meetings are held at times that are suitable for all team members, increasing the likelihood of full attendance	Everyone
2	Team member fails to implement their subsystem	P	3	8	24	Weekly team meeting in which we discuss what work we have done the previous week	Pep talks during team meetings and support if a member is struggling	Team member who fails to implement, but possibly everyone if other subsystems depend on the subsystem not implemented
3	Work being lost or accidental deletion	P, T	7	4	28	Make frequent backups	Revert to a previous version of the project	Everyone
4	Team member has accident and cannot continue work	P	1	10	10	N/A	Accidents cannot be avoided; however, the individual tasks have been structured so that there less reliance in one part of the system to make another function	Everyone
5	Hardware system fails	T	1	7	7	N/A	If equipment fails (computer), use other machines that can run required software	Everyone
6	Inadequate requirement gathering	P	2	7	40	Referencing back to the Terms of Reference document to make sure team member are not	Conduct a large and tougher requirements analysis and stick to it	Everyone

						adding large amounts of functionality not discussed or evaluated as part of the requirement gathering		
7	Team member overrides GitHub files of another team member	P	4	7	28	Use GitHub branching functionality so members can't modify another member's work	Rollback GitHub to the last previous version to restore files	Everyone
8	GitHub servers being unavailable	T	1	9	9	N/A	Revert to another cloud-based storage system or version control	Everyone
9	Files corrupting	T	1	8	8	Make frequent backups	Revert to a previous version	Everyone
10	Failure to meet deadlines	P	3	10	30	Set a deadline earlier than the actual deadline	N/A	Everyone
11	Failure to combine all subsystems	P	3	8	24	Make sure all subsystems are integrated early so they can be tested properly	If there is an unsuccessful integration of a subsystem that effectively breaks the game, then restore to an unbroken version	Everyone
12	Failure to extensively test final product	P	4	6	24	Use the proposed testing strategy and allow enough time for extensive product testing before product demonstration	Try to follow the testing strategy to its fullest, and carry out testing incrementally during development	Everyone

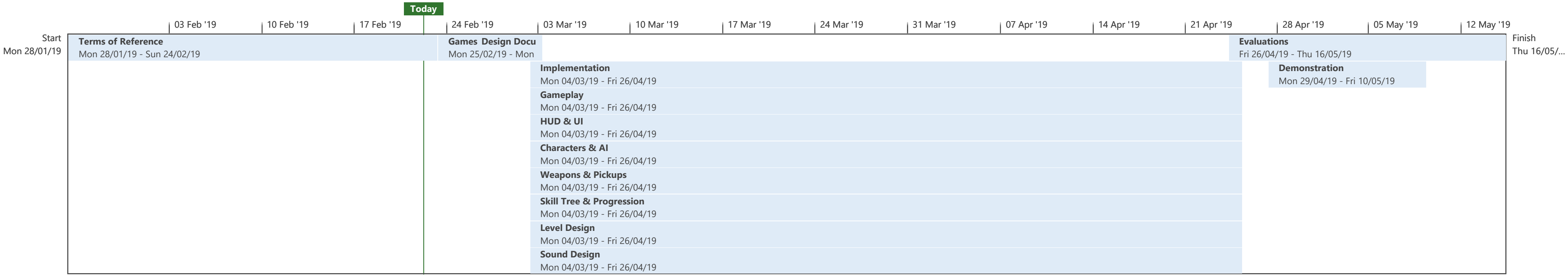
>= 75 ***Risk very high -*** *ur*gent action required

>= 50 < 75 ***Risk high*** - action as soon as possible

>= 25 < 50 ***Risk may be acceptable*** - more analysis required

< 25 ***Low risk*** - no gains expected from extra work

A4 Gantt Chart


Today
03 Feb '19
10 Feb '19
17 Feb '19
24 Feb '19
03 Mar '19
10 Mar '19
17 Mar '19
24 Mar '19
31 Mar '19
07 Apr '19
14 Apr '19
21 Apr '19
28 Apr '19
05 May '19
12 May '19
Start
Mon 28/01/19
Terms of Reference
Mon 28/01/19 - Sun 24/02/19
Games Design Docu
Mon 25/02/19 - Mon
Evaluations
Fri 26/04/19 - Thu 16/05/19
Finish
Thu 16/05/...
Implementation
Mon 04/03/19 - Fri 26/04/19
Demonstration
Mon 29/04/19 - Fri 10/05/19
Gameplay
Mon 04/03/19 - Fri 26/04/19
HUD & UI
Mon 04/03/19 - Fri 26/04/19
Characters & AI
Mon 04/03/19 - Fri 26/04/19
Weapons & Pickups
Mon 04/03/19 - Fri 26/04/19
Skill Tree & Progression
Mon 04/03/19 - Fri 26/04/19
Level Design
Mon 04/03/19 - Fri 26/04/19
Sound Design
Mon 04/03/19 - Fri 26/04/19