**Software Implementation and Testing Document**

**For**

**Group <32>**

Version 1.0

**Authors**:

Merrick M

Derek T

Dillon M

Isaac M

Jacob C

# Programming Languages (5 points)

We only use C# because that is what is supported in the Unity Engine.

# Platforms, APIs, Databases, and other technologies used (5 points)

We use Unity Engine for execution and visualization of our game. Github to store all of our files and visual studio code to write our code.

# Execution-based Functional Testing (10 points)

We tested the function requirements listed in our RD on Unity Engine. We also have tested all the features in our key progress highlights in our email to our stakeholders. Such as found below:

Zombie Mechanics: We have successfully implemented core zombie mechanics. These include movement, detection, health, damage intake, damage output, spawn patterns and attack patterns.

Player Mechanics: We have successfully implemented movement and interaction systems. This includes walking, camera, health system, shooting, two test maps and aiming, providing a smooth and responsive experience for users.

Gun Mechanics: We have developed a functional weapon system. Players can now equip a weapon and shoot bullets. We are working on multiple weapons, making sure the sound is in synch with shots as well as collision mechanics for obstacles where zombies spawn from.

Testing Map: Uploaded new testers and sprites and allow us to test out newly developed features. We have one map which does not work, and another map which works but has a few bugs when testing code. Our next step is making the obstacles on maps spawn points and so the player cannot walk or shoot through them.

Main Menu (Work in Progress): We've begun designing the main menu for the game, though it is still in its early stages. It currently links to another option screen and allows players to start the game.

# Execution-based Non-Functional Testing (10 points)

We tested non-functional requirements by playing the game on Unity Engine and seeing if it is enjoyable.

# Non-Execution-based Testing (10 points)

1. We performed non-execution based testing by have sufficient comments on our C# scripts, weekly meetings where we walked through the new features/code implemented into our game, and comments on newly updated code.