# Component 2: Project Retrospective Report

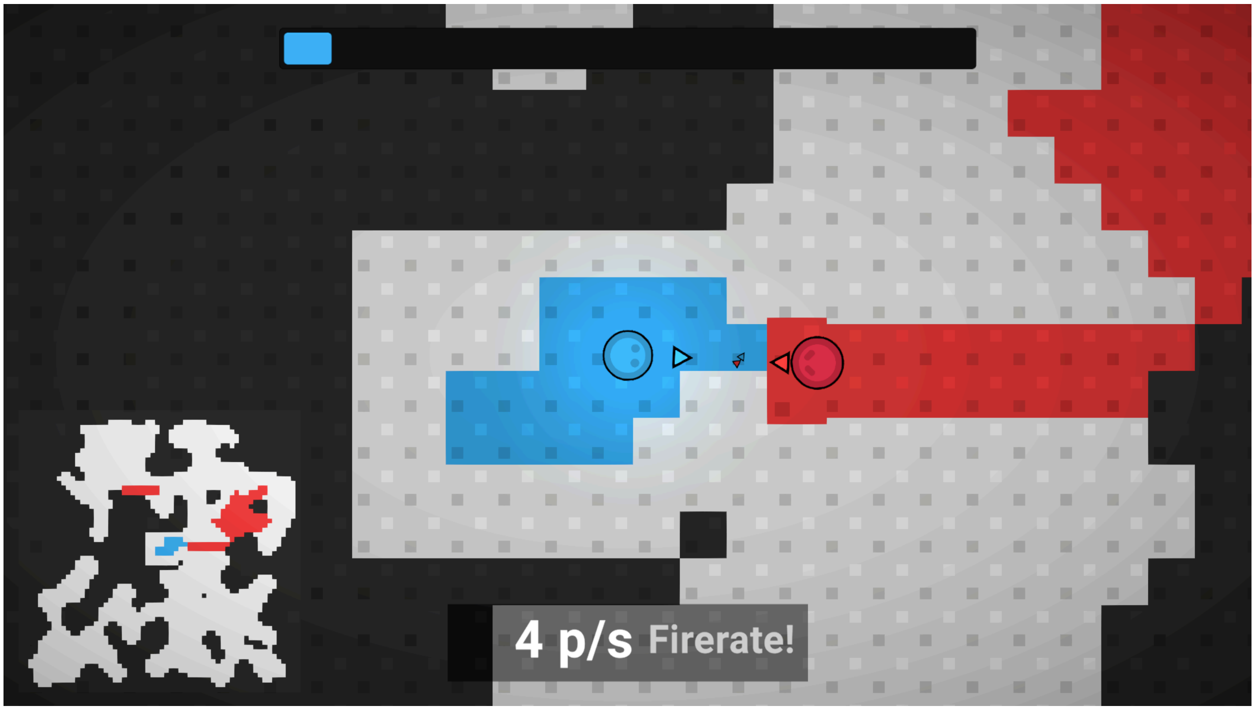
**Student Name:** Oliver Neal

**Student Number:** [redacted]

**Module:** GAD3002-N Advanced Development Techniques

**Module Leader:** [redacted]

**Word Count:** 916 / 1000



# Table of contents

# 1.0 Introduction

This project is a top-down shooter prototype investigating the ‘Territory-Linked Offensive Scaling’ to solve the problem of defensive camping. This document will analyse the iteration process during development, linking decisions to initial research on Splatoon and DOOM. This document will also reflect on the professional skills gained.

As a retrospective, this document will cover the development of this project by overviewing and reflecting upon the decisions made and the techniques used.

The project sought to prove that linking fire rate to territory control would force players to abandon defensive camping strategies.

# 2.0 Iterative Design and Development

## 2.1 Process Breakdown

Starting this project, I planned out the core gameplay loop and typed out a simple gameplay idea and loop. From this, I soon began a rapid prototyping process in the Unity engine using both the engine and C# code.

Before beginning the prototyping process, I set up a GitHub to establish version control for During prototyping, I initially set up the player controller as well as the tile-painting mechanic where the player paints tiles that they move over. From this point, I developed a rudimentary enemy AI to inch closer to a complete yet basic gameplay loop.

## 2.2 Iteration Cycle 1: The Core Mechanic

The core mechanic of painting for ammunition stemmed from initial research into Splatoon. Although the link isn’t explicit, I formed a connection between paint and ammunition which led to the idea.

During the early prototyping period, it was revealed that just moving around the floor to paint felt too unengaging due to the passive nature of the mechanic. This is because during playtesting, some testers noted that they felt ‘passive’ merely walking over tiles, indicating a sense of boredom with the dull gameplay experience. Therefore, I decided to iterate the design to make the painting trigger (inspired by Qix). This aligned better with the ‘Push Forward’ combat philosophy identified in DOOM.

## 2.3 Iteration Cycle 2: Enemy Feedback (The "Inflation" Mechanic)

When it comes to UI, standard health bars would clutter the space and distract from the territory mechanic. The solution I conjured was to develop an ‘Inflation’ system where enemies swell and gradually change colour (‘enemy red’ towards ‘player blue’) before bursting. This decision was driven by the need for diegetic feedback, ensuring the player’s eyes remained focused on the gameplay space. This decision was made through UI research, specifically on the different types of UI and causing me to realise this feedback idea (Corporation Pop, 2024).

### 2.4 Technical Implementation Challenges

A specific coding hurdle during the development process was the finite state machine for the enemy. Developing a finite state machine that both functions well and presents well in-game is a challenging task that I struggled with. I ended up initially developing a behaviour model that was noticed as ‘buggy’ and ‘unrealistic’ by playtesters.

Solving this, I decided to design the simplest decent finite state machine model. I had previously designed a finite state machine that I now deem was too complicated, and reducing this to an efficient but simple model ended up being part of the resolution. I also researched AI behaviours, learning fundamentally about how state machines work, which is where I learned about behaviour trees and ‘fuzzy logic’. Using this fundamental knowledge with the simpler efficient finite state machine model, I was able to iterate on the finite state machine where playtesters saw no issues with the AI behaviour.

## 3.0 Demonstration of Transferable Skills

### 3.1 Skill 1: Problem Solving (Technical)

A situation I was in was that enemies had a high likelihood of spawning within close proximity to the player, allowing an opportunity where the player could quickly eliminate the enemies and finish the game in an unreasonably brief amount of time.

After analysing the issue, I decided that increasing the map size and also implementing a procedural map would allow enough variability and size to extinguish the previous small and open map problem.

This improved several gameplay elements that rooted from this issue, demonstrating my ability to implement level design solutions to entity interaction problems.

### 3.2 Skill 2: Adaptability / Time Management

In the early phase of development, unique enemies were intended to be part of the gameplay experience to help present the advanced development technique. These enemies would have varying roles, challenging the player with a variety of elements to provide the best possible game space for the player.

However, due to slight overscoping in the project, I had to make the decision to adapt this feature by only designing one enemy. To prevent underscoping the enemies feature, I chose to create a single enemy with higher complication compared to the original intent. This was achieved without re-scoping to previous intentions.

This allowed me to de-scope a core element of the game without detrimenting the overall gameplay experience, demonstrating efficient adaptability when it comes to time management.

### 3.3 Skill 3: Resilience

In the later stages of development, I abruptly faced an issue where the minimap was causing significant performance issues. This was due to my minimap design including the use of a 2nd in-engine camera to act as a map-wide view. This issue was challenging to solve, as I repeatedly iterated with little/no success.

After an extended period of time, I eventually iterated the minimap feature by utilising the Unity tilemap system. I used a simple tilemap to represent the colours of the tiles in the map, presenting a high-performance minimap for the player.

This demonstrated my resilience within the developing of the project, shown through my time spent iterating and pushing ideas until it is resolved.

## 4.0 Insight and Future Development

While the territory-linked fire rate solved camping, it created a 'snowball effect' where the winning player became too powerful too quickly. Future development would require a 'catch-up mechanic' (like the Blue Shell in Mario Kart) to balance the territory advantage.

## 5.0 Bibliography

Corporation Pop. (2024). Corporation Pop. [online] Available at: https://corporationpop.co.uk/thoughts/game-ui-design.