University of Reading

Department of Computer Science

Virus Attack

Multiplayer Networked Game

Riya Mayor

Supervisor: Dr Carmen Lam

A report submitted in partial fulfilment of the requirements of
the University of Reading for the degree of
Bachelor of Science in *Computer Science*

22nd of April 2022

Declaration

I, Riya Mayor, of the Department of Computer Science, University of Reading, confirm that this is my own work and figures, tables, equations, code snippets, artworks and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced. I understand that if failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly.

Riya Mayor
14th of November 2021

Acknowledgements

Firstly, I would like to express my gratitude to my supervisor, Dr Carmen Lam, for being supportive and guiding me through the whole project.

I would also like to thank Gary Edwards for his guidance and motivation and for giving me strategies on how to progress with this project, assignments, and online lectures within the given timelines.

And finally, a massive thanks to my family for being by my side and providing me with valuable feedback and testing my game throughout the whole development cycle.

Abstract

The goal of this project was to create an online multiplayer networked 2D shooting video game, with good graphics and challenges that were developed using freely available software and tools which are also easy to install with quick download times. 2D games are popular in the gaming industry, and the implementation for this project provides a fun but challenging product.

This report focuses on the design and development of a top-down view of a 2D game, Virus Attack. The game is an online multiplayer 'shooting' game, and the objective is to eradicate all the viruses that are spawned from labs that are randomly generated during game play. The game progresses in level when either all the viruses have been eliminated from the game scene and the player completes the level or if the viruses defeat the players by reducing their health to zero.

The game is inspired by the Covid 19 pandemic that hit the world 2 years ago and the desire for mankind to eradicate the virus so that life can get back to 'normal'. The game starts with the idea that a player has an antidote which can be fired in the form of bullets, and the goal is to kill the virus whilst avoiding being infected by it. The initial functionality is to allow up to 4 players to join a match and play together over the network.

Contents

1. Introduction

The aim of this report is to show the whole concept along with the development processes and the analysis and design involved in making 'Virus Attack'. The game was developed in Unity 3D Game Engine using C# with Photon Unity Networking (PUN), and this version is designed to be run on a PC platform.

I have always enjoyed playing games on different platforms and this gave me the motivation to undertake designing and developing Virus Attack as my final year project. I enjoyed the challenge of learning and implementing the different tools and software required for development of this game.

When the world was struck by Covid 19, the gaming industry had a surge of interest as people were 'trapped' in their homes due to lockdown. I came up with the topic for my game as it is topical for this time. An objective I have is that I would like to progress my career in the games industry and by undertaking this project I will have gained a good understanding of how to develop games. This project is also something I can use to demonstrate my knowledge and skills when applying for jobs.

This game can be further developed, to include different maps, choice of avatars, more gaming options, starting the game at various levels to keep the player challenged and the ability to port the game to mobile platforms, and ideas and designs for future implementation have been provided in a separate section of this report.

1.1 Background

The gaming industry is one of the fastest growing industries bringing gamers across the world to play independently and with each other on different platforms with the main five being: Mobile, Microsoft Xbox, Sony PlayStation, Nintendo Switch and PCs. The network of super computers and technology advancements has made this possible and the fact there are freely available software choices which enables games to be developed in minimum time. There are various genres of games like action, adventure, shooter, racing, fighting, puzzles, and educational games that are available to suit every person whatever their age.

This project provides a platform to present an online, multiplayer networked 2D shooting game. Games of this type which have multiple players logging in and playing cooperatively are exceedingly rare in the market, and this project will provide a unique offering with these capabilities. The subject of the game is also very topical, with the current pandemic which is affecting the whole world being the source of inspiration for the game play strategy and presentation. The game play provides a vehicle where up to 4 players can sign in and play together, with the task of seeking and destroying virus spores which are being generated from testing laboratories. The virus spores will affect the players if they come into contact with the player avatar, and different mutations of the virus spore will reduce the player health with different intensity. The players have various tools available to them to help them in their fight against the virus spores, these tools give health boosts to increase the player health, and ammo boosts to provide the player with high capability antivirus projectiles. Other tools have also been identified with suggested capabilities to be implemented in future enhancements to this project, if so desired.

The Unity3D game engine has the ability to provide a powerful tool to create a multiplayer game. It has a fast engine and requires very little coding to see results. It also comes with a free version which gives developers the ability to create games in 2D or 3D. The engine offers a primary scripting Application Programming Interface (API) in C#. There is some support available from a thriving Unity community, with tutorials generated by other developers which highlight the methods undertaken by them to develop their specific projects. I decided to build my game on a PC as it gives the option of reaching a wider online audience and gives me the option to publish my game to the popular PC online gaming market stream.

In this project I used Photon Unity Networking (PUN) which is an asset that can be downloaded in Unity that enables players to play with each other across the network. The game can be played using the ‘WASD’ keys or arrow keys on the computer keyboard to move the player up, left, down and right respectively and use of the 'Z' key to move the player in conjunction with the mouse clicks to shoot the viruses.

The project report is about the development of a casual online 2D multiplayer game. Casual gaming has become popular as it is aimed at a wider demographic. Casual games are simple games which are intuitive and are quick to play unlike hardcore games that require high end graphics and high specification devices to execute them. They have become the favoured option for many players as they can be played effectively on handheld devices which only require internet connectivity to play with other players online. Casual games are the most heavily downloaded genre of games and quite a few of them are free to download.

A couple of articles below show the concerns of gaming during the pandemic and the effect on children who spent a lot of time playing online games.

In a gaming report in the unity website, (Gaming Report, n.d.) https://create.unity.com/gaming-report-2022 there was some concern among developers that after Covid-19, once the world returned to normal, gamers would leave games in favour of other entertainment. But the prediction is that gamers have formed new habits which will lead to growth in the future. There is room for more games to be developed.

According to the article in Cybercrew (cybercrew, n.d.), 93% of children have been actively involved in playing video games during the lockdown, which does not sound good, but gaming can cause many positive effects on children as it affects their tactile, audio, and visual senses. Fighting games specifically can develop muscle memory and reaction time.

The gaming industry is at its peak. According to a couple of articles on the Unity website, (Unity.com, n.d.): "The gaming industry is booming, and more games are being published in every category. 18 months after the peak of the pandemic, gaming is still up more than 50%. Every month, there are 5 billion downloads of apps built in Unity. There are now 2.8 billion monthly active end-users who are engaging with content created or operated by Unity solutions in 2020 (Unity Website, 2021). This shows that the popularity of games has not gone down post pandemic and there is a market for even more games to be developed".

In conclusion, the gaming industry is booming and there are games that are being developed daily for every class of player. Hence the desire to develop a game for my final year project.

1.2 Aims and Objectives

1.2.1 Aim

The aim of this project is to create a multiplayer networked 2D PC game that can have up to four players join the game. The players will collaborate with each other to eliminate all the viruses that are spawned in the game. The game will have different playing levels with different capabilities and objectives. The aim of the game would be to beat the high score that can be viewed from the Lobby screen.

1.2.2 Objectives

- Select a methodology that works for multiplayer games.
- Design flowcharts.
- Implement a basic standalone prototype.
- Integrate the networking features to allow multiple players to join the game.
- Add new features that would enhance the game.
- Minimise networking lags and synchronisation by developing a suitable method.
- Implement code for when the network is unreliable.

The game design will allow for smooth and intuitive connection to the game arena, with increasing levels of difficulty as the game progresses. The objective of the game would be to beat the high score, and this can be done by avoiding the viruses and going up the levels which carry on until the player/s are all eliminated.

The game would prove challenging as the number and speed of different viruses would increase as they travel towards the player, and one of the challenges will be to work out which features and capabilities to implement and update for the player to allow the player to compete effectively as the levels and opposition capabilities of the game become more difficult to overcome.

1.3 Problem Statement

There is a limited availability of games which fit the criteria of being multi-player, networked, with a collaborative goal to reach a common objective. The points to be taken into consideration were; how to select the best game engine to drive this project; the different points to consider when implementing a variety of graphical effects which are visually pleasing and yet effective; and how to manage time in a time-constrained project by working out what features to implement now and leave other features for future enhancements.

In designing Virus Attack, the first hurdle was to find examples in the industry of games which had multiple players playing over the internet in a networked environment, playing in the same arena against a common enemy or set of enemies. I explored the feasibility of developing a 2D multiplayer game as they have been taken over by the 3D gaming world which require high end graphics and computers to play. I researched for games in this genre and could not find many multiplayer PC based games. Hence, I was inspired to design and develop this game. I also investigated game engines and chose Unity3D which is the most popular game development tool which has a fully integrated professional game engine. It was decided to keep the visual design simple but effective and managed my time to create the current game.

1.4 Report Layout

This report will provide a detailed overview of the design and development of the game that is designed to work on a PC platform with an online multiplayer capability, which is the main feature of this game.

The main areas covered in this report would be:

- ❖ **Introduction (Chapter 1)** – This gives a background, aims, and concepts behind the project.
- ❖ **Literature Review (Chapter 2)** – This section is about researching the game.
- ❖ **Methodology (Chapter 3)** – Is about the methodology that is used to develop the project.
- ❖ **Design phase (Chapter 4)** – Gives a view of the design of the game, including the UI and functionality.

- **Development and Implementation (Chapter 5)** – The development process including options for other features and future enhancements.
- **Testing and Results (Chapter 6)** – Testing and results obtained.
- **Conclusion (Chapter 7)** – A summary of the process of creating the project with what went well and what did not go to plan.
- **Appendices** – Includes the code, screen shots of the game, references to sections of the project not defined in the main report.

1.5 Project Feasibility Report

This section discusses the feasibility of developing the game taking into consideration all the various aspects which are necessary to explore.

1.5.1 Financial Feasibility

The project was designed to use software and tools that are available on the market without cost. Virus Attack is developed using Unity 3D with the benefit of a tight integration with Photon Unity Networking (PUN) to launch a multiplayer game. This software is free to download for 20 concurrent network users and the only requirement would be for internet connectivity which most people have freely available.

1.5.2 Technical Feasibility

The tools and technology used in making Virus attack are:

- Game Engine: Unity 2021.2.18f1.
- Photon Unity Networking (PUN)- for multiplayer online network capability, only available to the EU region. The plan that was downloaded for this project gives 20 concurrent users (CCU) for free.
- Programming language: C#. Unity integrates with Visual Studio and makes writing C# code and executing in unity quite easily.
- Code Editor: Visual Studio 2019
- Background Image and other images used: These images have no copyright issues and are free to download. Details provided in the reference section.
- A PC/Laptop with Windows 10 Operating system with a minimum RAM of 4GB, connectivity to internet, and a decent GPU, recommended hard disk space at least 220GB.

The above software, development environment, and hardware is available for this project and in conclusion the project is completely feasible.

1.5.3 Legal Feasibility

As the project does not store any player information, there was no requirement to do a legal feasibility study.

As the feasibility study above shows, the process to create the game was financially and technologically possible.

To conclude this section, all the elements described above were combined to create the game, Virus Attack. I decided to implement a 2D multiplayer game as there is still room for these types of games for the casual player audience. With the feedback from friends and family who are avid gamers, I was able to implement the game in the way that I had originally envisaged.

1.6 Risk Analysis

When developing any software projects there are always risks and it is important to identify them so that solutions to overcome them earlier on in the software development cycle are managed.

The risk of using a new development environment, Unity3D and multiplayer capabilities using Photon Networking. This was partly overcome by going through online tutorials and forums to find solutions to any problems encountered. Risk of not meeting the deadline due to assignments in other modules. To manage my time effectively, I followed a project plan and tried to meet the deadlines.

Problems within development, objects not synchronising on the network. Further down into the implementation of the code, there were some issues with the network synchronising of the objects. This was overcome by redesigning the code to cater for this.

1.6.1 Cheating

Cheating in online games is possible and can happen in different ways. Regarding my project, the effects of cheating were considered as follows:

1 – Finding ways to enhance capabilities of killing viruses.

2 – making the player invulnerable to attack and so preventing the player from dying.

3 – Finding ways to affect score keeping to present a higher score than that actually attained.

Taking the above into consideration, in the scope of my project the capacity to cheat is limited and the effects of cheating would not have any great impact. The most obvious way to cheat, would be to amend the internally recorded score attained by the local player. At present this score is simply stored in a local text file, so the ways this could be tampered with could be:
Change the file to show a higher or lower score. This would not affect any other users, so the impact of cheating would be to satisfy personal ego .

Potentially an external user could hack into the server to change the score details. A way to avoid this issue would be to change the names of the files so that they are not obvious (currently the name includes the text "leaderboard"); the file could be changed from a standard text file, to storing the information in binary code so that it is harder to decipher and alter; a more elaborate form of encryption could be adopted to further hamper attempts to doctor the leaderboard information.

I cannot see any way that an external hacker could amend or affect the local PC to introduce software which could aid in aiming for the player; change the health or damage status of the player or virus; or any other ways to change how the software works. The current project is not sophisticated enough and does not reach a wide enough audience to warrant efforts in finding complicated ways to cheat, therefore I do not think I need to consider any avoidance methods apart from possibly encoding and renaming the leaderboard files.

2. Literature Review

During the research stages for the genre of game I wanted to develop, there were a few 2D multiplayer online games available to play on the PC. The goal was to develop a simple game for a casual player that is enjoyable and challenging. Three 2D multiplayer games were reviewed to see which features to include and new ones that I would like to implement in my game. These three games below were chosen for review:

- ZombsRoyale.io is a 2D multiplayer battle royale game that is set in a Battle Royale arena and can be played on iOS, Android, and web browsers. The game is played in rounds that last for several minutes. Each round admits between 50 to 100 players on a 2D grid-like map. Players can acquire weapons, skins and healing times and battle passes. (Zomb Royale, n.d.). Although this game is a completely different concept to mine, the game has features that have been implemented in Virus Attack, such as the WASD keys, left mouse click to shoot, mini map and health bars.
- BlockTanks is an online tank battle .io game. Simply shoot enemy tanks to increase your score. The maps are designed for tactical shooting. Be reactive and quick! (Crazygames, n.d.). This is an online shooting game that uses 'WASD' keyboard key controls to move, mouse scroll and is built on a web browser.
- Agar.io is a fun addicting Massively Multiplayer Online Game (MMO) in which you must eat or be eaten to strive to dominate the world of colourful cells. Try to grow larger and create the biggest cell. Users have excellent reflexes and keep moving whilst avoiding enemies. Fans of this game described the game as "a good abstraction of the fierce survival of the fittest competition" (Agario.io, n.d.). This game is built on a web browser, can be played with friends in the same arena, is playable on a maximised screen and has a survival for the fittest message. This is something I wanted to implement on my game where the player is shooting the virus to survive.

From the analysis of the three games that were chosen, shortcomings were found which I wanted to address so that I could improve the implementation of these and other features in my game. Some of the main features that have been included are:

- Enabling networked multiplayer capability.

- Creating a menu system.
- Creating a lobby and room to join.
- Enabling players to enter the lobby and choose the room they want to enter.
- A ‘Quick Match’ option to play the game in a random room that is automatically generated.
- Displaying the health of all the players and the viruses.
- Use of ‘WASD’ keys, arrow keys and mouse movements.
- Facility to display the whole of the game scene to enable players to see other players, structures, and objects in the whole arena.
- Health packs to gain health points.
- Allow a maximum of four players in a room.
- Displaying the name of all players.
- Having different viruses.

3. Methodology

There are various software development methodologies that are available with the most traditional one being the linear Waterfall method. This is a less flexible approach as each stage needs to be completed before the next one can begin. This method is suitable for large plan-driven teams who have a better understanding of the projects scope. Testing is only conducted at the final stages of work.

The methodology chosen for the development of Virus Attack is Extreme Programming (XP) which is an Agile software development methodology. It uses an approach of starting off by building simple code that can be improved continuously by using iterative cycles of development and continual testing and revision. This was the most appropriate method for this project as it is suitable for single person software development projects and regular testing is an important feature of this methodology.

The iterations in XP consist of five basic principles:

Planning

This stage involved creating the requirements of the project. The features and functionality of designing a 2D game were researched and a plan was created to determine the order in which to implement each feature as required in the specifications.

Designing

The design has simplicity in mind to code and test often to create fault free software. A simple system was created initially to get the flow of the game with simple square/circles of the player and viruses and was tested for the flow from one screen to another.

Coding

This is the most important phase in the XP life cycle. Coding standards are maintained to ensure consistency and readability. Test units were created to ensure the requirements were met.

Testing

Early testing is a key part in the game design. Unit testing was done during the development, and this is an important phase of the project lifecycle to get error free code.

As video games are meant to entertain and thus testing is more focused on the end-user experience.

Feedback

Feedback from the system shows the state of the system after implementing any changes. At each stage of development, the project was tested on two or more machines separately and the help of family was enlisted to play the game. Any feedback from the 'user' at that stage of the game was analysed, and if appropriate, the relevant areas of the game were redesigned and developed.

3.1 Requirements

The requirement analysis was carried out and a summary of the results are as below:

- ❖ The game is a 2D networked multiplayer shooting game.
- ❖ The game is developed to be played on a PC and is initially set to a maximised screen size of 1920x1080.
- ❖ Can be played as a single player or multiplayer game.
- ❖ Multiple game levels, every game starts at level one.
- ❖ 2D GUI and menus that gets the player to the gaming arena.
- ❖ A Login screen will be displayed to enter the username.
- ❖ A Lobby screen to join a random room (Quick Match) or the option to create a new room.
- ❖ A Room screen showing the name of the player and the room name.
- ❖ Play game screen.
- ❖ Mini map mode to view a mini version of the game.
- ❖ There will be a max of 4 players in any room.
- ❖ Labs created dynamically to spawn viruses and the number spawned increases with game level (maximum of four generated in the game).
- ❖ Create a path finding algorithm to calculate the shortest path from the virus to the nearest player, to create a 'homing missile' effect.
- ❖ Move the player around the arena using a combination of keys and mouse.
- ❖ A player cannot 'kill' another player.
- ❖ A virus cannot 'kill' another virus.

- There will be health packs where the player can collect extra health points.
- There will be Ammo packs to allow the player to pick up certain ammunition.
- Game Play scene:
  - Press the ‘M’ or Tab key to go to mini map mode.
  - The player should be able to navigate through the game with the common ‘WASD’ keys, Arrow keys, and mouse movements.
  - Use of up, down, left, and right arrow keys and combination of arrow keys to move:
    - up and left to go in the northwest direction.
    - up and right to the northeast direction.
    - down and left to the southwest.
    - down and right southeast.
  - ‘Z’ key implemented to move player in the direction the player is facing, i.e. towards the location of the mouse pointer.
  - Left mouse click to shoot an antidote.
  - Use of left mouse pointer to rotate the player.
  - Use of spacebar or right mouse click to fire a multi-hit bullet.
  - Leave option to exit the game to the Lobby Screen.
  - ‘Esc’ key to Logout/Quit or go back.
  - Display player name.
  - Display health of players.
  - Display health of viruses.
  - Display health packs.
  - Display Labs.
  - Display Ammo count.
  - Display High Score information.
- Player can move to next level once all viruses eliminated.
- Collision detection so that players and viruses are within boundaries and if the player collides with the virus, then both health of virus and player goes down.
- Collision detection of virus and players with the labs and other objects on the screen.
- Disconnect Manager.

- Terminating Screen.
- Game Over screen.

3.2 Development Plan

This section will give the plan of the simple game to get it up and running and then adding new features and functionality within the given time frame. An outline of what is required is shown below.

3.2.1 Project Creation

- Downloading Unity 3D and Visual Studio to edit C# code.
- Importing and setting SDK in Unity.
- Using Photon cloud's APIs used to integrate multiplayer features.
- Creating free account on Photon website. Get the AppID which needs to be used in Unity3D.

3.2.2 Login

- Background
- Layout of 'Login Button' and input name field on screen
- Screen in which the player will enter their name and login
- 'Esc' will Quit the application or go back to login screen
- Help icon will be displayed on top right corner.

3.2.3 Lobby

- Load the lobby screen where the player will have the option to create a room or select an existing room name or create a 'Quick Match' random room.
- The background, layout of buttons., text and input fields consistent with login screen.
- Leader board and help icons will be displayed on top right corner.

3.2.4 Room

- Display interim screen between the lobby and player screen which is purely an information screen that will display the player name and room name.
- Background and layout of text consistent with the first two screens above.
- Leader board and help icons will be displayed on top right corner.

3.2.5 Playing The Game

- Create player avatar.

- Spawn labs.
- Spawn viruses.
- Spawn health packs.
- Spawn ammo packs.
- Implement the camera to follow the player. The game will have a player view where the all the objects in the arena will be visible.
- Implement the background for the player view screen.
- Allow input so player can move around the map.
- Enable collision detection between players, viruses, labs and health packs, borders, ammo packs.
- Display count of multi-bullets available to the player

3.2.6 Mini-Map

Create a zoomed-out version of the player view which will be visible on the top right corner of the screen to help the players with orientating themselves and viewing items in their immediate vicinity. When the 'M' key or TAB key on the keyboard are pressed, the large map mode is displayed, and clicking 'M' or TAB key again brings the game back to player view mode.

3.2.7 Objects and Characters

There will be five main object types on the gaming arena, the player/s, labs, viruses, health packs and ammo packs.

3.2.8 Player/s

Develop game with functionality to play as standalone or networked with other players. The player name is displayed on the player avatar prefixed with the text "You" for the host. The host will display the text "Host" with the crown image on the top left side of the screen for the Host, or the crown image with the name of the Host if the display is for a client player.

3.2.9 Labs

Labs will spawn viruses dynamically, the number and type of viruses spawned will change depending on the game level attained.

3.2.10 Viruses
The viruses will be spawned from the labs. In the current implementation, 2 types of viruses will be spawned, known as Virus1 and Virus2, Virus2 has different capabilities than Virus1 and in general is faster, stronger and inflicts more damage on the player than Virus1.

3.2.11 Health Packs
Three versions of Health packs are available in the game. Health Pack1 will be spawned in Levels 1 and 2. Health pack2 will be spawned in levels 3 and 4. Health pack3 will be spawned in all successive levels.

3.2.12 Ammo Packs
Ammo packs are spawned in the game, the ammo packs will add a maximum of 3 multi-hit bullets to the ammo count available to the player. From level 1 to 4 the player can only have a maximum of 4 multi-hit bullets available at any time and from level 5 onwards the player can only have a maximum of 8 multi-hit bullets available at any time.

3.2.13 Player Controls
The player can move using mouse and keyboard controls as described elsewhere in this document. Player can fire the standard bullets using Left-Click on the mouse. If multi-hit bullets are available, the player can fire multi-hit bullets using the SPACE bar or Right-Click on the mouse.
The player can exit the game by clicking the ESC key, this will give options to display the HELP screen, or quit.

3.2.14 End Game
The game has been designed to have continuous levels with no maximum. The game will end when all the players who had been logged in are eliminated

3.2.15 Game Lost
Player has been killed by the viruses; game lost message should be displayed.

4. Design

The gameplay is action-based, and the goal is to kill all viruses that are spawned. The game is currently designed to be played online with a maximum of four players allowed in a room and with infinite game levels. All players will start at the first level and at the end of the game when all the viruses on that level have been eliminated, there will be an option to play the next level. The player starts the game with five health points. The attack from the viruses will reduce the health gradually and when the players health reaches zero, they will die. However, the player can pick up health power-ups to increase their health. The player has only one life and when the player dies, the rest of the players can carry on playing the game until they die. If the player kills all the viruses before their health reduces to zero, the team of players can proceed to the next game level.

4.1 Menu Layout

The menu flow chart below shows the design flow of the game that aided the implementation process. The process of the menu flow for the game is as below:

The player starts the executable.

- ❖ Login Screen.
  - Enter the Username.
  - There is an option to see the Help screen.
  - Click Login.

  The Lobby Scene is loaded, displaying the Lobby Screen at first.

- ❖ Lobby Screen.
  - The player will create a room.
  - Or the player will select a room from the list.
  - Or the player can choose the 'Quick Match' option.
  - There is an option to see the Leader board scores.
  - There is an option to see the Help screen.

  Once the above options are chosen, the Room Screen is loaded, within the Lobby Scene, with the player's name and room name selected is displayed on this screen.

- Room Screen.
    - Has the option to Play or Leave the game.
    - The first player to enter the room is by default the Host of the playing session.
    - Other players will be displayed in the player list, they will be client players for the game session.
    - There is an option to see the Leader board scores.
    - There is an option to see the Help screen.

- Play Game Screen
    - The player enters the game, game play is described further in this document in detail

4.1.1 Menu Flow Chart

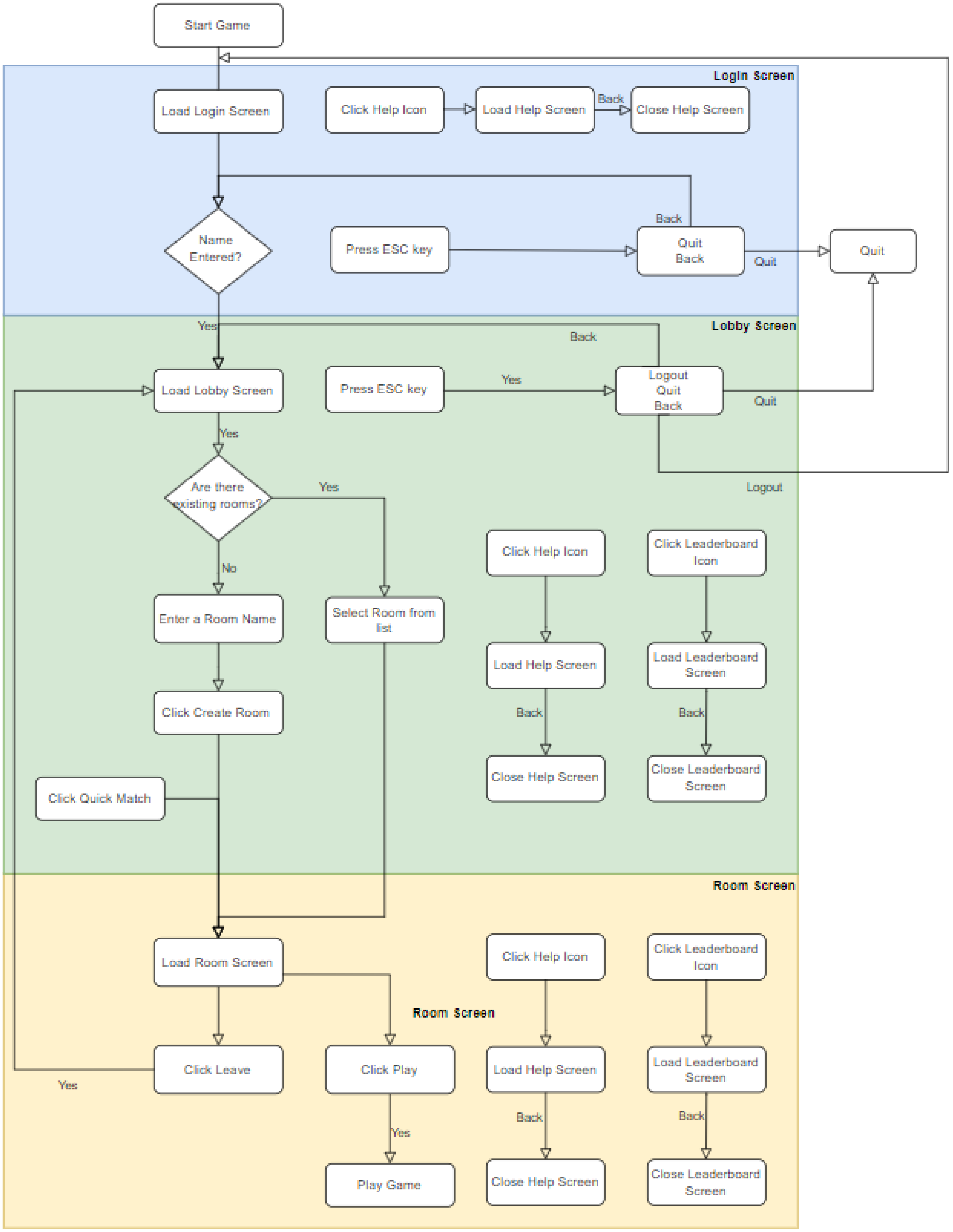


Figure 1: Menu Flowchart

4.2 Game Flow

The flow diagram “2 Game Play” shows the design flow of the game scene.

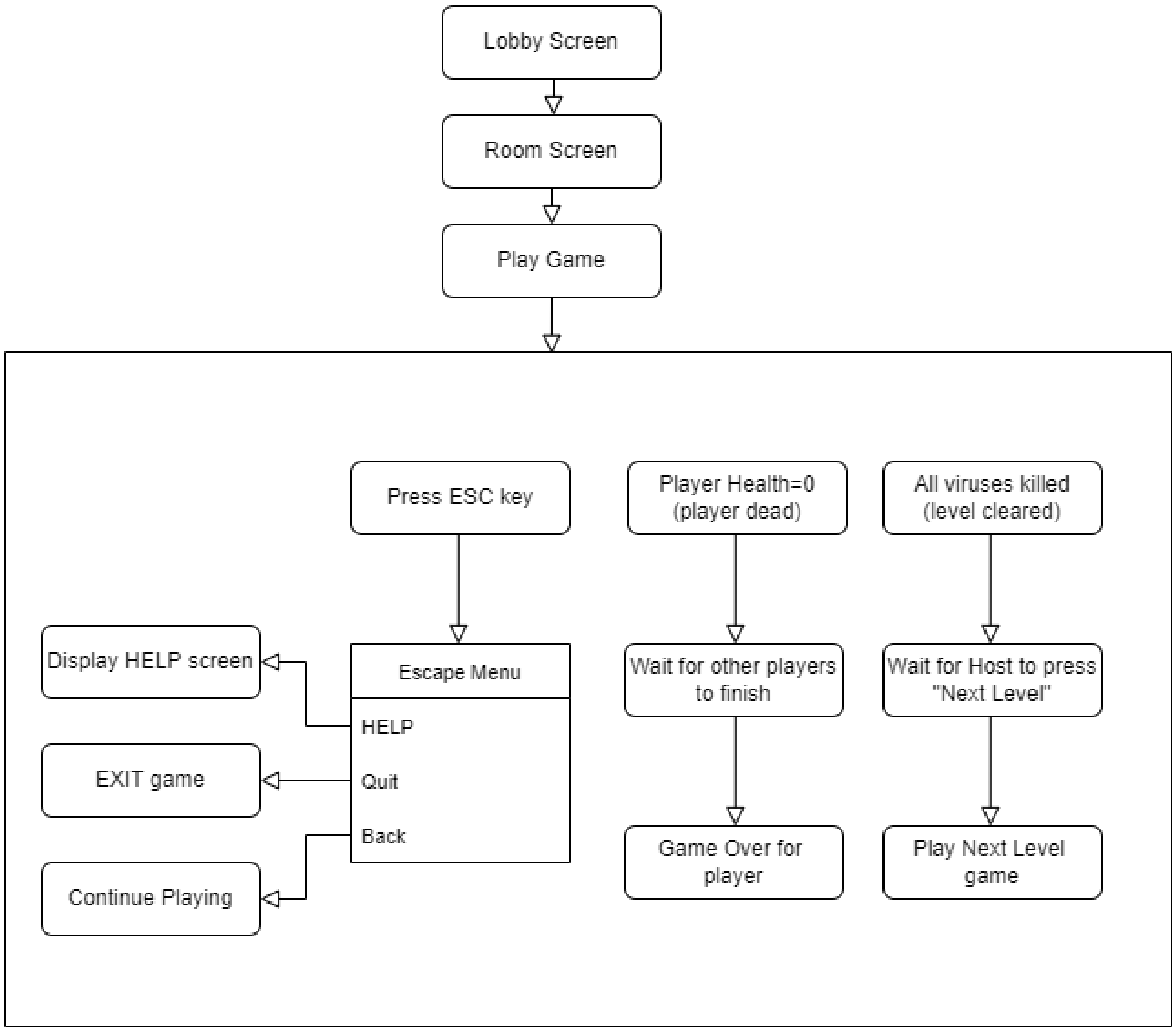


Figure 2: Game Flowchart

At this stage in the game, the player has started the game, the player, labs, and viruses are spawned on the map. There can be a maximum of four players, two labs and the number of viruses spawned from these labs is set to 4 per lab for level one. In Level 2, the labs spawn an extra 2 viruses each. The aim is for the players to kill all the viruses in that level, and if they succeed then they can move to the next level. The player has only one life and if their health reduces to zero, the player will die and they will not be able to continue game play, but they will still be able to see other players in the game.

4.3 Game Controls

The game is played by using the keys and mouse as described below.

Esc – To exit the screen.

W – Move the player up.

A – Move left.

S – Move down.

D - Move right.

Z- Move the player in the direction of the mouse pointer.

Mouse pointer – Defines direction the player will point (for firing or movement by Z key).

M or TAB – Invoke the mini map mode.

Mouse Left-click – Fire primary bullet.

SPACE bar or Mouse Right Click – Fire multi-hit bullet.

The arrow keys can also be used to move the player:

Move up, down, left, and right.

Pressing combination of WASD or the Arrow keys will move the player in a diagonal direction.

An illustration of the keyboard operations is shown below.

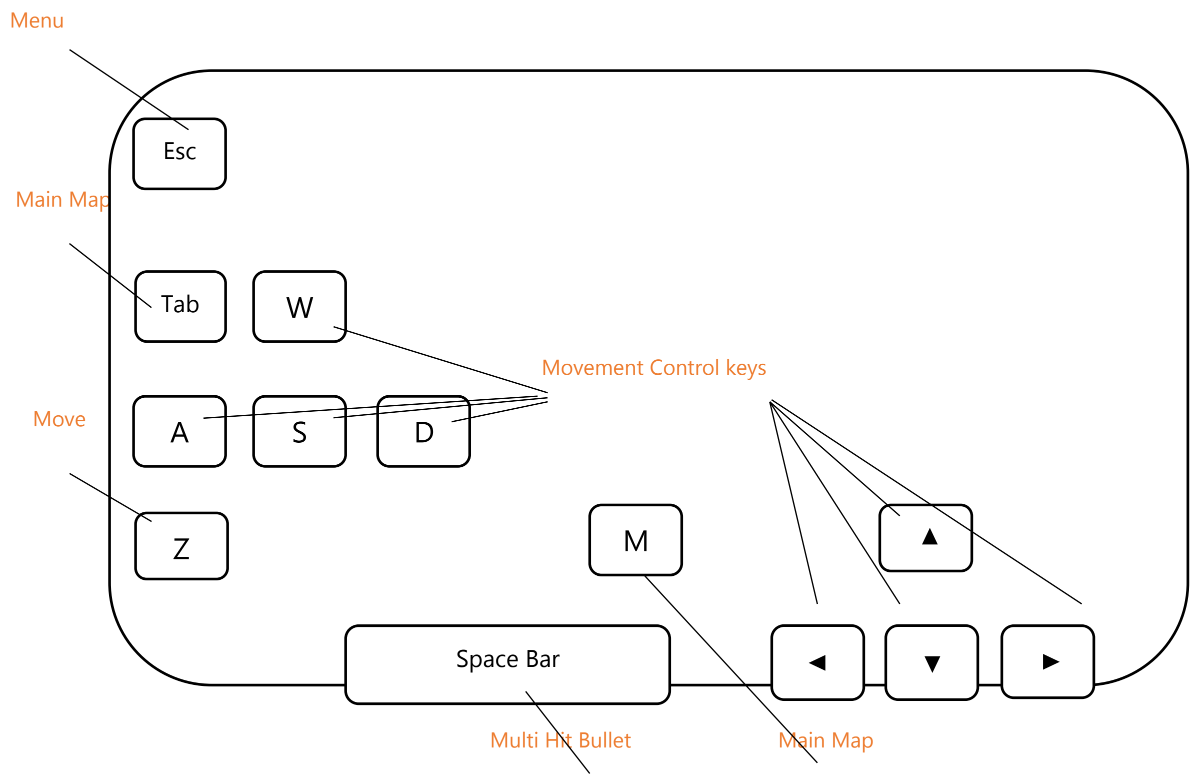


Figure 3: Game Controls

4.4 Game Play/Rules

4.4.1 General Game Rules

- Players are not able to navigate beyond the borders of the game.
- Players have unlimited Standard bullets.
- Players initially have 1 multi-Hit bullet, which is regenerated after a random amount of time.
- If a player shoots a virus with a standard bullet, the health of the virus is reduced by 1.
- If a player shoots a virus with a multi-hit bullet, the health of the virus is reduced by 3.
- If a player shoots an Ammo pack, the Ammo pack is destroyed.
- If a player shoots a health pack, the health pack is destroyed.
- Game Play will continue until either the player health falls to zero (Player is killed), or the viruses spawned on that Game Level have all been killed (Level Complete).
- If the player has been killed, the player is no longer visible in the game and may not participate in the normal game activities. The player screen will, however, still be able to view the game progress of other players, if they are present, and the player can still move around the arena albeit with no other interaction in the game.

- ❖ If all players are killed, the game will display a message and game play will stop. The players will have an option to exit the game or go back to the room/lobby.
- ❖ If all viruses are eliminated in this game level, the game will display the appropriate message and game play will be paused pending the following actions:
    - o If the player is a client player, then the player must wait for the Host player to click the button.
    - o If the player is the host player, they must click on the "Next Level" button.
- ❖ When the player has been killed, the game will compare the leader board, which is stored for that player on their server. If the player has exceeded the stored game level, and more viruses have been eliminated at that point than the number stored in the leader board, then the game will update the leader board with the player name, the game level reached, and the number of viruses left on that level at the time the player was killed.
  NOTE that the leader board is unique for each PC server that the game is played on. The leader board is not shared between the players, so the high score on different servers may not be the same across all players.

The leader board has 4 entries, labelled solo, duo, trio, squad. Solo means that the high score was attained when the player was playing on their own; Duo means that the high score was attained when two players joined the same room and played together; Trio means that the high score was attained when three players joined the same room and played together; Squad means that the high score was attained when four players joined the same room and played together.

4.4.2 Standard bullets

The players have an unlimited number of standard bullets. The bullets are fired by the player clicking on the left mouse button, and the bullet will fire in the direction of the player gun nozzle, which can be rotated by moving the mouse pointer in the appropriate direction.

4.4.3 Multi-Hit bullets

The players start the game with 1 multi-hit bullet. The multi-hit bullets are regenerated after use after a random amount of time up to 5 seconds, and the number of bullets in possession by the player are incremented by 1, up to a maximum of 1 bullet. Once fired, the multi-hit bullet cannot be replaced until the bullet has completed its cycle and is destroyed. Once it is destroyed, the multi-hit bullet is regenerated after a random time.

The multi-hit bullet inflicts damage of 3 on any virus it impacts with and will destroy any Health pack or Ammo pack it hits. The Multi-hit bullet is so called because the bullet will not destroy itself until it hits a lab, a health pack, an ammo pack, or the border. Therefore, one multi-hit bullet can damage multiple viruses on its trajectory.

The multi-hit bullet, once fired by the player, will initially set off in the direction of the player muzzle. However, the player can amend the course of the multi-hit bullet by moving the mouse pointer in the appropriate direction, so with experience, the multi-hit bullet can be guided to hit multiple targets by the player. However, a problem while doing this is that while the multi-hit bullet is being guided by the player, any other standard bullets that the player has fired will shoot in the direction of the player nozzle at the time the standard bullet is fired.

The Player will normally get 1 free multi-hit bullet, which is replenished once it is used after a random time. However, the player can acquire up to 3 additional multi-hit bullets by collecting an Ammo pack. The number of multi-hit bullets that the player has in their armoury is displayed on the screen, under the mini map, as a counter displayed next to the icon of the multi-hit bullet. The player can fire one or more multi-hit bullets up to the number of multi-hit bullets displayed at the counter, and the counter will decrease every time the multi-hit bullet is fired. The maximum number of multi-hit bullets that can be collected and stored by the player is maximum 4 bullets up to game level 4, and from game level 5 onwards, the player can collect and store up to 8 bullets. For example, if the maximum number of bullets is 4, and the player currently has 2 stored, then collecting an Ammo Pack would normally add 3 more bullets, i.e. total 5, but as this is more than the maximum allowed, the number of multi-hit bullets stored would be 4.

4.4.4 Game Level One

A player starts with 5 Health points.

V1 viruses have health level 3.

V2 viruses have health level 6.

The game will spawn 2 labs, the labs can be spawned in any 2 random locations out of 4 pre-defined locations on screen. Those locations will be reserved so that successive labs cannot be spawned in the same location.

Each lab will spawn 4 viruses. 3 of the viruses are Level1 viruses, 1 virus is a level2 virus (see description of viruses below).

The game will spawn up to a maximum of 2 health packs at random intervals at random locations on screen. As each Health Pack is used or eliminated, the game server will replace the health pack after a random interval of time in a random location. The Health Packs will be Level1 Health Packs (see description of Health Packs below).

The game will spawn up to a maximum of 2 Ammo Packs at random intervals at random locations on screen. As each Ammo Pack is used or eliminated, the game server will replace the health pack after a random interval of time in a random location.

The Pathfinder range minimum value will be set to 300 pixels (see description of Viruses below).

4.4.5 Successive Game Levels

The following limits will be set for all successive game levels, with the exceptions specified below:

All other Game Levels

- ❖ Pathfinder range limit increased by 50 pixels.
- ❖ Maximum number of V1 viruses increased by 1 per lab.
- ❖ Maximum number of V2 viruses increased by 1 per lab.
- ❖ Maximum number of Health Packs available on screen increased by 1.

Game Level 3

- ❖ The game will spawn 1 more Lab.
- ❖ The game will spawn 1 LESS V1 virus per lab (NOTE total number of viruses on this level will still increase as there is one more lab on the screen).
- ❖ The game will spawn 1 more V2 virus per lab.
- ❖ The Player Health maximum is increased to 8 health.

- ❖ The Health Pack level of any new health packs spawned after this time will increase to HealthPack2 (see Health Packs below).

Game Level 5

- ❖ The game will spawn 1 more Lab.
- ❖ The game will spawn 1 LESS V1 virus per lab (NOTE total number of viruses on this level will still increase as there is one more lab on the screen).
- ❖ The game will spawn 1 more V2 virus per lab.
- ❖ The Player Health maximum is increased to 10 health points.
- ❖ The Health Pack level of any new health packs spawned after this time will increase to HealthPack3 (see Health Packs below).

A summary of the maximum health that the player can have in the different game levels is shown below:

Game Level	**Maximum Player Health**
1 and 2	5
3 and 4	8
5 and above	10

Figure 4: Table showing the player health at each level

4.4.6 Mini-Map Mode

A mini map will be shown as a small map that appears in the top right-hand corner of the players' screen to show the player their location in relation to the game world. The Mini-Map will show a greater area of the game arena than can be seen on the game play screen. The player can see where the viruses, labs, health packs, ammo packs and other players are in relation to their own location.

4.4.7 Main Map

The player can press the TAB key or the M key to bring up the Main Map. The Main Map will show the whole map of the arena, including the positions of other players, viruses, labs, health packs, and ammo packs. While in Main Map mode, the player can move but cannot shoot their bullets, the player must return to the game screen to continue normal game play. The player can press the M key or the TAB key to get back from Main Map mode.

4.4.8 Labs

Labs are spawned dynamically within 4 specific locations when game play starts. The rules for the labs are:

- ❖ At game levels 1 to 2, there will be 2 labs spawned in the game.
- ❖ At game level 3, an extra lab is spawned in the game.
- ❖ At level 5 an extra lab is spawned in the game.

The number of viruses spawned by the Labs at different game levels will change depending on the game level, as described above. The numbers of viruses, and the maximum viruses in the game, will be as follows:

Game Level	Total number Labs on screen	Total Number Virus1 per lab	Total number Virus2 per lab	Max number of viruses per level
1	2	3	1	8
2	2	4	2	12
3	3	3	3	18
4	3	4	4	24
5	4	3	5	32
6 and above	4	3 +1 for each successive level	5 +1 for each successive level	32 + 8 for each successive level

Figure 5: Table showing the number of viruses and labs at each level

4.4.9 Viruses

The viruses will be coded with basic Artificial Intelligence (AI), otherwise known as pathfinder, that will attract them to the nearest player if the player within range. The pathfinder range limit will increase as the Game Levels increase. The virus will follow the player until it is killed by the player, or it falls outside the pathfinder range. If the virus falls outside the pathfinder range, it will generate a random direction and move in that direction.

The virus will change direction to rebound logically when it collides with an object on the screen. The exception to this rule is when the virus is in Pathfinder Mode, i.e. it is within range of a player and is heading towards that player. In this instance, the virus will try to calculate a route around the obstacle so that it will continue to target that player. The virus will only stop doing this if it falls outside the pathfinder range limits.

While the virus is in Pathfinder mode, if it encounters an obstacle which is in a line directly between the virus and the player, the pathfinder calculation will try to find a path around the obstacle which still travels towards the player. For example, if a lab is in between the virus and the player, if the virus is near the bottom left side of the lab, and the player is beyond the top right side of the lab, then the virus will try to move in an upwards direction to get around the lab. The code has also had an adjustment to the recalculating algorithm, to alter the path around the obstacle so that the virus has a 'wiggle' introduced to its avoidance path; this makes the travel of the virus appear to be more threatening to the player.

There are currently 2 types of viruses in use in the game:

- V1 – Has health 3 and can inflict damage 1 on contact with the player.
- V2 – Has health 6 and can inflict damage 2 on contact with the player.

The travel speed for V2 is faster that the travel speed for V1, and has a smaller size, thus making it more difficult to hit.

If the health of the virus falls to 0, then the virus will be destroyed, and the count of the number of viruses which will be available on that level will be reduced and the display updated on the top of the screen.

4.4.10 Health Packs

Health packs are collectible objects that are placed in the arena dynamically. Every player starts with a health of five points. When a bullet or player collides with a health pack it will disappear and respawn at a random space. If a player collects a health pack by colliding with it, their health level will increase by a number of points depending on the Health Pack Level of the Health Pack.

However, the Player Health level cannot have a higher value than the limit defined for the player at the appropriate game level, as described previously in this section. The maximum amount by which a player's health can be increased on collection of a health pack is as follows:

- HealthPack1 – 3 points.
- HealthPack2 – 5 points.
- HealthPack3 – 7 points.

As the player plays in different game levels, Health Packs of different levels are generated to replace one which has been collected or destroyed, as shown below.

Game Level	**Maximum number Health Packs on screen**	**Health Pack level of new Health Packs spawned**
1	3	1
2	4	1
3	5	2
4	6	2
5	7	3
6 and above	7 + 1 for each successive level	3

Figure 6: Table showing the number of health packs at each level

The maximum number of Health Packs available during each game level starts with 3 Health Packs on game level 1 and increases by 1 extra Health Pack for each successive game level.

As the Health Pack is collected or destroyed, it will disappear from the screen and the new Health Pack will be respawned after a random delay of between 10 and 15 seconds in a random location on the screen. The new Health Pack will be at a level which is appropriate for the current Game Level.

4.4.11 Ammo Packs

Ammo packs are collectible objects that are placed in the arena dynamically. The Ammo Packs, when collected by a player, will increase the number of multi-shot bullets the player has by 3 bullets, up to a maximum of 4 bullets if the current game level is between 1 and 4, or a maximum of 8 bullets if the current game level is 5 or above. Every player starts with one multi-shot bullet. There are a maximum of 2 Ammo Packs available on screen during the game play, and as each Ammo Pack is collected or destroyed, a new Ammo Pack will be regenerated after a random delay of between 10 and 15 seconds.

4.5 General Play

4.5.1 Player Movement

Movement for the players in general is undertaken by using the keyboard and mouse together. Movement by keyboard only is possible by using the Arrow keys, or by using the WASD keys. W or Up Arrow will move the player upwards; A or Left Arrow will move the player to the left; S or Down Arrow will move the player downwards; and S or Right Arrow will move the player to the right. A combination of the keys will enable the player to move in a diagonal direction.

Movement using the keyboard and mouse is possible by moving the mouse cursor in a position you want the player to move towards, and then press the Z key. This will move the player in the exact direction towards the cursor.

If the player collides with a virus, the health of the virus is reduced by 1, and the health of the player is reduced by 1 if the player collides with Virus type 1, or 2 if the player collides with Virus type 2. The amount of damage inflicted on the player will be indicated by text, which is visible for 5 seconds, displaying either "-1" or "-2" as appropriate, the text will be displayed in the position that the player was in when the damage was inflicted. After 5 seconds the text will disappear.

If a player collides with a Health Pack, the health of the player will increase depending on the Health Pack level that the player collided with, and the player health cannot increase more than a maximum for that game level. These limits are defined in the Game Rules above.

If a player collides with an Ammo Pack, the number of Multi-hit bullets available to the player to fire will be increased by 3, up to a maximum of 4 if the current game level is between 1 and 4, or a maximum of 8 if the current game level is 5 or above.

The player will not be able to move beyond the boundaries of the screen borders.
The player movement will be blocked if the player encounters a Lab; the player will need to navigate around the lab and will not be able to go through it.

4.5.2 Player Controls
The player can fire bullets in the following ways:

- ❖ Standard bullets can be fired using the Mouse Left Click. There is no limit of Standard bullets that the player can fire.
- ❖ Multi-hit bullets can be fired by using the SPACE bar, or the Mouse Right Click. The Multi-Hit bullet icon is displayed on the screen with a number next to it, this number shows how many multi-hit bullets are available for the player to fire. When fired, the number will decrement, and the player will not be able to fire another bullet unless the number is greater than 0. If the number of bullets falls to zero, the bullet must be expended, i.e. the bullet needs to destroy itself, and then a random delay of up to 5 seconds is run, and after the completion of the random delay, a new multi-hit bullet is available for the player to use, and the counter is updated accordingly.

The player can pick up the various packs from the screen. The different Health Packs and Ammo Packs are described in the section above, and as each pack is collected, the icon on the screen is removed, and the pack is respawned at a random location on the screen after an interval of time will be a random value of between 10 and 15 seconds for each new pack.

4.5.3 Virus
There are 2 types of viruses implemented in the game, Virus1 and Virus2. Virus1 is slightly larger and will inflict damage 1 on the player if the virus collides with the player. Virus2 is smaller than Virus 1, and therefore harder to hit with the bullets from the player. Virus2 is also slightly faster than Virus1, and has double the health of Virus1, and will inflict damage 2 on the player if it collides with the player, therefore it is harder to kill than Virus1 and is more likely to kill the player on impact.

The virus is instantiated by the lab, and its initial direction of travel is set randomly. Once it is moving, if the virus comes across any obstacle, the virus will calculate its position relative to the object and will 'bounce' in a direction which would mimic real life. If a virus hits another virus, it will calculate the rebound angle relative to the direction of travel both viruses were travelling in.

Viruses have a pathfinder functionality inbuilt into them, at game level 1, the range of the pathfinder is 300 units, and every subsequent game level the pathfinder range increases by 50 units. The role of the pathfinder is initiated when the virus finds a player who is in range, and the virus will calculate a direction of travel which will target the player directly. The virus is continuously scanning for all players, so if it comes across another player who is nearer to it than the one it was following, the virus will target and follow that player instead. The players travel speed is marginally faster than that of Virus 2, and if the player travels out of the range of the pathfinder, the virus will detect this and assuming there are no other targets near enough, the virus will recalculate a random direction to travel in and move in that direction.

If, while in pathfinder mode, the virus comes across an obstacle, the virus will calculate a trajectory which takes it in a direction around the object, so that it can continue to target the player. The calculation around the obstacle includes in it a ‘wobble’ to the redirection route, so that the virus appears to ‘wriggle’ around the obstacle, this was done to make the virus appear more sinister in its attempt to reacquire its target, as opposed to the smooth route it would take around the obstacle otherwise.

4.5.4 Standard Bullets

The standard bullets are fired by the player and will take their travel of direction in the direction that the player weapon is pointing. The standard bullet will continue its path until it hits the arena boundary or any other obstacle, in which case the bullet will destroy itself. If the bullet hits a Health Pack or an Ammo Pack, the pack will be destroyed. If the bullet hits a virus, it will inflict 1 damage point on the virus impacted. The player has an unlimited number of standard bullets, and the range of the bullets is only bounded by the arena.

4.5.5 Multi-Hit Bullets

Multi-hit bullets are so called because they do not get destroyed if they hit a virus. The multi-hit bullet will damage the virus, and then go around it and damage another or all viruses that it subsequently hits.

The multi-hit bullets will be fired initially in the direction that the player weapon is pointing in. Subsequently, the multi-hit bullet direction of travel can be amended by moving the weapon muzzle with the mouse, and this gives some capability for the player to try and hit more than one virus if there is a group of viruses following the player. The multi-hit bullets give 3 damage to the virus it hits. Therefore a bullet can destroy a Virus 1 in one hit, or a Virus 2 in 2 hits.

4.6 Game Future enhancements

4.6.1 Game Optimisation

- Thorough review of game logic and methods to optimise efficiency of code with particular focus on network related calls and procedures.
- Review of current implementation to ensure best practices are improved on for the whole project.

4.6.2 Game Difficulty Levels

- Introduce Easy/Medium/Hard levels which the Host can select for the game play.
- Different game levels will implement different features and limits for the players and viruses, and different number and availability of bullet types, Health Packs and Ammo Packs.
- Game parameters to control the type and availability of all variables could be made available for the Host to select prior to commencement of the game session .

4.6.3 Extra Virus Levels

Virus Level 3 which is more aggressive as it is faster and inflicts greater damage on the player.

Virus Level 4 which can shoot a weak projectile towards the nearest player.

4.6.4 In-Game Obstacles

Obstacles in the game (barrier) for the players and viruses to avoid and go round. These obstacles can work for and against the players, as they can act as an impedance to the movement of the player and block them from avoiding a virus; or the barrier could act as a barrier to stop the virus hitting the player. The obstacles would also block player bullets so a player could not hit a virus if a barrier were in the way.

4.6.5 Changes In Player Attributes

- Increase player health.
- Increase player speed.
- Increase player bullet damage.

4.6.6 Extra Bullet Levels

- Bullet missile:
  - A bullet which can pass through a virus it impacts with, destroying it immediately.
- Bullet missile 2:
  - A bullet missile which can pass through multiple viruses.
- Bomb:
  - A bullet which when it hits a virus, will explode with a blast radius which will destroy or damage all viruses within that radius. Need to consider if self-damage can be inflicted.
- Spread fire bullet:
  - Munition which will fire multiple missiles in a spread (3 or 5 bullets).

4.6.7 Increase Ammo Pack levels

- Increase the maximum number of multi-hit bullets a player can hold.
- Increase the number of multi-hit bullets a player can collect from that ammo pack.
- Provide different types of bullets as described above.

4.6.8 Increase Health Pack levels

- Increase the maximum Health level of the player collecting the new Health Pack.
- A Health Pack which will give the player Maximum Health immediately on collection.
- A Health Pack which will give the player invincibility for a limited amount of time, during that time the player could target viruses to collide with and damage in that way.

5. Implementation

This section is about the implementation of the project using Unity3D and C# scripts.

A brief outline on how the Unity environment is used to organise the project.

- ❖ The hierarchy panel organises the objects that can be placed in the scene. The SceneManager controls the scenes in the game and there are four scenes: Login, Lobby, Game and End scenes.
- ❖ Project panel has the hierarchy of all the assets that are used in the project.
- ❖ The Inspector panel is used for inspecting and adjusting all the attributes of the selected assets like its position, rotation and any constants used in the code can be set here. I have set the total game levels and total labs values here.
- ❖ The Scene panel is where the objects can be set up and moved around.
- ❖ The Game panel is a viewport for when the game is run and is a good indicator of what the game looks like.

Unity has inbuilt functions that are used in the project and a brief explanation is described below:

- ❖ UnityEngine.UI – Handles the display of UI, the clickable area for user input, e.g buttons and text.
- ❖ UnitySceneManagement – Handles different scenes in the game, like the game over or pause screens.
- ❖ SceneManager – Loads the scenes based on user input via the UI or if any event occurs like collision of player/virus/labs.
- ❖ RigidBody2D – Adding a Rigidbody2D component to an object will put its motion under the control of Unity's physics engine and will react to collisions with incoming objects if the right Collider component is also present. Colliders are vital in shooting games and are invisible in the game play. For Virus Attack, I used BoxCollider2D and CircleCollider2D. The collision event is handled by the script attached to the objects involved in the collision and Unity has inbuilt functions to detect collisions.
- ❖ Quaternion.Euler – Returns a rotation that rotates z degrees around the z axis, x degrees around the x axis, and y degrees around the y axis (in that order).

- ❖ EventTrigger – Receives events from the EventSystem and calls registered functions for each event.
- ❖ System.Collections – This namespace contains interfaces and classes that define various collections of objects, such as lists, queues, bit arrays, hash tables and dictionaries. Required if using classes like IEnumerator.

5.1 Game Engine

There are four main components to be implemented:

- ❖ The Game Engine is the heart of the game and provides services such as player movement, rendering graphics, collision detection and handling input.
- ❖ Networking using Photon game engine which provides connectivity to the cloud servers and enables the players to connect and play with each across the network.
- ❖ The Graphics used for the game are designed to look simple but effective. Any images that are downloaded have been included in the Citations and References section .
- ❖ The Scripts in the game handle all the collisions, input, and network communications.

5.1.1 Player Movement

To control the movement of the player, the RigidBody2D component of Unity is used which handles the physics like rotation and the movement of the player. This is handled by the PlayerMovement.cs script file where the rotation of the player towards the mouse position and the movement vectors from the keyword keys pressed are all calculated. Based on these calculations, the player moves and rotates accordingly.

5.1.2 Collision Detection

To detect collisions and simulate the real-world physics system, Unity has a built-in engine that handles all the maths behind collision detection. The response when a collision occurs is handled in the code using a collider 2D component. The lab and health pack objects have a BoxCollider2D component, and the virus, player and player bullet have CircleCollider2D components attached to them. The RigidBody2D component allows the physics engine to control the object and is affected by the collisions.

The collisions that are handled when one object collides with another one is:

- Virus-Player: Player health is reduced.
- Virus-Bullet: Virus health is reduced.
- Virus-Virus: Rebound and change direction depending on angle of impact.
- Virus-Border: Rebound and change direction depending on angle of impact.
- Virus-Lab: if the virus is not in pathfinding mode, it will rebound off the lab at an angle which depends on the angle at which it hit the lab. Otherwise, it will calculate the quickest way around the lab towards that player and move towards the player.
- Bullet-Virus: reduce health of virus and bullet destroyed.
- Health Pack-Player: health pack destroyed.
- Player-Health Pack: player health increased.
- Player-Lab: compare the angle at which the player is hitting the lab using the centre of the player object and the lab object. Depending on the angle of the impact, this will determine in which direction the player should move in to get around the lab.
- Player-Border: the player will calculate the angle it impacted the border with to work out which direction to go in to continue the path without going outside the border.
- Health Pack-Bullet: health pack and bullet destroyed.

5.1.3 Pathfinding Algorithm

The pathfinding algorithm stores all objects, with the "Player" tag attached, into an array. It loops through the array and finds the closest player. If the distance between the virus and the closest player is below the threshold, the movement vector is calculated which will enable the virus to move towards the player. Otherwise, the movement vector is randomised which means that the virus will move in a random direction.

5.1.4 Photon Transform View Classic

The Photon Transform View Classic component synchronises the position and rotation of all non-static objects by using lerp. Settings for synchronisation are explained below:

- If the distance between the last updated position and the current position is greater than 1000, the object is teleported to the current position. This enables the object's position to update when the internet connection is unreliable for the players.

- ❖ The interpolation setting is used for the synchronisation of the position and rotation of the object. This allows the object to smoothly move or rotate to a new position. Lerp is used which results in a rubber band effect for the synchronised object.
- ❖ Received values are a little bit out of date since it takes time for data to reach the object. The extrapolate setting resolves this issue by predicting the next position based on the last received position.

5.1.5 Invoke Functions

Over the course of the development of the game, there has been a significant difficulty with lag when there were more than four objects in the game. I soon came to realise that the network traffic was extremely high due to the number of RPC function calls made by the various objects on the screens as they synchronised their display on the clients' screens. There were too many of these messages per second and I realised that a much lower number of messages could provide a sufficient amount of synchronisation between the servers. Unity provides a functionality using an Invoke function, which allows the developer to initiate a method, with a specified delay, and Unity will queue the processing of this method, whilst at the same time continuing processing the code which immediately follows the Invoke function call. The This has led me to a solution that involves using invoke functions. The method which was called will be set off independently after the delay has elapsed, therefore normal processing of the block of code can continue without any delay, while the calls to the function which does not require to be run all the time can be reduced. The Invoke function can be controlled by setting a Boolean flag to indicate that the invoke function has been set off, so subsequent repetition of the code in successive frames will not run the invoke call. The method called in the Invoke would include logic to switch off the Boolean flag to allow the invoke function to be called in the next frame.

5.1.6 Networking using Photon

The master server is where all the players and rooms of the game will be located. Players can view all the existing rooms, join, or create new rooms. The players are known as clients and the client is a computer that is connected to the network and each room has a master client (aka the host). The host is the player who created the room. The host controls the game and if the host exits the game, the game ends. The host will also spawn the health packs, viruses and labs and ammo packs.

5.2 Scenes

The three scenes used for this project are described below.

5.2.1 Login Scene

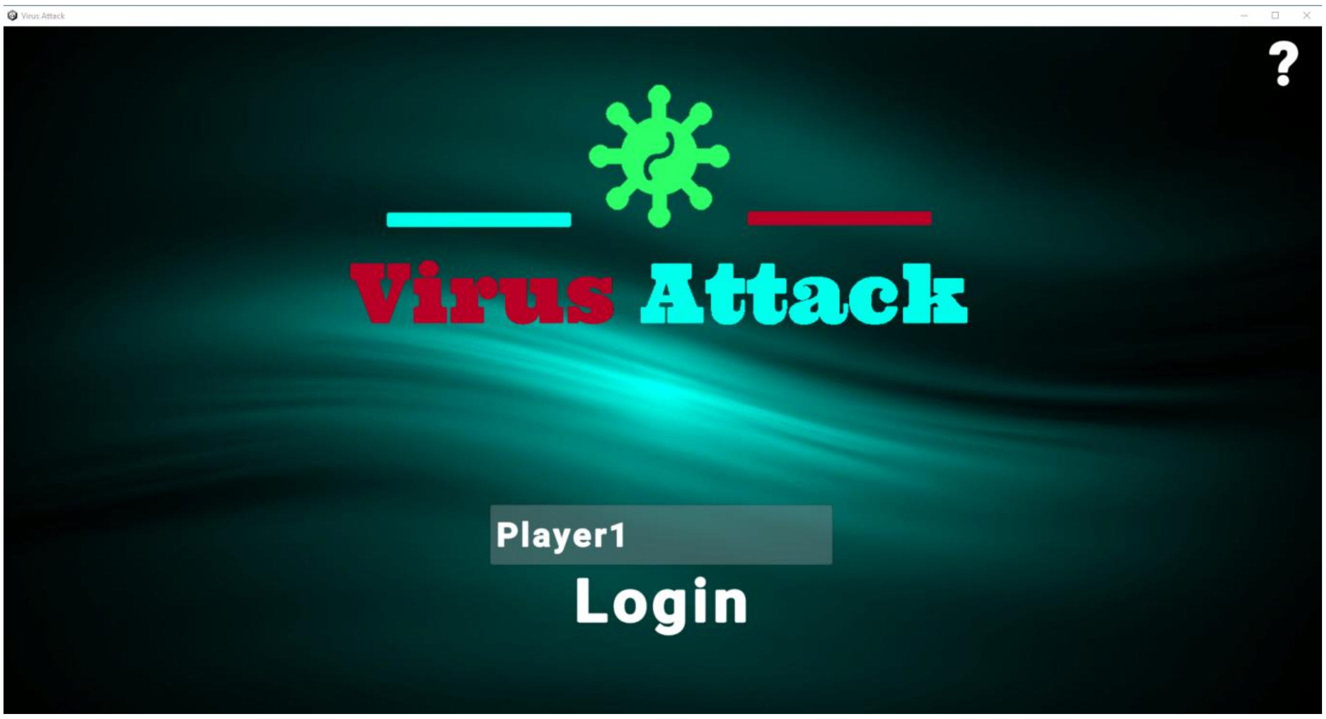


Figure 7: Login Panel

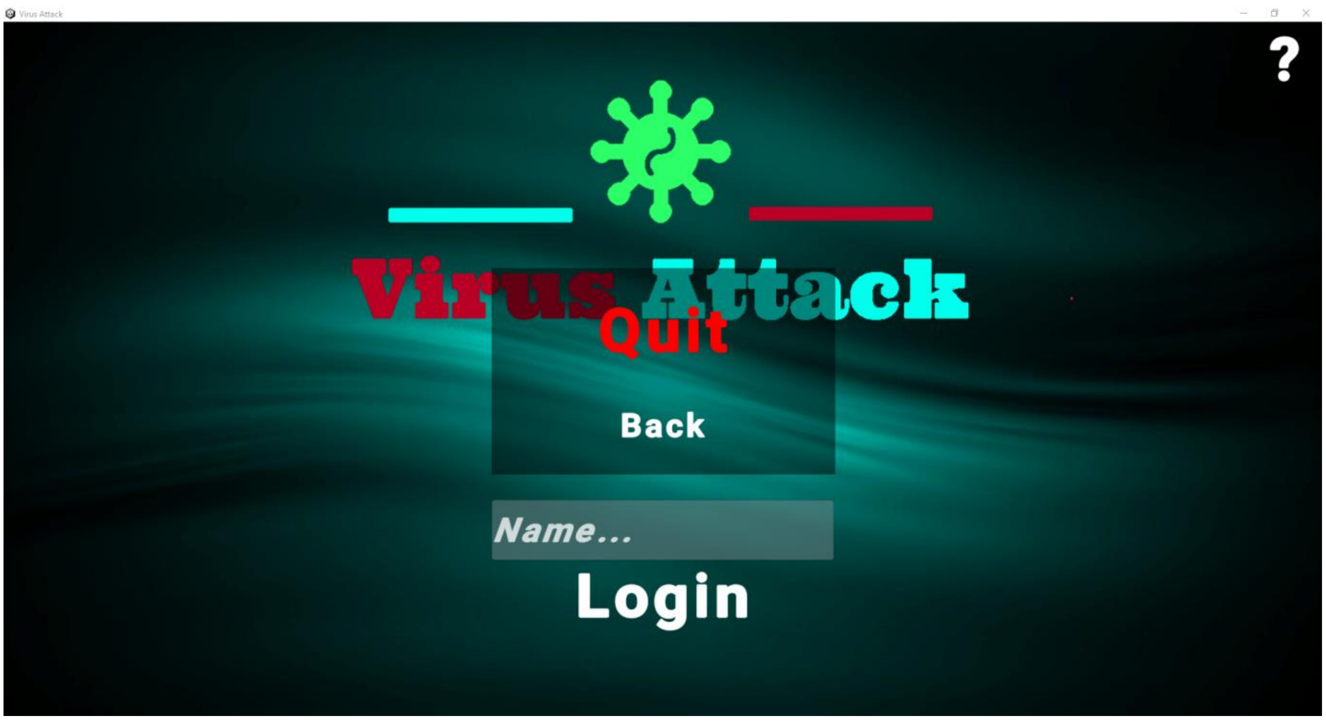


Figure 8: Quit Game Panel

The Login scene has:

Main Camera

Is the default camera for this scene which enables the players to view the scene.

Canvas

This screen size is set to 1920 x 1080 and has the following:

- Background Image.
- Login Panel.
  - A message which is displayed when the name entered is empty or too long.
  - The logo of the game.
  - Input field for the player to enter their name.
  - Login button which enables them to load the Lobby scene.
  - Help button displayed on the top right corner which activates the Help Panel.
- Connecting Panel.
  - See Appendix 5A.
- Help Panel.
  - See Appendix 5A.
- Quit Game Panel (Esc key pressed).
  - Quit button which enables the player to quit the application.
  - Back button which closes the Quit Game Panel.

5.2.2 Lobby Scene

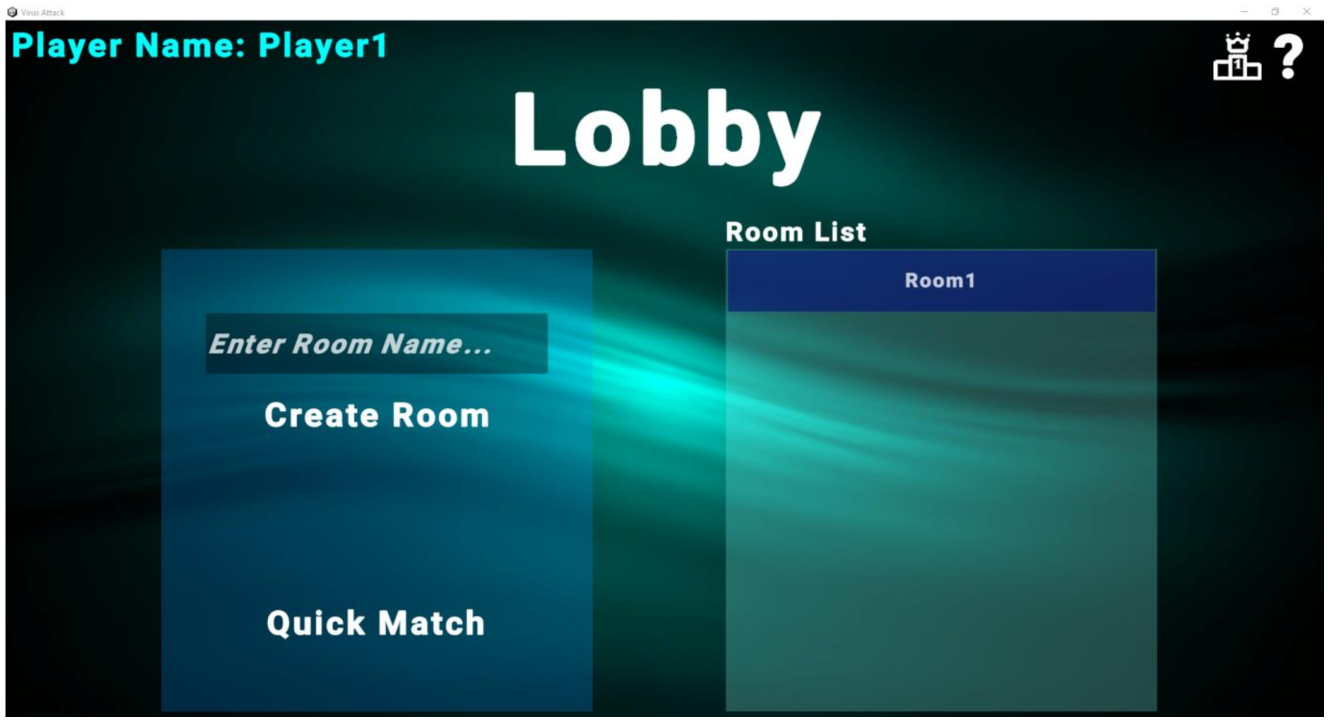


Figure 9: Lobby Panel

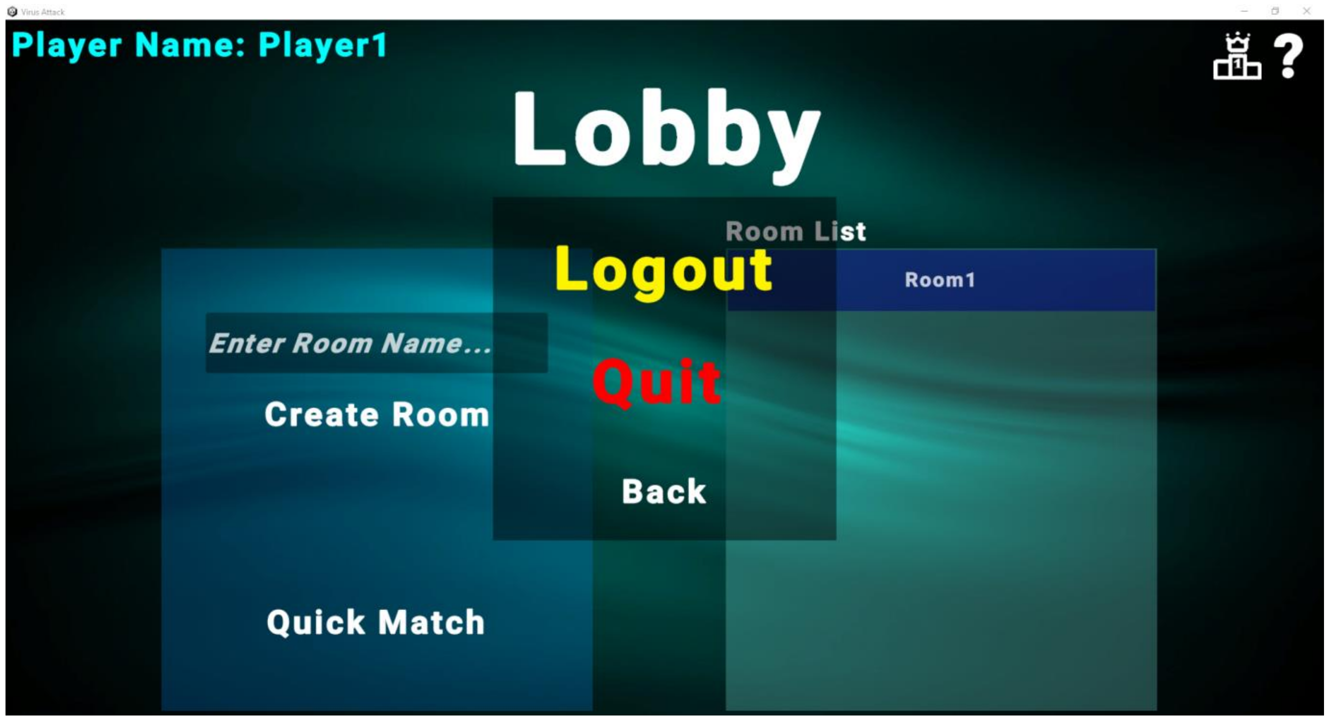


Figure 10: Lobby Quit/Logout Panel

The Lobby Scene has:

Main Camera

Is the default camera for this scene which enables the players to view the scene.

Canvas

This screen size is set to 1920 x 1080 and has the following:

- Background Image.
- Player name (from Login scene) displayed on the top left corner of the screen.
- Lobby Manager.
    - Lobby Panel which consists of multiple panels that make up the Lobby screen.
    - Connecting Panel: See Appendix 5A.
    - Room Panel which consists of multiple buttons and spawn points that make up the Room screen. There is more information below.
    - Help Panel: See Appendix 5A.
    - Logout/Quit Game Panel (Esc key pressed).
        - Logout button which enables the player to disconnect from the Photon Master Server.
        - Quit button which enables the player to quit the application.
        - Back button which closes the Logout/Quit Game Panel.

Room Panel

Once a room has been selected/created, the Room screen is displayed. If the player created a room, the screen, figure This screen shows the room that enables players to join and play.

- Title of screen is the Room name.
- Player name is in the top left corner of screen.
- Leader board Icon on top right corner of screen.
- Help Icon on top right corner of screen.
- The Leave button takes the player back to the Lobby screen.
- The Play button starts the game which starts the game at level one.

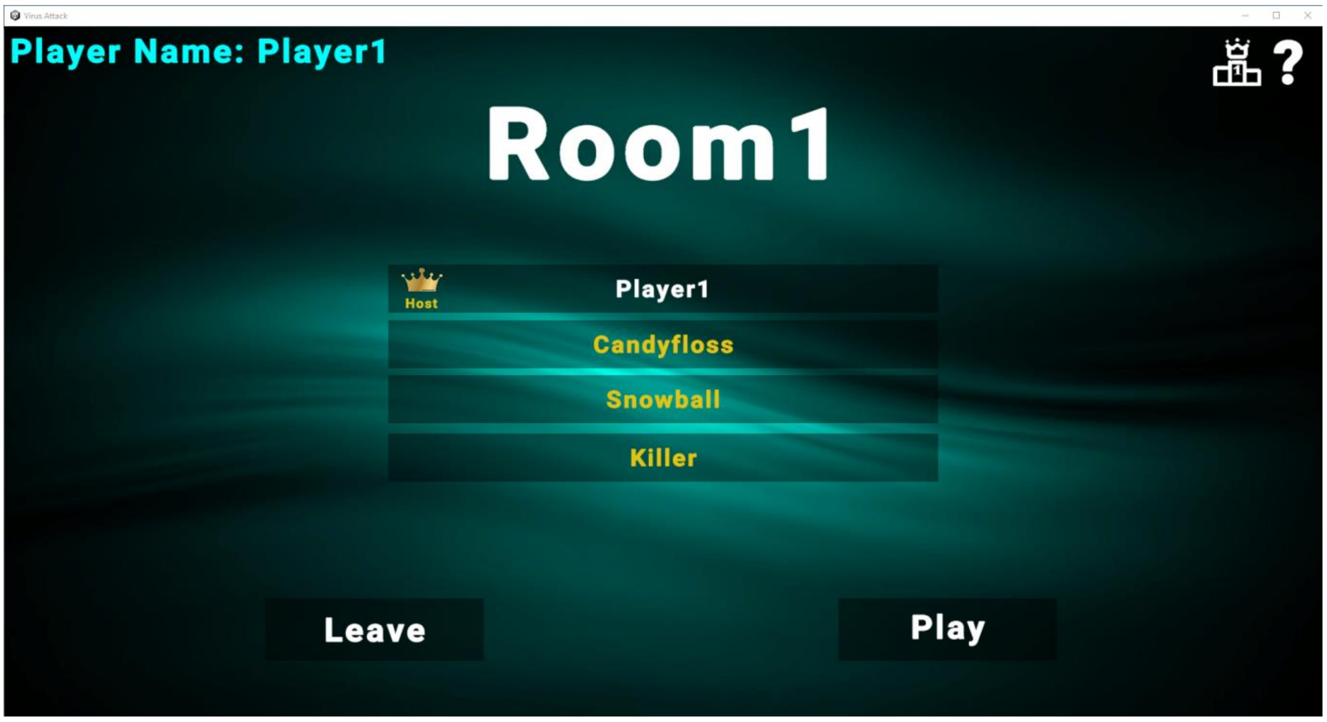


Figure 11: Room Create Room Panel

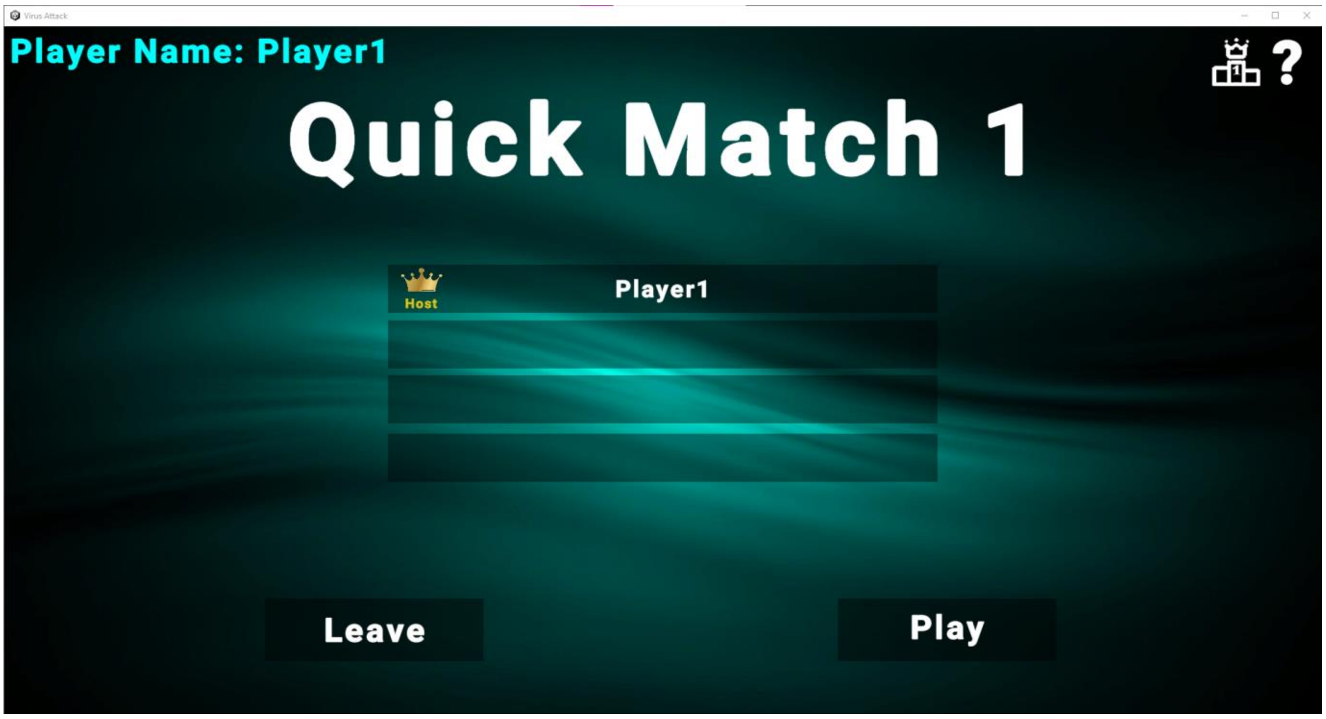


Figure 12: Room Quick Match Panel

The LobbyManager and RoomItem scripts are attached to this scene.

The LobbyManager script:

- ❖ Enables player to join the lobby.
- ❖ If player is not connected to the Photon Master Server, the player is taken back to Login screen.

- ❖ If player does not input a room name, they are prompted to do so, and a new room is created, and the room screen is loaded.
- ❖ This screen shows the player name, room name and the play button.
- ❖ When the 'Quick Match' button is selected, the player joins a random room, and the room screen is displayed.
- ❖ The player leaving the room and the disconnection to the master server are handled here.
- ❖ Once the room has been selected/created and the Play button is clicked the game begins.

5.2.3 Game Scene

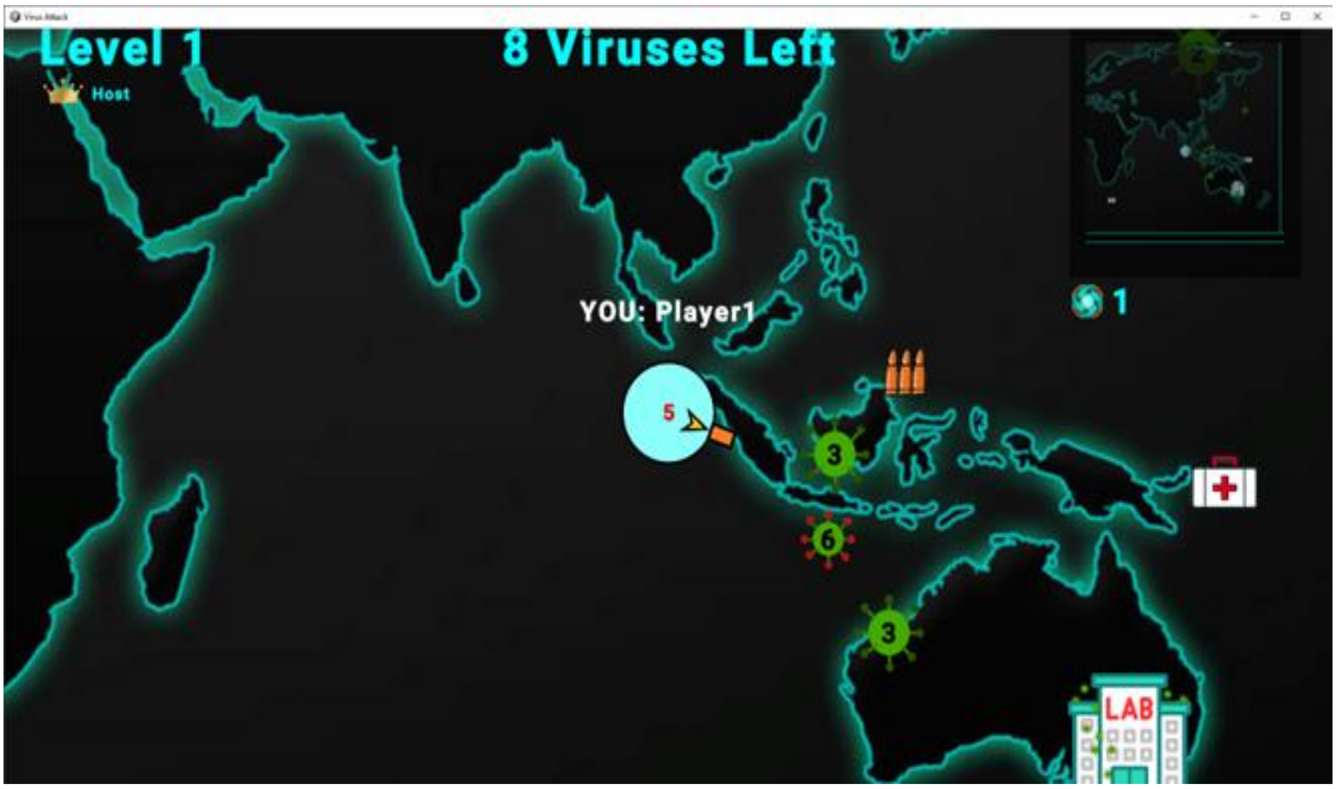


Figure 13: Game Scene

CanvasPivotPoint Object

Is the parent object of all objects in the Game Scene.

Main Camera

Is the default camera for this scene.

Canvas

This screen size is set to 1920 x 1080 and has the following:

- ❖ Title of screen, Lobby.

- GameManagerObject sets the variables for the Game_Manager script.
- NetworkManager object, calls the NetworkManager script.
- GameInfo, calls the GameInfo script.
- Player image and prefab asset.
- Lab image and Texture 2D.
- PlayerBullet image and Texture 2D.
- Virus image and Texture 2D.
- Weapon image Texture 2D.

The following scripts drive the main engine of the game and sets the parameters for the game play. These scripts are stored different folders.

Game Folder

- AmmoPack.cs – Detects the whether the ammo pack has collided with a player or a bullet and if that is true, the ammo pack object is destroyed over the Photon Network.
- Bullet.cs – Handles the collision detection of the bullet with other objects. It gets destroyed over the network unless it is colliding with the player who spawned it or another bullet.
- CharacterState.cs – Has an enumerated value that sets the status of the player or the virus to either 0 (Alive) or 1 (Eliminated).
- DirectionHandling.cs - Calculates the vector direction from the angle that the object is travelling in. It also calculates the rebound directions when the virus collides with the border.
- EscapeCanvasHandling.cs - Displays the menu when the Esc key is pressed mid game. It also handles when one of the buttons is clicked.
- FreezeRotation.cs – Freezes the camera rotation of canvases mid game, such as the Level Completed canvas.
- Game_Manager.cs – This class sets up the game and manages the objects in the scene over the network: -
    - Manages all players in the room: the spawning, health, and death of the players, updates the players killed counter
    - Manages labs, and the spawning of the viruses.

    - Updates viruses killed counter.
    - Spawns health packs.
    - Sets the next game level.
- GameInfo.cs – Receives updated values from the GameManager class and synchronises these values to the other players' GameInfo class in the game. It also checks if a player who has recently been eliminated has taken first place in the leader board. If that is true, it updates the leader board.
- HealthPack.cs – Handles the collisions between the health pack and either the player or bullet.
- Hit.cs – Destroys the text object after 5 seconds.
- Lab.cs – Handles the direction and number of viruses spawning from the lab.
- Network_Manager.cs – Detects if another player has left the game and calls a function from the Game Manager class.
- ObjectFace.cs – Stores the enumeration values of all the faces of the object: Top, Right, Bottom, Left
- Player.cs – Handles the player initiation as well as the health of the player. It also checks for any player input: - use of 'Z' key to move player, spawning bullets and initiating map mode. It also handles when the game is moving on to next level and activating appropriate canvases when necessary.
- PlayerMovement.cs - Handles the player movement and rotation.
- PlayerWeapon.cs - Changes the name of the weapon to identify that it is associated with the player.
- UICanvasHandling.cs – Sets up the player UI and updates the values retrieved from the GameInfo class.
- Virus.cs – Handles the movement, collisions, settings, and the pathfinding mode for the object of the Virus.

Lobby Folder

- LeaderboardPanel.cs – Displays all the Leader board information in the Leader board screen.
- LobbyManager.cs – Provides the functionality to all buttons in the Lobby Scene and handles when the player creates or joins or leaves a room. It also updates the room list in the lobby.

- PanelHandling.cs – Handles panels activation and deactivation when the user is switching screens within the Lobby Scene.
- RoomItem.cs – Provides functionality to the room button in the lobby.
- RoomList.cs – Updates the room list in the room screen when a player is joining or leaving the room.

Login Folder

- LoginManager.cs – Provides functionality to all objects that are in the Login Scene.

Universal Folder

- DisplayPlayerName.cs – Displays the name of the player at the top left corner of the screen in the lobby and room screens.
- Leaderboard.cs – Reads scores from the text files and overwrites new scores to the text files. Controls and manages the Leader board information stored for the server.

Level Completed Canvas

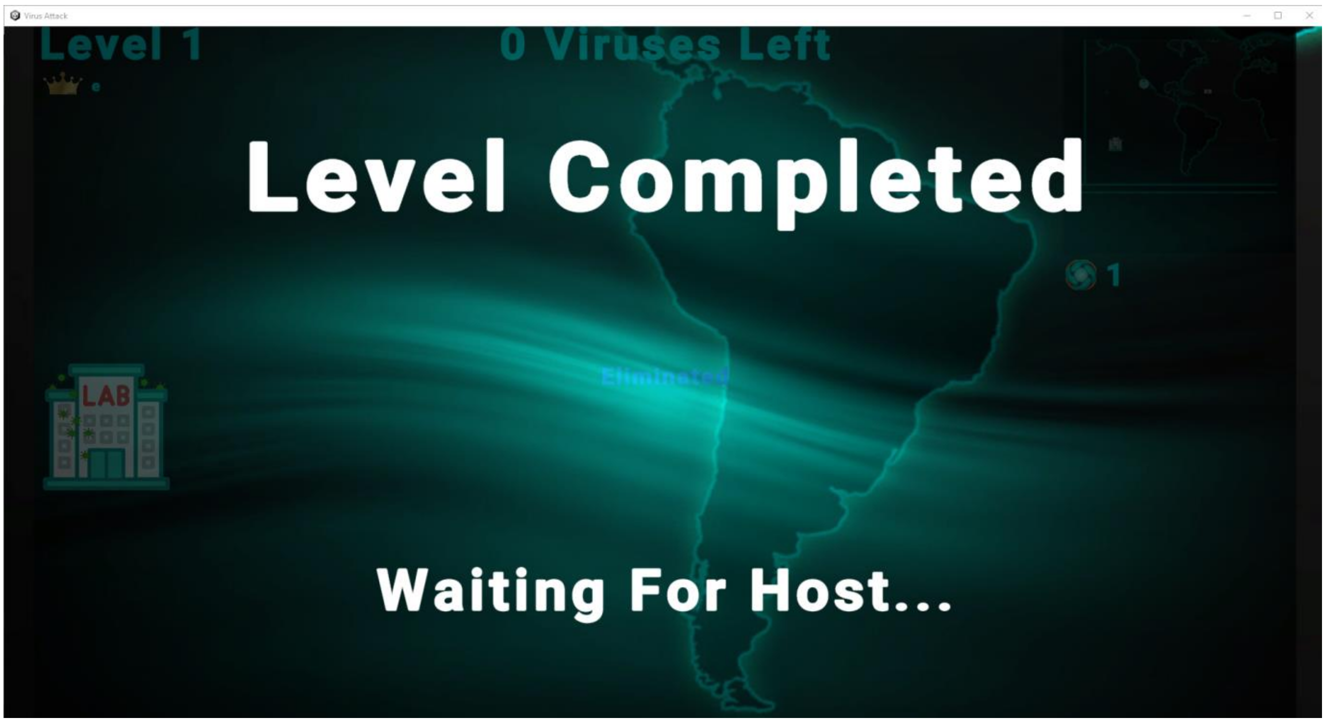


Figure 14: Level Completed Client Screen

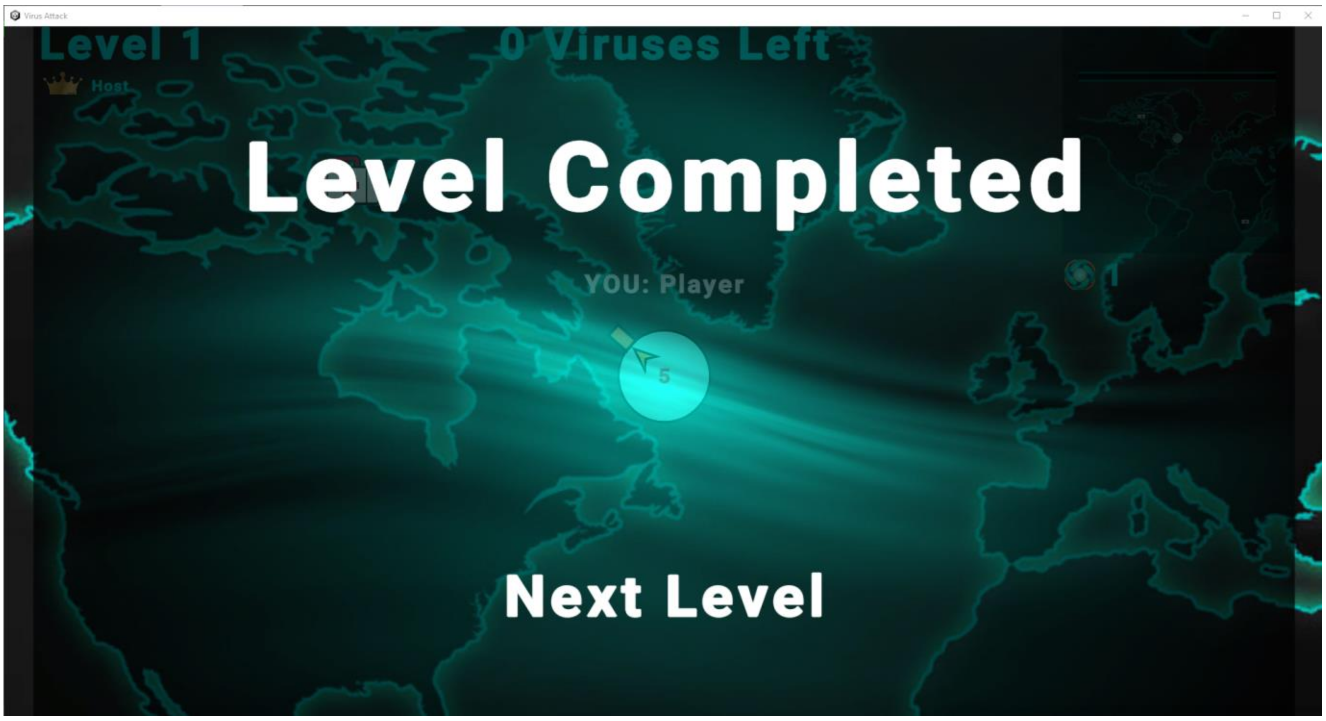


Figure 15: Level Completed Host Screen

When all the viruses are killed, the level completed screen is displayed. For the host, 'Exit to Lobby' button is displayed whereas for the client the 'Waiting For Host...' message is displayed. Once the host clicks the button, the game will move on the next level.

Canvas Object

This screen size is set to 1920 x 1080 and has the following:

- Title of screen, Level Completed.
- Background.
- Player name (got from Login scene), displayed on the top left corner of the screen.
- Game Level.
- No of Viruses left.
- Next Level button is displayed.

Game Over Canvas

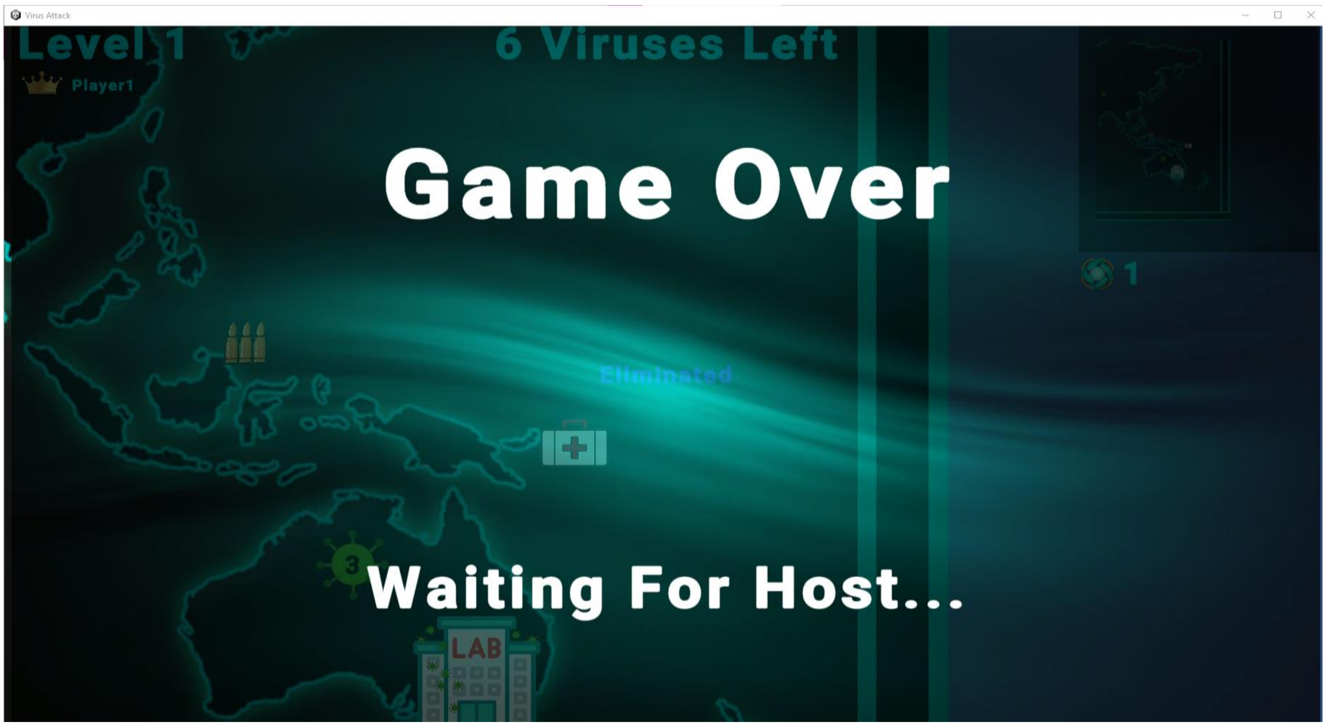


Figure 16: Game Over Client Screen

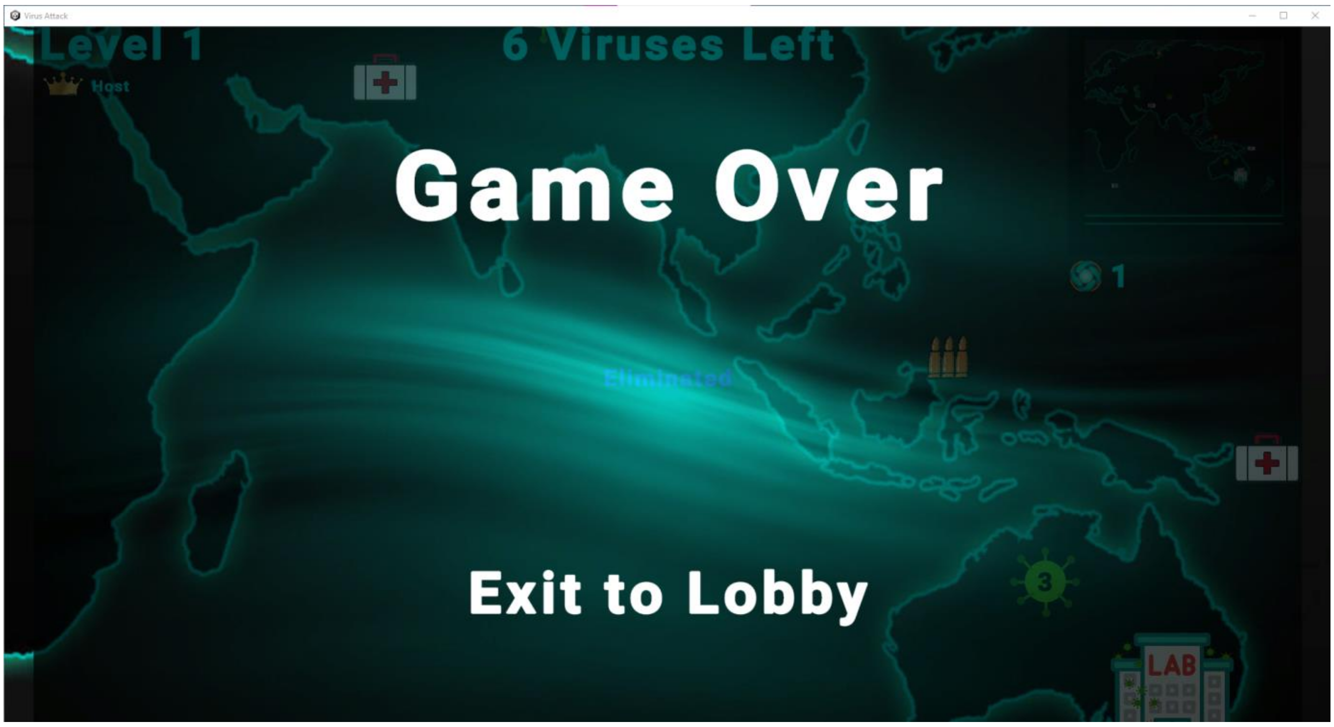


Figure 17: Game Over Host Screen

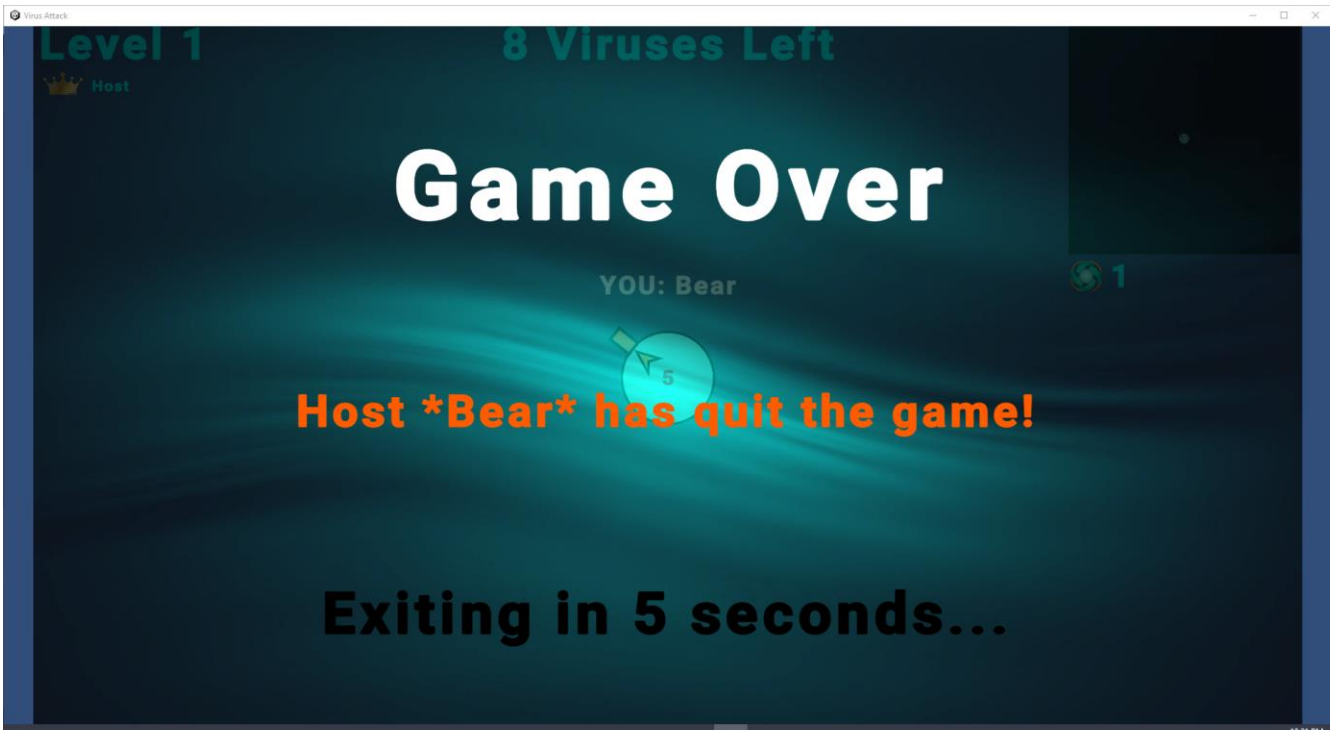
When all the players are killed, the game over screen is displayed. For the host, 'Exit to Lobby' button is displayed whereas for the client the 'Waiting For Host…' message is displayed. Once the host clicks the button, all players will go back to the lobby and leave the room.

Canvas Object

This screen size is set to 1920 x 1080 and has the following:

- Title of screen, Game Over.
- Background.
- Player name (got from Login scene), displayed on the top left corner of the screen.
- Game Level.
- No of Viruses left.
- Exit to Lobby message is displayed.

Host Quits Game Canvas


Level 1

8 Viruses Left

Game Over

YOU: Bear

Host *Bear* has quit the game!

Exiting in 5 seconds...

Figure 18: Host Quits Game

This screen will appear when the host decides to quit the game. A message is displayed to let other players know the situation. The players automatically go back to the lobby after five seconds.

Canvas Object

This screen size is set to 1920 x 1080 and has the following:

- Title of screen, Game Over.
- Background.
- Player name (got from Login scene), displayed on the top left corner of the screen.

- Game Level.
- No of Viruses left.
- Exiting message is displayed.

6. Testing and Results

Game testing is an integral part of the software development lifecycle. The code that is generated is tested during compile time and any errors must be corrected before it can be executed. As this was a multiplayer networked player game, it was more challenging to test the players simultaneously, on different windows and/or PCs using the normal methods of testing.

The game was tested as a standalone game to check that one player could kill the viruses that were spawning and ensuring the health of the player and viruses was being reduced appropriately. This was also checked for the viruses attacking the player and monitoring the health values of both player and viruses. The number of labs, viruses, health packs were tested to ensure that they were being spawned correctly.

The game was then tested as a two-player game by building the game and running it on two separate machines. The number of viruses spawned from the labs from reduced to two viruses initially and this was gradually increased. The syncing of the objects and their values was checked to ensure that it was consistent with what was expected. The full scope of the game is to allow between one to four players to play over the internet. Final testing was to ensure that the complete gameplay worked as expected for all scenarios having between one and four players logged in and playing together.

The mini map was checked to see that the map view of the players and other objects on the mini map were in the same location on the main map.

There are many levels of testing that were used in the development of this game. The aim is to ensure that Virus Attack is an error free game. The testing is divided into three main phases:

- ❖ **Module Testing** – This was performed during the coding by using debug messages to check that the written code produces the desired results. The requirement is that the code will compile with no bugs.
- ❖ **Integration Testing** – This was performed to ensure the different elements of the code work well together. This was done by building the project and creating executable files and running on the same PC and on another machine to test two online players.

- ❖ **Functional Testing** – This was performed after the game had reached its final version. During this testing phase, the game was tested to see if it meets the requirements.
- ❖ **Usability Testing** – This test was performed to see how easy it was to learn to play the game. The testing was performed by friends and family.

6.1 Unit Testing

During the game development I used Unity's debugger and console to monitor variables and objects to ensure the correct result was being derived.

The Debug.Log and Debug.LogError functions were used to check the values. This is a type of black box testing where the behaviour of the system is determined by looking at the input and output values.

The built-in debugger in Visual Studio was used for code that was being run on the host machine but could not access the code that was run on the client server. For this, the Debug.LogError function was helpful to check the values of the host and client.

Pathfinding- there is AI coded in the game where the virus will follow the player when they are at a certain distance from them. This was a challenging aspect of the game as the viruses were often seen hovering hear the player and became increasing difficult to kill as the game level increased and more viruses were introduced in the game.

Viruses – The number of the viruses in the screen was reduced to two to enable to test the health of the virus when colliding only with a player. The health was checked to see that a collision resulted in the health going down.

6.2 User Testing

Being the developer of the game, it is important to get other people to test the game as some things can be easily missed. I created a questionnaire which I gave to family and friends to give me feedback. The table below shows the questions that they answered:

Questions	Yes	No	Comments
1. Was it easy to start the game?			
2. Did you understand the instructions of gameplay?			
3. Were the graphics appealing?			
4. Did you find the game challenging?			
5. Did you like the concept?			
6. Do you think audio is an important feature to implement?			
7. Did you enjoy playing the game?			
8. Did you feel motivated to continue playing the game?			
9. Did your game crash?			
10. Did you experience any bugs/errors when playing the game?			
11. Do you play PC games?			

Figure 19: Questionnaire Questions

The demographic of users ranged from novices to hardcore players and the results of the questionnaire are in Appendix A7. The results show that novice players found the game challenging to play as they are not used to keyboard controls and were eliminated from the game much earlier on. The more experienced players initially thought that a 2D game would be much easier to win but as this game gets more challenging with the game levels and number and speed of viruses, they found that they wanted to keep on playing to beat the high scores. Generally the User Interface of the game was liked, and it was easy to navigate through the menus and simplicity of the design was liked.

Overall, the feedback was positive but a few features for future enhancements were highlighted which would have been outside the scope of this project due to time constraints.

6.3 Test Cases

The table below shows all of the test cases.

Section	**Test/Visual Input**	**Expected Result**	**Result as expected**	**Comments**
1. Launching the game (Standalone mode)	Double click on virusattack.exe to load the game	Login Screen is displayed	Yes	
2. Screen Controls	Maximised window loads (1920x1080)			
	Screen can be minimised/maximised using the buttons on the top right of the screen			
	Screen has exit button that closes the window			
3. Login Screen	Do test section 2. Screen controls test		Yes	
	Click 'Login'	'Please enter your name' text displayed	Yes	
	Enter username and click Login	Lobby Screen displayed	Yes	
	Enter username and press Enter	Lobby Screen displayed	Yes	
4. Lobby Screen	Do test section 2. Screen controls test		Yes	
	Player name is displayed on top left corner of window		Yes	
	'Create Room' clicked	'Please enter a room name' text displayed	Yes	
	Enter Room name and click 'Create Room'	Room screen displayed	Yes	
	Click on 'Quick Match'	Room screen displayed	Yes	
Leaderboard Screen	Click Leaderboard Icon	Leaderboard screen displayed with four entries	Yes	
Leaderboard Screen	Click Back button	Control returns to Lobby	Yes	

Help Screen	Click "?" Help Icon	Help Guide Screen displayed	Yes	
Help Screen	Click Back button	Control returns to Lobby	Yes	
	Press Esc Key	Pop up menu displayed with Logout, Quit, Back	Yes	
Pop up menu	Click on Logout button	Control returns to Login screen	Yes	
Pop up menu	Click on Quit button	Game Exits	Yes	
Pop up menu	Click on Back button	Control returns to Lobby screen	Yes	
5. Room Screen	Do test section 2. Screen controls test		Yes	
	Check Player name is displayed on top left corner	Player name is on top left corner	Yes	
	Room name is displayed	Room name is in the middle of the screen	Yes	
		Host Player: Player name is displayed on line 1 with crown symbol and 'Host'	Yes	
		Client Player: Player name is displayed on line 2, 3 or 4	Yes	
	Clicking on 'Leave' button goes to Lobby screen	Leave button goes to Lobby screen	Yes	
Leaderboard Screen	Click Leaderboard Icon	Leaderboard screen displayed with four entries	Yes	
Leaderboard Screen	Click Back button	Control returns to Room	Yes	
Help Screen	Click "?" Help Icon	Help Guide Screen displayed	Yes	
Help Screen	Click Back button	Control returns to Room	Yes	
	Press Esc Key	Pop up menu displayed with Logout, Quit, Back	Yes	

Pop up menu	Click on Logout button	Control returns to Login screen	Yes	
Pop up menu	Click on Quit button	Game Exits	Yes	
Pop up menu	Click on Back button	Control returns to Lobby screen	Yes	
	Host enters room Client enters room Host leaves room	Client becomes Host on client screen	Yes	
	Host enters room Client enters room Client leaves room	Only Host remains on host screen	Yes	
	Host enters room Client1 enters room Client2 enters room Host leaves room	Client1 becomes Host on both Client1 and Client2 screens	Yes	
	Client1 leaves room	Client2 becomes host	Yes	
	Click on Play button	Play begins in the Game scene	Yes	
6. Game Scene (UI display)	Do test section 2. Screen controls test		Yes	
	Virus count is displayed on top middle of screen			
	Own player name visible above own player avatar with a YOU: prefix		Yes	
	Other player avatars visible with their player name above their avatar		Yes	
	Player health on displayed in player avatar		Yes	
	Virus with health count in the virus avatar		Yes	
	Two Labs visible at random positions on screen at start of game		Yes	
	Health packs visible at random positions on screen		Yes	

	Ammo packs displayed on screen		Yes	
	Multi-hit bullet icon displayed on screen with count of number of multi-hit bullets available to player		Yes	
	Mini map is displayed on top right corner of screen	Mini map displays all players, labs, viruses, health packs, ammo packs and all bullets which are present in the game within view	Yes	The map displayed is only a portion of the arena
	Press 'M' key on keyboard to show 'Map View' of game	Map displays all players, labs, viruses, health packs, ammo packs and all bullets which are present in the game within view	Yes	The map displayed is the whole of the arena
	Press 'M' or TAB key on keyboard again to return to 'Player View' of game		Yes	
	Press TAB key on keyboard to show 'Map View' of game	Map displays all players, labs, viruses, health packs, ammo packs and all bullets which are present in the game within view	Yes	The map displayed is the whole of the arena
	Press 'M' or TAB key on keyboard again to return to 'Player View' of game		Yes	
	Press Esc Key	Pop up menu displayed with Help, Exit, Back	Yes	
Pop up Menu	Press Help button	Help Guide Screen displayed	Yes	

Help Screen	Click Back button	Control returns to Room	Yes	
Pop up menu	Click on Exit button	Control returns to Lobby screen	Yes	
Pop up menu	Click on Back button	Control returns to Game screen	Yes	
6.1 Player Movement	Player rotates in direction of the mouse pointer		Yes	
	Player is shooting a standard bullet when left mouse clicked		Yes	
	Player is shooting a multi-hit bullet when right mouse clicked		Yes	
	Player is shooting a multi-hit bullet when space bar pressed		Yes	
	Player is moving in the correct directions when Arrow keys presses		Yes	
	Player is moving in the upwards when 'W' key presses		Yes	
	Player is moving towards the left when 'A' key presses		Yes	
	Player is moving downwards when 'S' key pressed		Yes	
	Player is moving towards the right when 'D' key pressed		Yes	
	Player is moving in the northeast direction when up and right arrow keys are pressed simultaneously		Yes	
	Player is moving in the northwest direction when up and left arrow keys are pressed simultaneously		Yes	

	Player is moving in the southeast direction when down and right arrow keys are pressed simultaneously		Yes	
	Player is moving in the northwest direction when down and left arrow keys are pressed simultaneously		Yes	
	Player is moving in direction they are facing when 'Z' key pressed		Yes	
6.2 Standalone player-only Host logged in	Check that there is only one player on the Game arena		Yes	
	Check only one player on the mini map		Yes	
	Check number of viruses remaining is eight at start of game		Yes	
	Check Health of player is five at start of game		Yes	
	Check player health does not get affected when they collide with the Labs		Yes	
	Check player health increases when they collide with the Health packs.	Levels 1,2, max health = 5 Levels 3, 4, max health = 8 Level above 5, max health = 10	Yes	
	Move player to all four screen boundaries and check player does not go into/through them and health remains the same	Colliders stop player from going past the screen boundaries	Yes	
	Check health of player decrements correctly	Virus 1 inflicts damage 1	Yes	

	when colliding with the viruses	Virus 2 inflicts damage 2 Amount of damage is displayed for a short period of time as "-1" or "-2" in the location where the player was when they were hit		
	Check health of virus decrements correctly when shot by the player	Standard bullet inflicts damage 1 Multi-hit bullet inflicts damage 3	Yes	
	Check the viruses are coming towards the player that they are closest to if they are within pathfinder range	If virus comes in range, then it will change direction directly towards the player	Yes	
	Check the viruses are moving in a random direction when out of pathfinder range	If virus goes out of range, then it will change direction to a random direction	Yes	
	Check virus health is zero before they 'die'		Yes	
	Check player health is zero before they 'die'		Yes	
	Count number of viruses killed by player and check number of remaining viruses is correct		Yes	
	If player kills all viruses, next game level option is displayed		Yes	
	If player health reduces to zero, then check that player eliminated message is displayed on screen and player avatar is no longer visible	Game over message is displayed with Exit to Lobby	Yes	

6.3 Multiplayer game mode	Load the game on two different machines	The game loads	Yes	
6.3.1 Lobby Screen (Host)	**Create a Host player** Do Test Section 3: Login Screen, on the first PC and create a room with 'Quick Match option'	New room created with a random name	Yes	
6.3.2 Room Screen (Host)	Room screen will have the host players name and room name		Yes	
6.3.3 Lobby Screen (Client)	**Create a Client player.** Do Test Section 3: Login Screen, on the second PC. Join the room created by the host by clicking on the room name in the room list or click on Quick Match button		Yes	
6.3.4 Room Screen (Client)	Room Screen will have the client players name		Yes	
6.4 Two players (starting the game)	Host clicks Play from the room screen. Game starts		Yes	
	Client clicks Play from the room scene, and joins the host in the same room		Yes	
	Synchronisation Check if both players can see each other and their names are correct		Yes	
	Synchronisation Repeat Test Section 5 (Room Screen) and 5.1 on both host and client player view game windows		Yes	
6.5 Two Players (check the host and client screens for each of the tests and ensure they are	Check both players are visible on the player view from their respective game windows		Yes	

synchronised in all respects)				
	Check both players are visible on the mini map from their respective game windows		Yes	
	Check number of viruses is eight		Yes	
	Check Health of both players is five		Yes	
	Check Health of all eight viruses is 3 before collisions between players and viruses		Yes	
	Check players health does not get affected when they collide with the Labs		Yes	
	Move player to all four screen boundaries and check player does not go into/through them and health remains the same		Yes	
	Check health of player decrements correctly when colliding with the viruses and is synchronised across both game windows		Yes	
	Check health of virus decrements correctly when colliding with the player		Yes	
	Check the viruses drawn towards the player that they are closest to		Yes	
	Check virus count is zero before they 'die'		Yes	
	Check player count is zero before they 'die'		Yes	
	Check how many viruses are killed by		Yes	

	player when they are shooting, and check count of remaining viruses is correct			
	If player kills all viruses, check virus count is zero, 'Level Completed' message is shown, 'Next Level' game is displayed. Check the player health is the same value when going to next level		Yes	
	Virus count on next level will be higher and the game was tested in each of the sections as described above		Yes	
	If player is killed by viruses, 'Game Over' message is displayed. Continue takes the player back to the lobby.		Yes	
	If one player leaves or dies, then the rest of the players can carry on playing.		Yes	
7. End Game	Game Over screen is displayed		Yes	
8. Disconnection due to Internet Issues	Switch off Internet connection	Check appropriate message is displayed when connection to internet fails	Yes	

Figure 20: Test Cases

7. Reflection

The development of this game was challenging but a rewarding process. Learning photon to use its network capabilities within the Unity3D game engine did not always produce the desired results. This was my first online multiplayer game and despite the lack of experience, I managed to develop a 2D game.

This project has provided me with invaluable experience in developing online games and this was of particular interest to me as it is an area I would like to progress in my career.

The challenges I faced in the project were mainly to do with the implementation of the game on the network. Sometimes the game would not synchronise properly, or the characters were not moving around on the network as expected, or network traffic was significantly heavier than anticipated due to the way network synchronisation is implemented. However, I overcame these by researching forums and managed to resolve the issues I was presented with. Synchronisation was corrected with changing the placement of network mirroring calls. Network traffic issues were corrected by analysing the frequency of network calls and making alterations to reduce unnecessary calls where they were not important; to implement a delay methodology to ensure that calls were not made too frequently; and setting Boolean flags to ensure that calls were only made when required.

I would have liked to implement more features, but the scope of improvements was too large to implement in their entirety, and the features I ultimately selected and finished the project with are sufficient to provide a fun and challenging game play experience whilst at the same time present a broad scope of challenges to overcome. The knowledge and experience that I have gained whilst completing this project is invaluable and will stand me in good stead in my future prospective career.

Something I learnt from this is that a good project plan is very important and writing well commented code will help understand what is happening in each section, and this knowledge and the code could possibly be re-used in other gaming projects.

Overall, I am satisfied with what I set out to do with this project and I have achieved the project brief and provided a game which has been tried and tested by my family and friends and has received excellent comments and reviews from them. As there is scope to add more features on this version of the game, and possibly look at presenting a 3D version as well, I intend to carry on working on the game after the project has been submitted, with a view to distributing the game to family and friends to play with and enjoy together.

Code Listings

The project code and executable can be accessed via the CSGitLab link:

HTTPS: https://csgitlab.reading.ac.uk/fe010336/final-year-project.git

SSH: git@csgitlab.reading.ac.uk:fe010336/final-year-project.git

Citation and References

Citations

(n.d.). Retrieved from Unity.com: https://unity.com/

Agario.io. (n.d.). Retrieved from CrazyGames: https://www.crazygames.com/game/agario

Crazygames. (n.d.). Retrieved from BlockTanks.io: https://www.crazygames.com/game/blocktanks-io

create.unity.com. (n.d.). Retrieved from Gaming Report: https://create.unity.com/gaming-report-2022

cybercrew. (n.d.). Retrieved from Cybercrew: 30+ Astonishing Video Games Sales Statistics UK Edition [2022] [2021] (cybercrew.uk)

Gaming Report. (n.d.). Retrieved from https://create.unity.com/gaming-report-2022

Unity. (2021). Retrieved from https://create.unity.com/2021-game-report

Unity. (2022). Retrieved from https://create.unity.com/2022-game-report

Wikipedia. (2022). *Wikipedia*. Retrieved from https://en.wikipedia.org/wiki/Unity_(game_engine)

Zomb Royale. (n.d.). Retrieved from Crazygames: https://www.crazygames.com/game/zombsroyaleio

References

Extreme Programming: Tips and Advantages - DZone Agile – For software development methodology.

https://wallpaperaccess.com/full/3004502.jpg – For the gaming arena background.

https://doc.photonengine.com/en-us/pun/v2/demos-and-tutorials/pun-basics-tutorial/intro

https://www.raywenderlich.com/1142814-introduction-to-multiplayer-games-with-unity-and-photon

https://unity.com/ - Unity.

https://www.photon.com – Photon.

https://www.udemy.com/course/introduction-to-game-development-with-unity/

https://www.youtube.com/watch?v=93SkbMpWCGo

https://www.youtube.com/playlist?list=PLhsVv9Uw1WzjI8fEBjBQpTyXNZ6Yp1ZLw

https://www.youtube.com/watch?v=833pOzSfEZE&list=PLgAF6rpCsTCh212dNwwXlRZvk8pnvhS8N

Appendices

A1 Glossary of Abbreviations

API	**A**pplication **P**rogramming **I**nterface
PC	**P**ersonal **C**omputer
PUN	**P**hoton **U**nity **N**etworking
MMO/ MMOG	**M**assively **M**ultiplayer **O**nline **G**ame
WASD	'**W**', '**A**', '**S**', '**D**' PC keyboard keys
2D	**2**-**D**imensional
SDK	**S**oftware **D**evelopment **K**it
HUD	**H**eads **U**p **D**isplay
AI	**A**rtificial **I**ntelligence
UI	**U**ser **I**nterface
GUI	**G**raphical **U**ser **I**nterface
RAM	**R**andom **A**ccess **M**emory
XP	**E**xtreme **P**rogramming
RPC	**R**emote **P**rocedure **C**alls
GPU	**G**raphics **P**rocessing **U**nit

A2 Project Plan

The image below shows the Gantt chart, displaying the project plan.

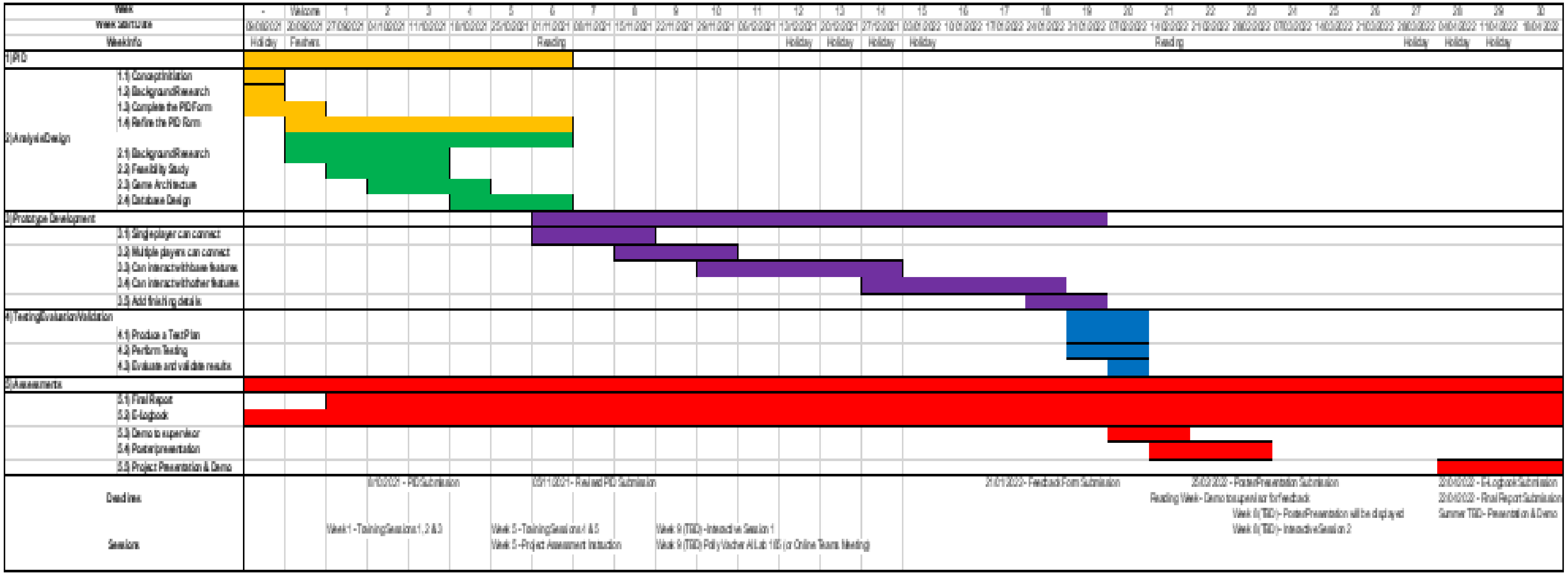


Figure 21: Gantt Chart Displaying Project Plan

To take a closer look at the chart, please access the document via the Microsoft OneDrive Link [Project Plan Final Version.xlsx]() or the CSGitLab Link

https://csgitlab.reading.ac.uk/fe010336/final-year-project.git

A3 Project Logbook

The table below shows the details of the meetings I had with my supervisor.

Date	Points Made	Plan For Next Meeting
09/08/2021	Discussed what projects I could potentially do. Looked through the PID.	Research and pick a project that I want to do.
23/08/2021	Defined the type of game that I am going to implement – MMORPG. Remark: MMORPG is still too complex for an individual project, note that those MMORPG that you see in the market are created by larger teams with years. For RPG, you'll need to handle player movement, player interaction with objects, items, inventory, equipment, level up, etc. For Multiplayer Online, you'll need to handle server setup, client-server connection, restricting players controls, synchronizing players movements, etc. For Massive, it would need extra resources for massive number of players, handle lagging and synchronization due to player connectivity, DDOS attack, cheating, etc. So, MMORPG would mean you'll need to do all the above which would be impossible. My suggestion is to narrow the project scope to either multiplayer game or RPG. Below are some references and tutorials: Multiplayer Game: • https://www.raywenderlich.com/1142814-introduction-to-multiplayer-games-with-unity-and-photon • https://www.youtube.com/watch?v=93SkbMpWCGo	Create an E-Logbook which is a table of 3 columns – date of meeting, points made in meeting, plan for next meeting. Look through the player reviews and find flaws in existing games. Fill in the Aims & Objectives section. Remark: For player reviews, aims & objectives, you may focus on multiplayer game or RPG instead of MMORPG. Brainstorm - what I am going to write in the Initial Project Specification.

	• https://www.youtube.com/playlist?list=PLhsVv9Uw1WzjI8fEBjBQpTyXNZ6Yp1ZLw • https://www.youtube.com/watch?v=833pOzSfEZE&list=PLgAF6rpCsTCh212dNwwXlRZvk8pnvhS8N • https://www.youtube.com/playlist?list=PL_eGgISVYZkd67NTyP7c4bjCZ7mdgk_9- RPG: • https://www.youtube.com/playlist?list=PLPV2KyIb3jR4KLGCCAciWQ5qHudKtYeP7 • https://www.youtube.com/watch?v=l4Hijtk4NDc&list=PLZ1b66Z1KFKgp-sjQ8ldU3eh8DoQ3a14P&index=1 • https://www.youtube.com/watch?v=oHFOkMffPDc&list=PLX-uZVK_0K_6JEecbu3Y-nVnANJznCzix To make the project manageable and able to be completed, you may consider starting with a multiplayer game and if the progress is good then later you may extend your game by adding some RPG elements. Discussed what I need to write in the Aims & Objectives section. Briefly looked at another section – Initial Project Specification. In this section will need to define the game engine, the game features, how many players will be supported, how many levels are there. The game will be an improvement on existing games so will need to do background research on player reviews.	
07/09/2021	Talked about how MMORPG can be quite complicated so choosing a new game would be best. Discussed options: Make a 3D version of an existing simple game and add features. Make a mobile version of an existing pc game and add features.	Choose a game. Draft the PID.

	Shooting and RPG are potential game types. The Escapist is okay, but it could end up being too complicated as there are a lot of features that would be needed to implement. Games in the tutorial videos, listed above, are suitable for a final year project.	
21/09/2021	Went through the draft of the PID and discussed what needed to be modified in each section. Talked about the possibility of using a game engine.	Complete the PID.
01/10/2021	Talked about the next steps.	Background research – Look for player reviews of similar existing games, compare the good and bad points and put them into paragraphs.
08/10/2021	Further discussion on player reviews.	Put down a few more points for the 3rd game. Put 2nd, 3rd game and your game in a table with some aspects to compare. Research the game development, game technologies, game platforms and different types of games.
15/10/2021	Briefly went through the player reviews and discussed the 3 limitations which will be needed to solve. Also talked about the technological and historical development of the game industry.	Conclude the pros and cons that are common across all 3 games. Write down 3 limitations which will be needed to solve. Capture screenshots of the UI. Write down the technological and historical development of the game industry.
22/10/2021	Went through the technological and historical development of the game industry that was listed.	Write down more about the technological development of the game industry.

	Talked about the next steps.	Research different gaming platforms and game engines and explain why Android and Unity 2D were chosen. Note down the references & citations.
16/11/2021	Went through the first draft of chapters 1 and 2.	Complete chapters 1 and 2. Create a single player connection to the world in Unity.
26/11/2021	Discussed the next steps.	Come up with the design of the world.
14/01/2022	Went through the progress that was made. Discussed the next step.	Implement the online aspect into the game.
28/01/2022	Went through the progress that was made. Discussed the next steps.	Integrate the online program into the game.
11/02/2022	Have shown the online aspect. Went through the progress that was made. Discussed the next steps.	Create the base game.
03/03/2022	Poster Presentation with Carmen and Miguel. After the presentation, have shown the progress that was made.	Continue creating base game.
01/04/2022	Shown the progress that was made. Not much progress was made as was focused on assignments. Mutually decided to stop weekly meetings.	Finish making the game.

A4 User Guide

User Guide

The player controls the game play using the keyboard keys and the mouse:

Direction keys

- 'W' or 'Up Arrow' key – Move the player up.
- 'S' or 'Down Arrow' key – Move the player down.
- 'A' or 'Left Arrow' key – Move the player left.
- 'D' or 'Right Arrow' key – Move the player right.
- 'Z' key – Follow the mouse position automatically.

Other Keys

- 'Esc' key – Terminate the game.
- 'M' key – Invoke main map mode and pressing 'M' key again to get back to player view.
- TAB key – Invoke main map mode and pressing 'M' key again to get back to player view.
- 'P' key – Pause the game.
- SPACE bar – Shoot a multi-hit bullet in the direction of the player weapon.
- Up arrow key – To move the player upwards.
- Down arrow key - To move the player downwards.
- Left arrow key – To move the player left.
- Right arrow key – To move the player right.
- Combination of up and left keys to go in the northwest direction.
- Combination of up and right keys to the northeast direction.
- Combination of down and keys left to the southwest.
- Combination of down and keys right southeast.

Mouse Movement – Rotate player weapon to point in the direction according to the mouse movement.

Left Mouse Click – Shoot a standard bullet in the direction of the player weapon.

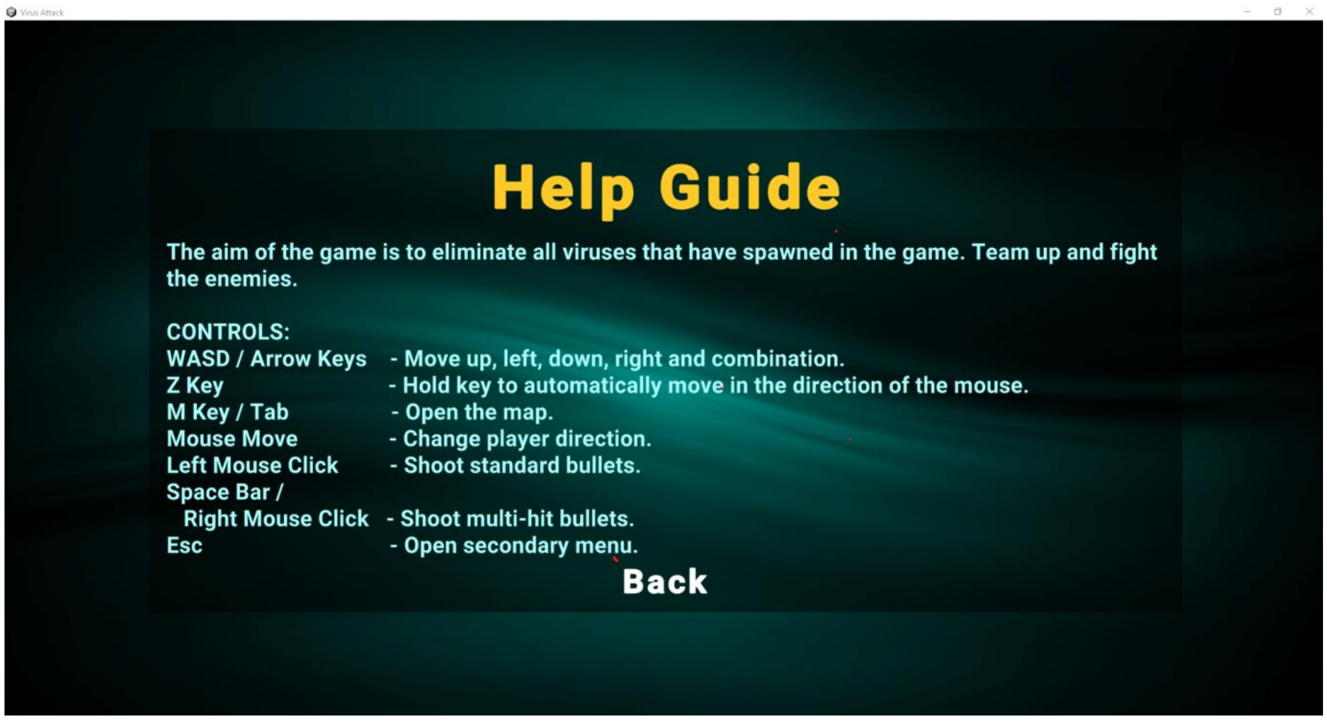
Right Mouse Click – Shoot a multi-hit bullet in the direction of the player weapon.

A5 Commonly Used Screens

Help Panel

The Help Panel consists of:

- Title of the panel.
- Text object which displays the description of the game as well as the controls.
- Back button which enables them to go back to the Login scene.



Help Guide

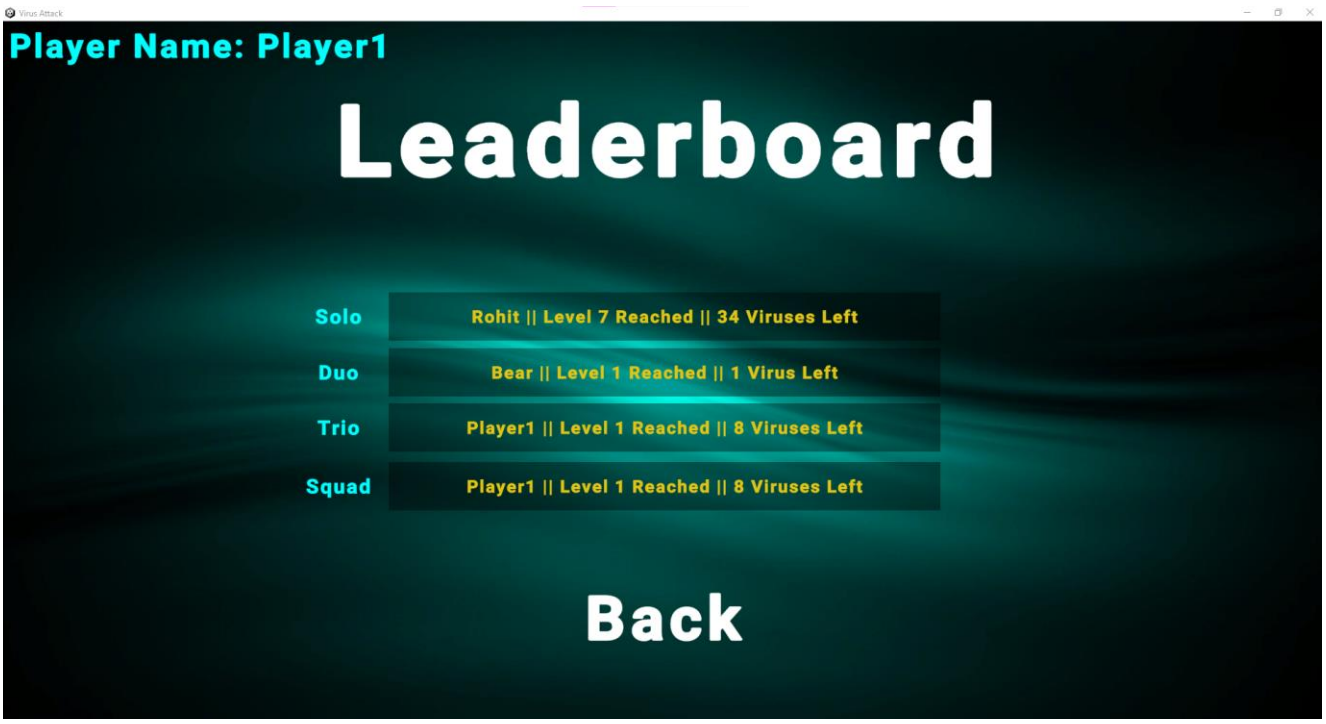
The aim of the game is to eliminate all viruses that have spawned in the game. Team up and fight the enemies.

```
CONTROLS:
WASD / Arrow Keys   - Move up, left, down, right and combination.
Z Key               - Hold key to automatically move in the direction of the mouse.
M Key / Tab         - Open the map.
Mouse Move          - Change player direction.
Left Mouse Click    - Shoot standard bullets.
Space Bar /
  Right Mouse Click - Shoot multi-hit bullets.
Esc                 - Open secondary menu.
```

Figure 22: Help Panel

Leaderboard Panel
The Leaderboard Panel consists of:

- ❖ Title of the panel.
- ❖ Player Name.
- ❖ The top scorers in one player game (solo), two player game (duo), three player game (trio), and four (squad) player game.



Solo	Rohit || Level 7 Reached || 34 Viruses Left
Duo	Bear || Level 1 Reached || 1 Virus Left
Trio	Player1 || Level 1 Reached || 8 Viruses Left
Squad	Player1 || Level 1 Reached || 8 Viruses Left

Figure 23: Leaderboard Panel

Connecting Panel

The Connecting Panel consists of:

- Text displaying “Connecting...” when another scene is loading.

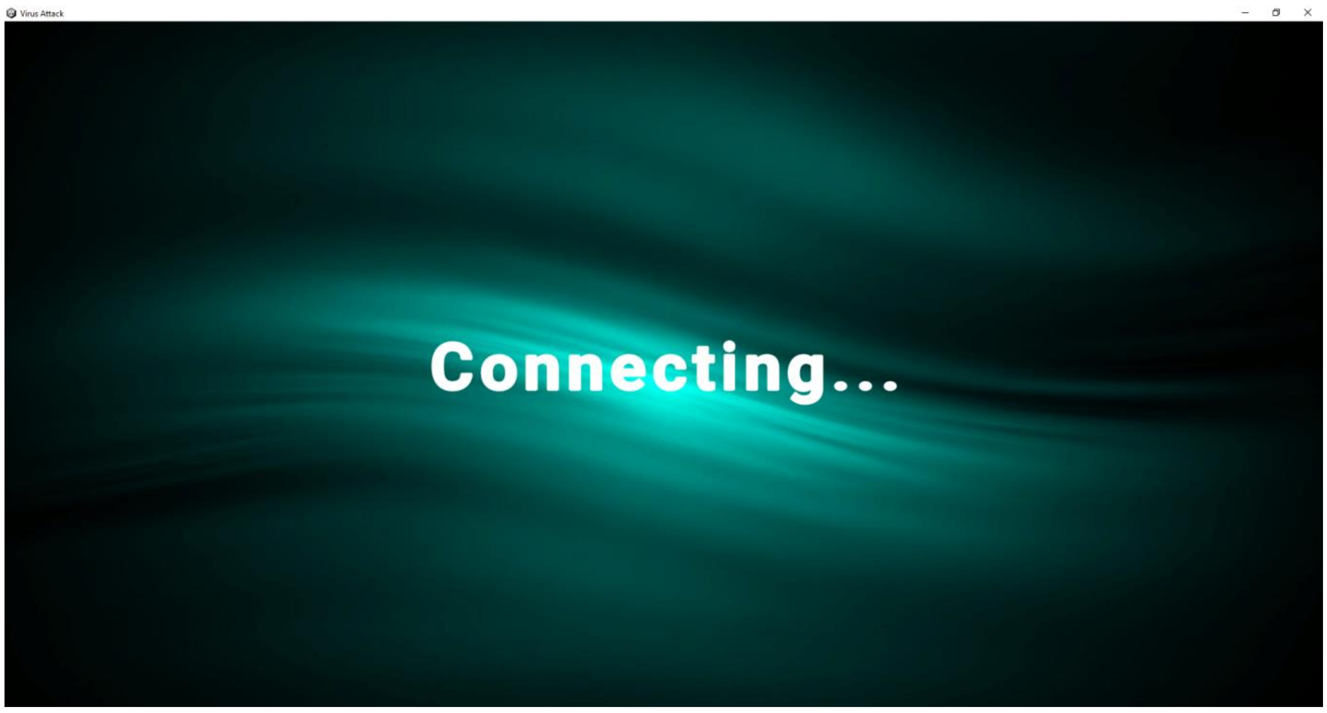


Figure 24: Connecting Panel

A6 Class Diagram

A class diagram shows the associations of the classes that are in the project.

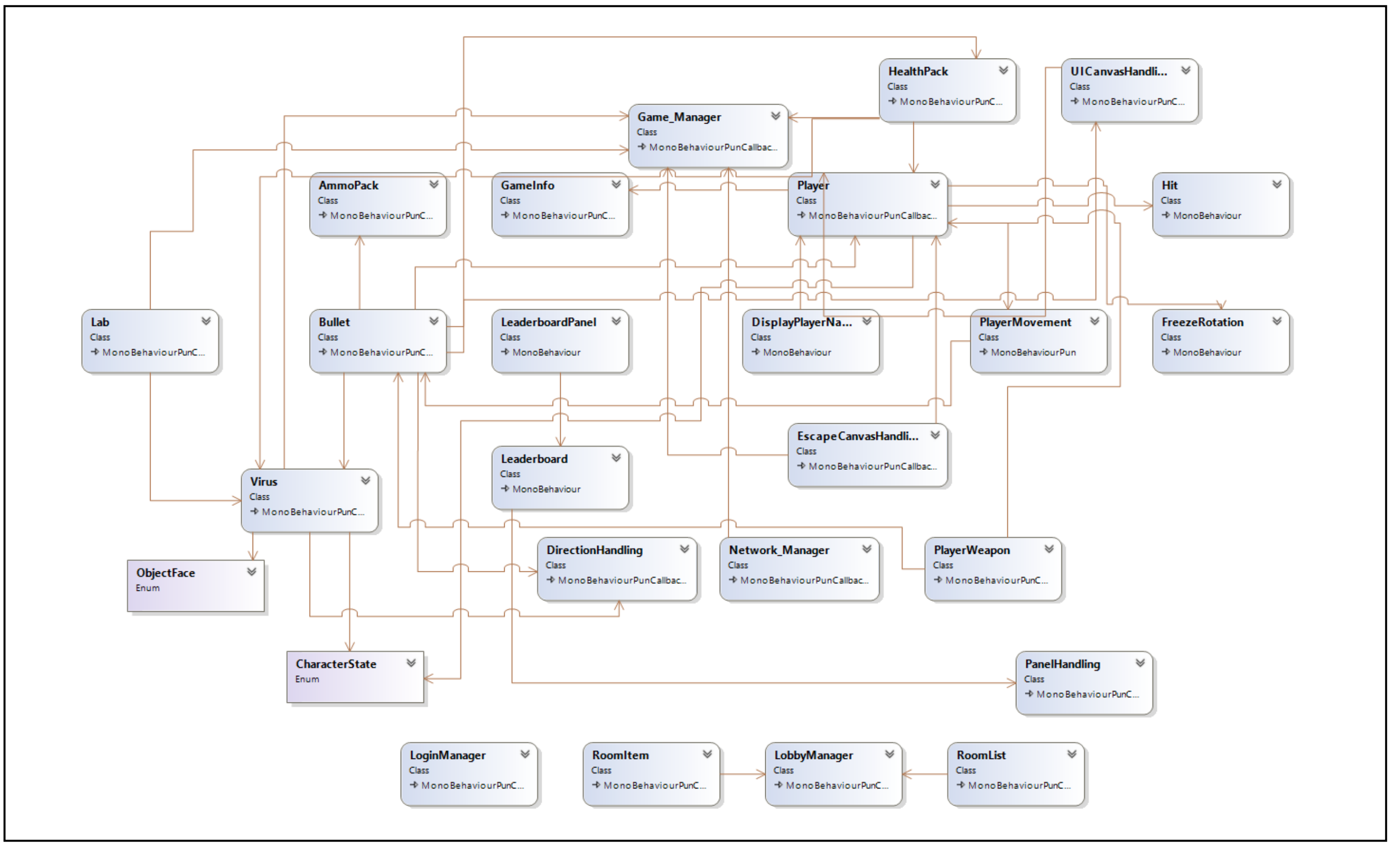


Figure 25: Class Diagram

A7 Questionnaire – Virus Attack Game

Please complete this questionnaire after you have played the game.

Name: ____________________

Please enter ticks in the appropriate sections of the table below with any comments that you wish to include:

Questions	Yes	No	Comments
1. Was it easy to start the game?			
2. Did you understand the instructions of gameplay?			
3. Were the graphics appealing?			
4. Did you find the game challenging?			
5. Did you like the concept?			
6. Do you think audio is an important feature to implement?			
7. Did you enjoy playing the game?			
8. Did you feel motivated to continue playing the game?			
9. Did your game crash?			
10. Did you experience any bugs/errors when playing the game?			
11. Do you play PC games?			

Do you have any recommendations you would like to make about this game?

Results of Questionnaire

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Comments
Player1	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes	Yes	Although game is a simple 2D game, found it challenging to kill all viruses. Wanted to carry on playing to beat high score
Player2	Yes	Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes	Yes	
Player3	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	
Player4	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	
Player5	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	No	Audio can be distracting during game play
Player6	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Found it too difficult to kill viruses
Player7	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Would like to be able to change the player avatar
Player8	Yes	Yes	Yes	Yes	Yes	No	Yes	No	No	No	No	Found the game challenging to play as novice player

Figure 26: Results of Questionnaire