

COMP4801 Final Year Project

A 3D Game to Raise the Teenagers' Awareness on Cybersecurity

FYP19040 Final Report

Supervisor: Dr. T. W. Chim

Member: Leung Ho Pong James (3035374840)

Date of Submission: 3rd May, 2020

Abstract

With the rapid increase in the numbers of cybercrime occurrences, educating the general public, especially the teenager group, about the concept of cybersecurity proves more important now than ever. In response to this need, this project proposes a 3D VR game to serve as an educational tool to raise the awareness of general public on cybercrimes. HTC Vive set will be used to provide the VR environment needed, and Unity game engine will be adopted to complete the 3D setting. The current project version supports smooth gameplay for players to experience captivating VR experience along the process of combatting different types of cybercriminals. In the future, more weapons and maps will be added to further enhance the game diversity and equip players with stronger cybersecurity knowledge.

Acknowledgement

I would like to express my deep gratitude to Dr. T. W. Chim, my project supervisor, for his enthusiastic assistance during the development of this project. His readiness and generosity has been very much appreciated.

I would also like to thank everyone who has participated in giving precious assist and valuable feedback to this project. The inspirations gained from these advice greatly promote the planning and development of the game.

Table of Contents

Acknowledgement	I
List of Figures	II
Abbreviations	III
1 Introduction	7
<i>1.1 Background</i>	<i>7</i>
<i>1.2 Objective and Scope</i>	<i>9</i>
<i>1.3 Report Outline</i>	<i>10</i>
2 Methodology	11
<i>2.1 VR Equipment and Set-up</i>	<i>11</i>
<i>2.2 Game Engine</i>	<i>12</i>
<i>2.3 Backend Scripting and Database Support</i>	<i>13</i>
3 Implementation Details	14
<i>3.1 Virtual Game Scene</i>	<i>14</i>
<i>3.2 Player Design</i>	<i>15</i>
<i>3.2.1 Unity Vive Input Binding</i>	<i>16</i>
<i>3.2.2 Movement Controller</i>	<i>17</i>
<i>3.2.3 Weapon Controller</i>	<i>18</i>
<i>3.2.4 Camera Positioning</i>	<i>18</i>
<i>3.3 Enemies</i>	<i>19</i>
<i>3.3.1 Man-in-the-browser</i>	<i>20</i>
<i>3.3.2 Malicious Code Injection</i>	<i>21</i>
<i>3.3.3 Ransomware</i>	<i>22</i>
<i>3.4 Weapons</i>	<i>23</i>
<i>3.4.1 Encryption Laser Blaster</i>	<i>23</i>

3.4.2 <i>Light Saber</i>	24
3.5 <i>Items</i>	25
3.5.1 <i>Firewall</i>	25
3.5.2 <i>DDOS Mine</i>	26
3.6 <i>Health System</i>	27
3.6.1 <i>Player Health</i>	27
3.6.2 <i>Enemy Health</i>	28
3.7 <i>Game Managers</i>	28
3.7.1 <i>Enemy Manager</i>	28
3.7.2 <i>Item Manager</i>	29
3.8 <i>UI</i>	30
3.8.1 <i>Start Menu</i>	30
3.8.2 <i>Player Statistics</i>	32
3.9 <i>Database</i>	33
4 Challenges Encountered	35
4.1 <i>Character Modelling</i>	35
4.2 <i>Room-scale VR Setup</i>	36
5 Future Work	37
5.1 <i>New Weapons</i>	37
5.2 <i>More Map Choices</i>	37
5.3 <i>Clear Explanation of Cybersecurity Concepts</i>	38
6 Conclusion	39
References	40

List of Figures

Figure 1: Watch Dogs	8
Figure 2: Rec Room	8
Figure 3: HTC Vive set	11
Figure 4: Unity	12
Figure 5: Virtual game scene	15
Figure 6: SteamVR input window	16
Figure 7: Developer binding UI	17
Figure 8: Screen capture of script	17
Figure 9: Screen capture of script	18
Figure 10: Screen capture of script	19
Figure 11: General cybercriminal model	20
Figure 12: Man-in-the-browser model	20
Figure 13: Screen capture of script	21
Figure 14: Malicious code injection model	22
Figure 15: Ransomware model	23
Figure 16: Encryption laser blaster in action	24
Figure 17: Light saber in action	25
Figure 18: Firewall model	27
Figure 19: DDOS mine model	27
Figure 20: Screen capture of script	28
Figure 21: Screen capture of script	30
Figure 22: Start menu	31
Figure 23: Leaderboard	32
Figure 24: Screen capture of script	32

Figure 25: Player statistics	33
Figure 26: Screen capture of script	34
Figure 27: Screen capture of script	34
Figure 28: Horse mask model	35
Figure 29: Base stations ideal positioning	36

Abbreviations

Three-dimensional: 3D

Virtual reality: VR

Operating system: OS

User interface: UI

Software development kit: SDK

Distributed denial-of-service: DDOS

1 Introduction

The following section introduces this report. Firstly, a brief background on cybersecurity will be given. The motivation of this project and related inspirations are also presented below, followed by an outline of the report content and structure.

1.1 Background

Over the recent decade, the general public have started relying on digital devices at an increasing scale due to the rapid technological advancements. With the increase in dependence on digital devices, escalation in the numbers of cybercrimes has also come along. Millions of people have been victims of cybercriminals without even noticing, as suggested by the fact that there were almost 700 million people suffering from cybercrimes in 2018 [1]. Currently, cybercriminals generate revenues of \$1.5 trillion annually, and this statistic is anticipated to continue growing, with the total damage of cybercrimes costing \$6 trillion annually by 2021 [2]. Among these many victims, teenagers is suggested to be the major age group that is fueling this growth, as they usually have the highest amount of exposure to non-evaluated applications, which are usually the sources of cybercrimes, making them the most vulnerable [3]. In view of the above situation, raising the awareness of the general public, especially the teenagers, on cybersecurity proves more important now than ever.

In order to efficiently explain and teach the teenager group about the concept of cybersecurity, game is chosen as the way to convey the message, particularly 3D VR game. VR is the concept of simulating human experience that can be similar to or completely different from the physical reality. 3D environment and VR both possess advantages for learning over other types of games, mostly because of their visual,

auditory and spatial elements which result in better recall for players on what they have learnt [4]. Moreover, the immersive setting created by them helps learners involve emotionally into the game due to realism, hence enhancing the learning experience further [5].

In the aspect of 3D game, the fundamental concept of this project is inspired by the 3D game “Watch Dogs” after reviewing its idea of introducing hacking technology into a game (see Figure 1). Its protagonist, who is a proficient hacker, will make use of cyber-abilities such as identity framing to achieve various tasks, and in the process it actually reflects how powerful or dangerous misuse of digital data could be, hence bringing up the importance of cybersecurity. On the other hand, in the aspect of VR game, the type of this project is inspired by the VR game “Rec Room” (see Figure 2). It features a VR world in which players could navigate around a social sandbox of places, and in different places players may engage themselves into various mini-games. After reviewing its mechanics, many would understand why VR game could provide a more immersive environment than other types of game. With the help of devices such as HTC Vive, players’ physical movements actually translate into the game, causing them to feel like they are really interacting inside the virtual world.



Figure 1: Watch Dogs



Figure 2: Rec Room

1.2 Objective and Scope

The objective of this project is to teach the general public, particularly the teenager group, about the nature, types and adverse effects of cybercrimes if they do not maintain better cybersecurity. Consequently, it should be able to help players better detect the occurrence of cybercrimes and protect themselves, or even protect others from suffering from these crimes, and eventually combat cybercriminals together.

This project will feature a 3D VR world in which player will be acting as a cyber-police. The sole aim of the player is to survive as long as possible under the ferocious attacks launched by all kinds of cyber-criminals and gain as many points as possible before he/she dies.

Different types of cyber-criminals will be modelled as different enemies. For instance, hackers that launch malicious code injection will be modelled as cybercriminals with blasters that can shoot out malicious code in the form of lasers. On the other hand, different security tools will be modelled as different weapons for the player. For example, anti-virus software will be modelled as light saber which could kill enemies with just couple of slashes.

At the start of game, player will have default weapons such as blasters to battle against enemies, and he/she could pick up items to enhance their survivability. As the game progresses and player kills more and more enemies, he/she will gain points and the difficulty level will rise. As the level rises, more and more enemies will be spawned, and player needs to rely on strategic route planning to prolong his/her survival. Once the player's health reaches 0, he/she dies and the game ends.

The target audience of this game is teenagers, ideally aging from 12-18, as they are the most vulnerable age group towards cybercrimes. It will be a single-player game, and as it incorporates both 3D and VR elements, HTC Vive set will be utilized as the VR equipment to realize the VR setting whereas Unity will be employed to manage the 3D aspects. Last but not least, a number of the most prevalent cyber threats, such as man-in-the-browser attack and ransomware, will be adopted as the models of enemies. In the long term, we hope to include more and more types of cybercrimes so as to keep players updated with the current development of cybersecurity issues and better equip them against newly emerging ones in the future.

1.3 Report Outline

The remainder of this paper is organized as follows. Firstly, Chapter 2 details the methodologies used in this project, including a description on the VR equipment used and game engine adopted. Following in Chapter 3 is an explanation of the implementation details of this project, which includes the design of player, modelling of enemies and development of game managers. Then, the difficulties encountered during the development will be listed in Chapter 4, such as the modelling of different characters and setting up of VR environment. Next, the future work plan is given in Chapter 5. Lastly, a short conclusion is provided in Chapter 6.

2 Methodology

The following section explains the methodologies employed for realizing the VR setting, game environment and backend support, which includes an introduction to HTC Vive set and description on both Unity Game Engine and PHP scripting language.

2.1 VR Equipment and Set-up

This project will make use of HTC Vive set as its main tool (see Figure 3) to realize the VR setting. HTC Vive set is one of the best VR equipment offered in the market currently. It mainly comes with a HTC Vive Headset, two HTC Vive Controllers and two HTC Vive Base Stations.

Through positioning the base stations in opposite ends of the room in which you want to set up a VR setting, they scan and process the physical 3D information of the whole room and precisely track the location of the headset and controllers within the area. On the other hand, the headset simulates a 360-degree VR gaming environment and provides directional audio to ensure that with the help of the clear-cut location tracking, players could fully experience every bit of the VR environment. Last but not least, the two controllers also give players the ability to virtually interact with anything in the VR environment, such that they could thoroughly immerse into the VR experience.



Figure 3: HTC Vive set

2.2 *Game Engine*

This project will be using Unity as its game engine. Unity is a cross-platform game engine that is supported by more than 25 different platforms, which virtually means that most of the OS could run Unity games. Moreover, Unity supports both 3D and VR games, therefore it is perfectly suitable with the theme of this project.

In addition, Unity has a large and supportive community with its long development history, therefore many difficulties encountered during implementation have solutions available in the community forum. Furthermore, its official documentation is well written and easy to understand, which simplifies the learning process a lot. Also, Unity has its own extensive asset store where users could share or sell the models, shaders or any other resources they have created. By utilizing these high-quality assets, cost on designing models from scratch could be minimized, such that more time could be invested in game implementation and testing.

However, there is also limitation to Unity besides the abovementioned advantages. As Unity VR games that make use of HTC Vive set only support Windows OS [6], Windows Platform is the only option we could choose as the development platform for this project.



Figure 4: Unity

2.3 Backend Scripting and Database Support

A leaderboard function will be implemented to compare and show the five highest score achieved by the player base over the course of gaming to increase the incentive for players to play the game. As teenagers like to compete with one another, the competitiveness introduced by this function will be used as a driving force for the players to continue equipping themselves with cybercrime knowledge.

In order to support this feature, backend scripting and database are required. For the backend scripting, PHP will be used to add records to the database whenever a player has finished a round and retrieve the top five records from the database whenever the leaderboard function is initiated. For the database support, MySQL will be used to record their usernames and scores.

3 Implementation Details

The following section details the implementation of the whole project. First of all, structure of the game scene will be explained, followed by the design of player and enemies. Then, various weapons and items will be introduced, together with a description on the health system. Last but not least, implementation of the game managers, UI and database will be talked about to cover most aspects of the game.

3.1 *Virtual Game Scene*

The game scene for this project has been specifically designed to embrace a cyber theme in every aspect (see Figure 5). First of all, the skybox, which refers to the entire scene that shows what the world looks like beyond the main venue geometry, has been rendered with a pitch-black material with fluorescent blue slashes to simulate an environment that is within a computer system. The overall color scheme of the game venue is inclined to a darker theme, so as to create a more mysterious atmosphere.

For the main structural components of the scene, there are a lot of tunnels and shelters coated with a transparent light blue material to again enhance the cyber atmosphere of the game. These tunnels also serve to increase the sense of multi-level and complexity of the map. Moreover, as these tunnels are transparent, players could observe where the enemies are approaching at any time, which helps them plan their route and raise the pace of the game.

Furthermore, the arena is surrounded by red laser bars to restrict the playground and prevent players from falling off the map. The choice of using lasers as the boundary is also aimed at reinforcing the cyber atmosphere of the game.

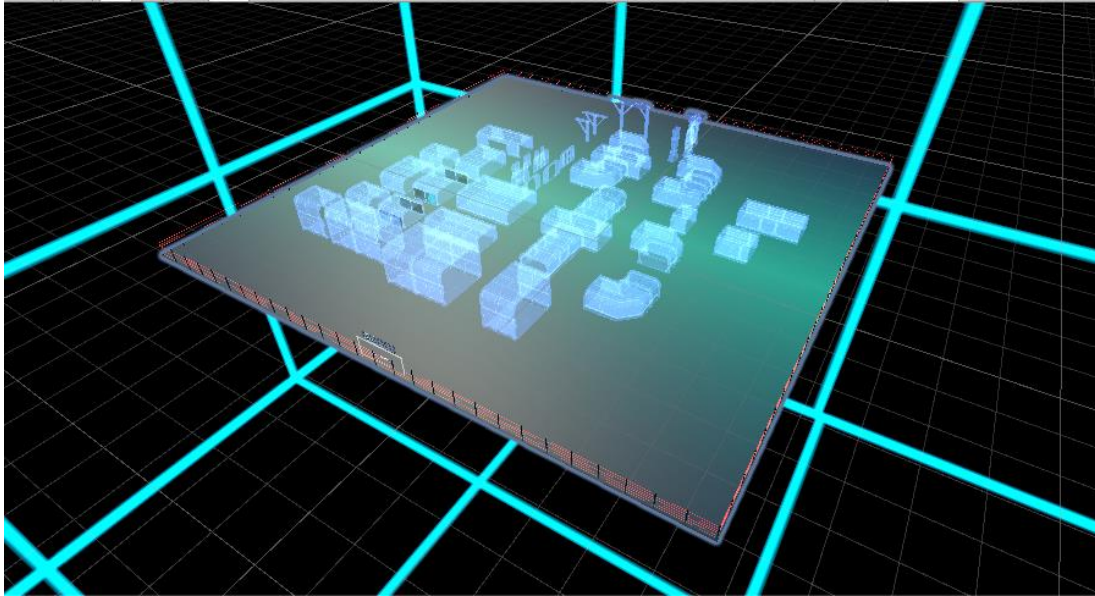


Figure 5: Virtual game scene

3.2 *Player Design*

As mentioned in the section of objective and scope, player will be acting as a cyber-police and will need to survive as long as possible under the attacks of cybercriminals in order to gain increasing amount of points in this VR game. Therefore, it is essential to allow the in-game character to possess smooth movement and weapon controls under VR setting, and these are also the major components for our player implementation.

Throughout the development process of the in-game character, two main stages were experienced. The first stage involved the development of a 3D character with decent controls, full set of weapons and related animations handled by respective animators. In this stage, VR capabilities have yet to be implemented and the core aim is to complete all 3D aspects of the player such that it could serve as a foundation for the more complex VR implementation. The second stage then involved the migration towards VR character, changing the controls from keyboard input to HTC Vive Controller input and switching from normal main camera to VR camera. The resulting VR character could

interact with the VR environment smoothly with precise location tracking and clear scene rendering, all thanks to the seamless linkage between Unity Engine and HTC Vive Set.

3.2.1 Unity Vive Input Binding

In order to smoothly reflect HTC Vive Controller input in Unity, OpenVR SDK has been utilized. It supports SteamVR platform such that once the SteamVR Unity plugin is installed into the project, we could bind self-defined actions in the SteamVR Input Window (see Figure 6) with the actual input of the controllers through the binding UI provided (see Figure 7). After this binding, whenever certain input of the controllers is performed, the respective action bound will be triggered. By listening to these actions in the scripts of Unity (see Figure 8), we could carry out corresponding character operations inside the game.

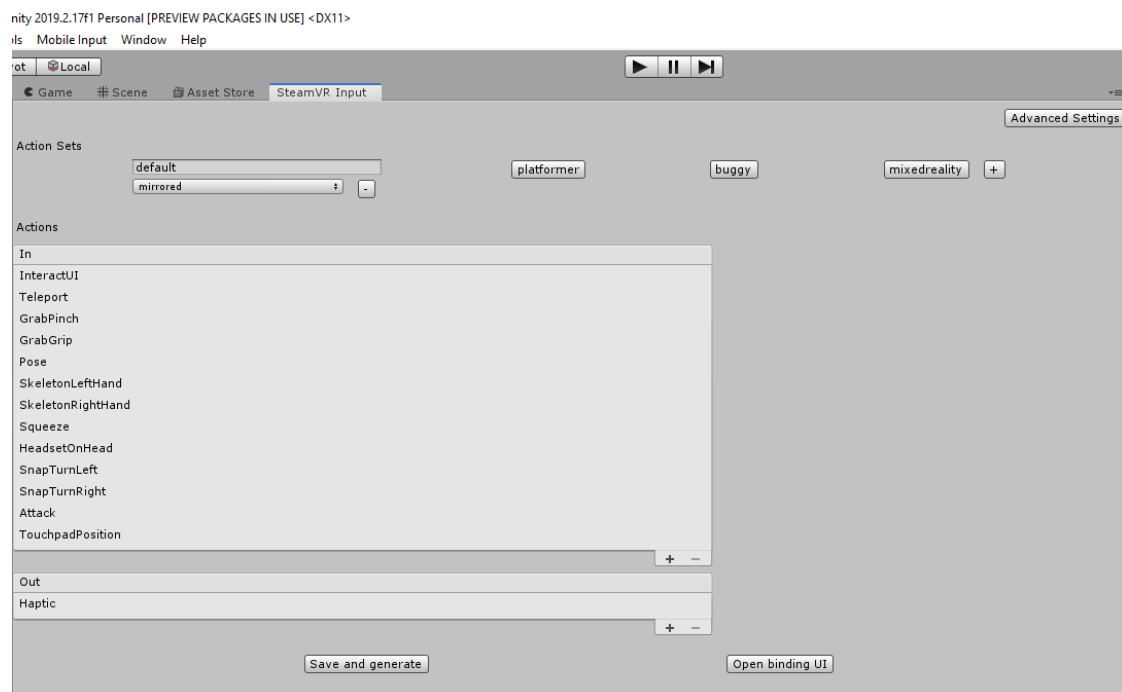


Figure 6: SteamVR input window

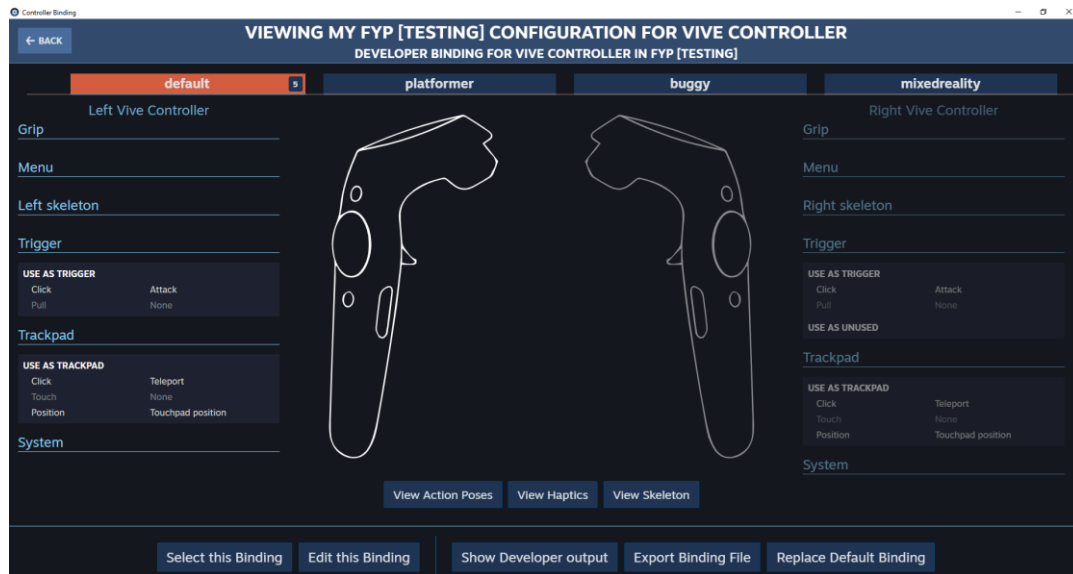


Figure 7: Developer binding UI

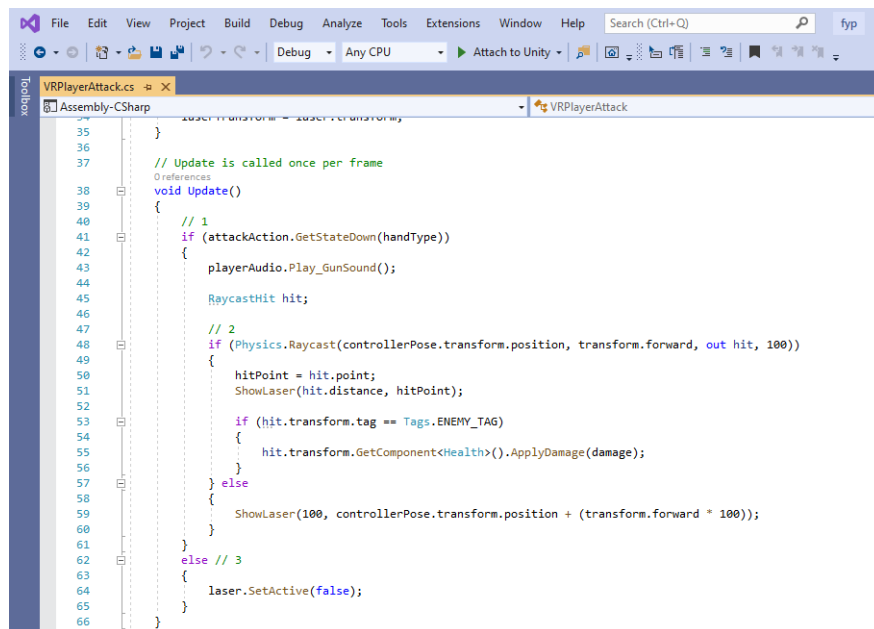
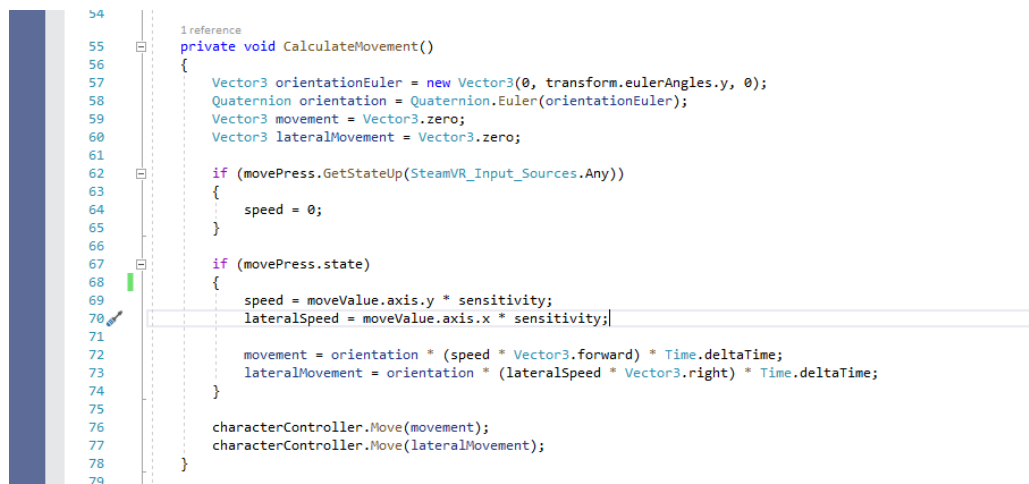


Figure 8: Screen capture of script

3.2.2 Movement Controller

Successful implementation of the input binding makes movement control a lot easier. Through binding the trackpad click with the movement initialization, whenever the player presses on the trackpad on either of the HTC Vive Controllers, the movement controller script will base on the position of the press on the trackpad to determine the

direction and speed of the character movement (see Figure 9).



```
>4
55 1 reference
56 private void CalculateMovement()
57 {
58     Vector3 orientationEuler = new Vector3(0, transform.eulerAngles.y, 0);
59     Quaternion orientation = Quaternion.Euler(orientationEuler);
60     Vector3 movement = Vector3.zero;
61     Vector3 lateralMovement = Vector3.zero;
62
63     if (movePress.GetStateUp(SteamVR_Input_Sources.Any))
64     {
65         speed = 0;
66     }
67
68     if (movePress.state)
69     {
70         speed = moveValue.axis.y * sensitivity;
71         lateralSpeed = moveValue.axis.x * sensitivity;
72
73         movement = orientation * (speed * Vector3.forward) * Time.deltaTime;
74         lateralMovement = orientation * (lateralSpeed * Vector3.right) * Time.deltaTime;
75     }
76
77     characterController.Move(movement);
78     characterController.Move(lateralMovement);
79 }
```

Figure 9: Screen capture of script

3.2.3 Weapon Controller

Using similar concept as the movement controller, weapon controller script listens to the weapon change action, which is bound with the trigger input of the left HTC Vive Controller, such that whenever the trigger is used, player could switch to another weapon (details of the weapons will be discussed in later section).

3.2.4 Camera Positioning

VR scene rendering requires a VR camera to accomplish, therefore we need to replace the normal main camera setting of the 3D environment with a VR camera. Fortunately, SteamVR plugin provides a lot of prefabs designed for VR game development, and “CameraRig” and “SteamVR” have the capabilities to help us achieve our aim. “CameraRig” has a VR camera embedded in it while “SteamVR” is responsible for the rendering function of all VR cameras in the scene. By mounting these two prefabs onto the player as its children and managing the VR camera position with specific script such that it always follow the headset position (see Figure 10), the game will constantly maintain a first person perspective for the player view.

```

44 1 reference
45 private void HandleHead()
46 {
47     Vector3 oldPosition = cameraRig.position;
48     Quaternion oldRotation = cameraRig.rotation;
49     transform.eulerAngles = new Vector3(0.0f, head.rotation.eulerAngles.y, 0.0f);
50
51     cameraRig.position = oldPosition;
52     cameraRig.rotation = oldRotation;
53 }
54

```

Figure 10: Screen capture of script

3.3 Enemies

In this project, enemies are modelled as cybercriminals whose aim is to strike down the player and take control of the whole system. Different cybercriminals will be modelled as different enemies, as mentioned in the section of objective and scope. In order to achieve this, all enemies have an enemy controller script attached to them, which handles their movement, attacks, animations and audio.

In particular, their movements are specially designed such that they could automatically locate the player and chase him or her across the map with accurate path-finding mechanism. To make this feature possible, we first mounted a “NavMeshAgent” component to the enemy. Then, we carefully selected the scene geometry that could affect the navigation and marked them as navigation static. Next, based on whether the enemies should be able to walk on the structural components or not, we classify the surfaces as “walkable” and “not walkable”. After this, all setting has been completed, and we could carry out the NavMesh baking process. This generates a NavMesh for the “NavMeshAgent” of the enemy to automatically find the optimal path towards its target, which is set to be the player, and this is how we succeeded in generating an automated path-finding mechanism for the enemies.

In the following parts, details specific to each type of enemy will be introduced, including their design and meaning.

3.3.1 Man-in-the-browser

Man-in-the-browser attack refers to a form of security attack where cybercriminal installs a Trojan horse malware on a victim's web browser to modify his/her web transactions [7]. In the game, it is modelled as cybercriminal with horse mask. The general cybercriminal model (see Figure 11) symbolizes its identity as a hacker, whereas the horse mask appended on its head represents its trait of utilizing Trojan horse malware (see Figure 12). To conform to the theme of this enemy type, they have a unique audio which resembles the neighing of horse. For their attacks, they make use of their long heads to initiate head-butts, which is governed by the melee attack script. This script casts a `Physics.OverlapSphere()` around its head (see Figure 13) such that whenever this sphere collides with the player, he/she will be damaged (details of the health system will be discussed in later section).



Figure 11: General cybercriminal model

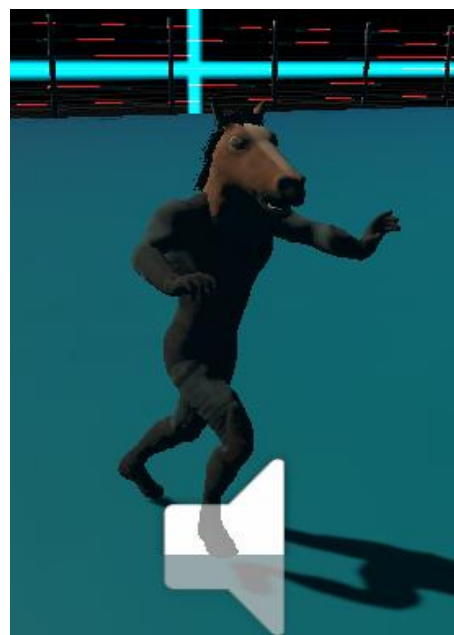


Figure 12: Man-in-the-browser model

```

5 public class MeleeAttack : MonoBehaviour {
6
7     public float damage = 20f;
8     public float radius = 1.5f;
9     public LayerMask layerMask;
10
11     void Update () {
12
13         Collider[] hits = Physics.OverlapSphere(transform.position, radius, layerMask);
14
15         if(hits.Length > 0) {
16
17             hits[0].gameObject.GetComponent<Health>().ApplyDamage(damage);
18             gameObject.SetActive(false);
19         }
20     }
21 }
22
23 // class
24
25
26

```

Figure 13: Screen capture of script

3.3.2 Malicious Code Injection

Malicious code injection attack refers to the exploitation of input validation flaws in software to introduce and execute malicious code [8]. In the game, it is modelled as cybercriminal with laser blaster that can shoot out malicious code. The general cybercriminal model is again used but no more horse mask is adopted. Instead, they are equipped with a dark-themed blue blasters that allows long range attacks rather than melee attacks (see Figure 14). Therefore, contrary to man-in-the-browser, which initiate attacks only when they reach the player, malicious code injection starts attacking from much further distance. For their attack form, it is undoubtedly laser shooting, and these laser bullets will damage the player if he/she fails to dodge them.

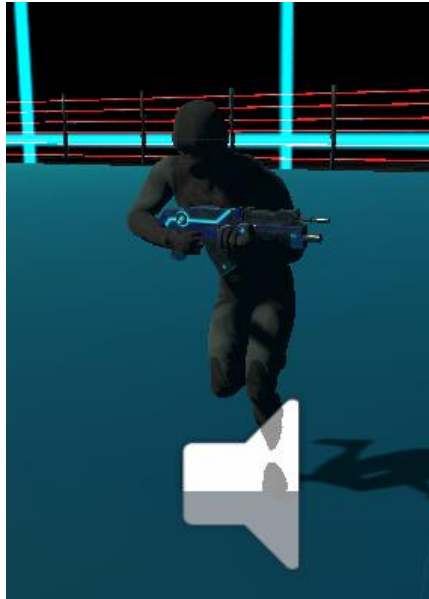


Figure 14: Malicious code injection model

3.3.3 Ransomware

Ransomware refers to a type of malware that prevents victims from accessing their personal data stored unless a ransom is paid [9]. More advanced one makes use of cryptoviral extortion to encrypt the victim's file and cause them inaccessible. Without the corresponding decryption key, it is extremely difficult to recover the files. In this project, we are exactly modelling this type of advanced ransomware as we want an enemy type to serve as semi-bosses in the game so as to enhance the excitement and surprising element, and its seriousness perfectly suits the role. In the game, it is modelled as a mutated monster (see Figure 15). Its bulkiness and monstrous appearance reinforce the feeling that it is a semi-boss. Moreover, its attack that swings its sickle forward will kill the player in one hit, which symbolizes the serious consequences that may occur if you got attacked by a ransomware in real life situation.

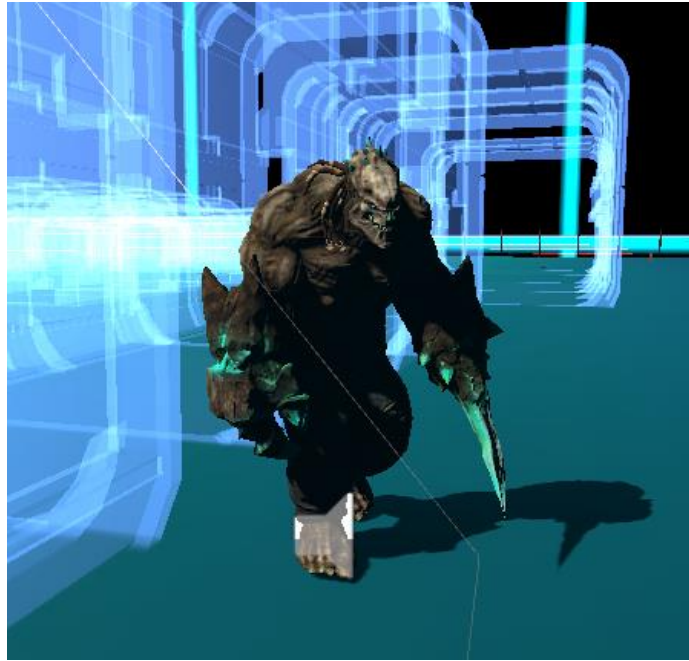


Figure 15: Ransomware model

3.4 *Weapons*

As mentioned in the section of objective and scope again, different security tools will be modelled as different weapons for the player. Although player has a weapon controller script to handle the changing of weapons throughout the course of gaming, the interactions of every weapon is governed by its own specific script. Furthermore, to render the weapons in a way such that players could feel like they are grabbing the virtual weapons naturally with their hands, we precisely mounted the weapon models onto the in-game HTC Vive Controller models and accurately followed their orientation and scale. As a result, the weapon models practically replaced the in-game controller models and every locational or trigger input from the controllers were flawlessly reflected by the weapon models.

3.4.1 Encryption Laser Blaster

Besides malicious code injection uses laser blasters, players also have laser blasters as

their weapons. However, the former uses laser blasters to inject malicious code bullet into the player's body, whereas the latter shoots out laser beams which represent encrypted data stream to attack enemies (see Figure 16). As the laser beams hit the enemies, damage will be dealt, and this symbolizes the situation where cybercriminals fail to decrypt encrypted messages we send with encryption such as DES.

This weapon is governed by the blaster script, which listens to the right HTC Vive Controller trigger and shoots out laser beams whenever the trigger is pulled. The direction of the laser beams is programmed to always be the forward direction of the blaster, such that with good aim and steady posture, damaging the enemies with the encrypted data stream should be an easy task.



Figure 16: Encryption laser blaster in action

3.4.2 Light Saber

Other than encryption laser blaster, players could also use anti-virus software to defeat enemies. In the game, anti-virus software is modelled as light saber that could kill most enemies with 2 strikes (except ransomware as it has significantly higher amount of

health). In comparison to blasters, this weapon specializes in melee attack, and should be used when multiple enemies are in close range.

The light saber script attached to it monitors the movement of the saber, such that whenever it is swung and slashes the enemies, damage will be dealt. With proper route planning to group enemies together, this weapons could kill tens of enemies in mere seconds.



Figure 17: Light saber in action

3.5 *Items*

Besides enemies and weapons, there are items spawning in random positions throughout the map. These items could only be picked up by the players. They add to the complexity of both the enemy and weapon roster as interacting with different items may bestow beneficial or negative effect to the players.

3.5.1 Firewall

Firewall refers to a kind of network security system that combines both hardware and

software to isolate an internal network from the Internet based on predetermined security rules. It helps to protect internal computer system by blocking off data packets from suspicious site in external network, and is one of the most common security tools. In the game, it is modelled as a red-brick wall (see Figure 18) which upon picking up, it could boost the player's defense to the highest extent that could make him/her immune to damage from any type of enemies. This effect of blocking off attacks symbolizes the power of actual firewall that helps to prevent suspicious content from any external network from compromising your system. It persists for around 10 seconds and after that the player's defense will be reset to normal level.

3.5.2 DDOS Mine

DDOS refers to a type of cyber-attack in which cybercriminal causes an online service unavailable to its users through overwhelming it with huge amount of network requests originating from multiple sources [10]. In the game, it is modelled as a static mine (see Figure 19) which upon stepping on, it causes the player to move extremely slow in order to symbolize the overloading situation of system attacked by DDOS. Same as the firewall, this effect stays for about 10 seconds before wearing off. In this situation, players usually become very vulnerable to attacks as he/she could not possibly dodge the attacks with this altered speed.



Figure 18: Firewall model



Figure 19: DDOS mine model

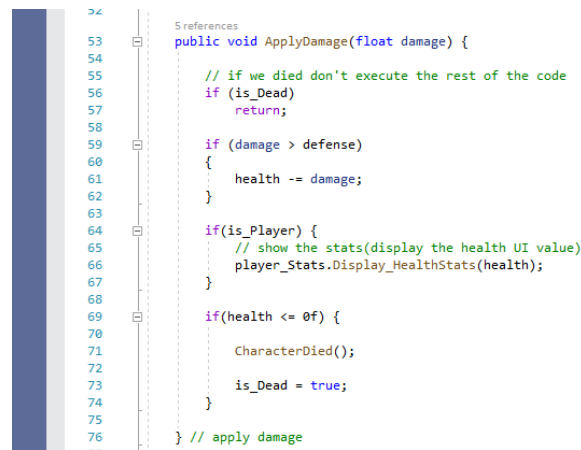
3.6 *Health System*

Health is the most important resource for players in the game, as once the player's health reaches 0, he/she will die and the round will end. Therefore, monitoring the remaining health is a very crucial task during the course of survival. In this project, we make use of a health script that is attached to both player and enemy to build the general health system. This health script facilitate interactions between them such that they could properly deal damage to one another and reduce their health.

3.6.1 Player Health

The health script attached to player is directly linked to the health bar displayed at the top left corner of the view (details of the UI will be discussed in later section). As a result, whenever damage is taken by the player, the health bar volume will be reduced. On the other hand, whenever player deals damage to the enemies, `ApplyDamage()` function will be called (see Figure 20) for the enemy's health script to lower its health.

By default, each player has 100 health points at the start of round. When his/her health reaches 0, he/she dies and all movements or attacks will be prohibited. After around 5 seconds, player may proceed with another round to try surpassing his/her past records.

A screenshot of a script editor window. On the left, a vertical scrollbar and line numbers from 53 to 76 are visible. The script code is as follows:

```
53 public void ApplyDamage(float damage) {  
54  
55     // if we died don't execute the rest of the code  
56     if (is_Dead)  
57         return;  
58  
59     if (damage > defense)  
60     {  
61         health -= damage;  
62     }  
63  
64     if(is_Player) {  
65         // show the stats(display the health UI value)  
66         player_Stats.Display_HealthStats(health);  
67     }  
68  
69     if(health <= 0f) {  
70  
71         CharacterDied();  
72  
73         is_Dead = true;  
74     }  
75  
76 } // apply damage
```

Figure 20: Screen capture of script

3.6.2 Enemy Health

For the enemy, ApplyDamage() function will be similarly called for the player's health script if they succeed in dealing damage to the player. Different types of enemy has different amount of health points. For instance, man-in-the-middle has 100 health points while ransomware has 500. When the enemy's health reaches 0, their death animation will be played, and their corpse will disappear in about 3 seconds.

3.7 Game Managers

Game managers are used to manage the spawning of various things during a round. There are 2 managers in this project, namely enemy manager and item manager, which handle the spawning of enemies and items respectively.

3.7.1 Enemy Manager

Enemy manager spawn enemies in random positions across the map. We have put many

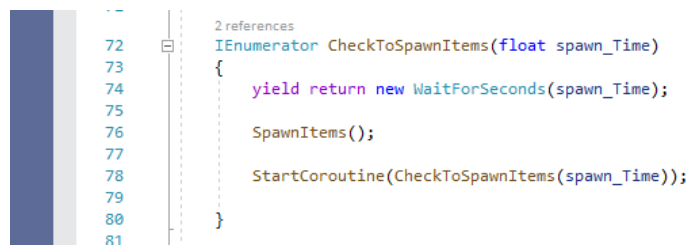
spawn points on the map such that when the enemy manager needs to spawn enemies, it could randomly select from this huge pool of locations to generate a randomized output. At the start of game, there will be 5 man-in-the-browser and 5 malicious code injection; however, as one wave of enemies is killed, the difficulty level will rise, and more enemies will be spawned in subsequent waves. To accomplish this behavior, enemy manager is programmed to spawn 2 more man-in-the-browser and 1 more malicious code injection in every wave; moreover, based on current score of the player, the enemy manager will start spawning ransomware if he/she reaches a checkpoint, which significantly increases the difficulty of the game.

Besides the spawning of enemies, enemy manager also maintains the score of player. Killing a man-in-the-browser will reward 10 points to the player, whereas killing a malicious code injection and ransomware will reward 15 and 30 points respectively. When the player dies, it will send the score to our static database class, which further send it to the database and store it up (details of the database will be discussed in later section).

3.7.2 Item Manager

Item manager is responsible for the spawning of items. Similar to enemy manager, there is a huge pool of spawn points such that it could randomly select one to spawn the required item. However, different from enemy manager, it does not spawn when all items are consumed. Instead, it checks the game situation every 5 seconds and spawns the consumed items in random positions with the help of StartCoroutine() (see Figure 21). At the start of game, there will be 4 firewall and 10 DDOS mine in the map. Whenever player consumes an item, item manager will count the number of depleted

item such that when the periodic checking arrives, it will know how many items to spawn back into the map.



```
72 2 references  
73 IEnumerator CheckToSpawnItems(float spawn_Time)  
74 {  
75     yield return new WaitForSeconds(spawn_Time);  
76     SpawnItems();  
77     StartCoroutine(CheckToSpawnItems(spawn_Time));  
78 }  
79  
80  
81
```

Figure 21: Screen capture of script

3.8 UI

In order to accurately and clearly display the present situation throughout the gaming process and substantially ease the interactions between player and game, UI is a paramount tool that could not be neglected. In the game, start menu and player statistics are our main UI that have applied totally different set of features to realize their purposes.

3.8.1 Start Menu

Start menu is composed of a virtual keyboard, start button and leaderboard button (see Figure 22). The virtual keyboard is constructed such that player could type their desired player name for each round by clicking on the keys with their controllers, as they could not use their actual keyboards in a VR setting. By clicking the start button, the game will start after a 5-second countdown; however, if the player has not typed his/her player name, the start button will not function. Last but not least, by clicking the leaderboard button, a leaderboard will be displayed, and the top 5 scores achieved by the current player base will be shown (see Figure 23). In order to see your name in the leaderboard, you must attain the top 5 ranking in the whole player base; as a result, seeing your name in the leaderboard is a solid proof of your skills.

For the sake of allowing proper interactions with the start menu, we first set the render mode of its canvas to “World Space” rather than “Screen Space”. The difference between “World Space” and “Screen Space” is that the former gives the canvas a fixed physical location where the canvas is actually positioned, such that whenever players want to use the start menu functions, they could simply move to that particular location; whereas the latter gives no fixed location to the canvas, such that even though the canvas will always follow the orientation of the camera and be rendered, players actually could not interact with it as it may be positioned really far from them. Besides this setting, a pointer attached to the right controller has also been built for players to actually click the buttons through sending pointer event to them. As the buttons receive pointer event, their `IPointerClickHandler` will be invoked, and respective functions will be handled (see Figure 24).

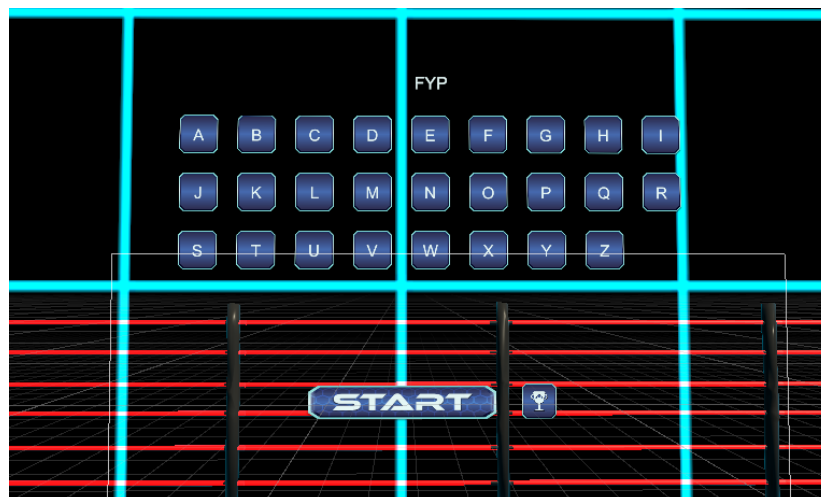


Figure 22: Start menu



Figure 23: Leaderboard

```

7 public class PlayButtonClick : MonoBehaviour, IPointerClickHandler
8 {
9
10     private GameObject pointer;
11     private GameObject player;
12
13     public GameObject playerStatsCanvas;
14     public GameObject leaderboardButton;
15     public GameObject keyBoard;
16     public GameObject username;
17
18     private void Awake()
19     {
20         pointer = GameObject.FindWithTag(Tags.POINTER_TAG);
21         player = GameObject.FindWithTag(Tags.PLAYER_TAG);
22     }
23
24     public void OnPointerClick(PointerEventData eventData)
25     {
26         if (username.GetComponent<Text>().text != "")
27         {
28             player.GetComponent<Health>().setUsername(username.GetComponent<Text>().text);
29
30             pointer.SetActive(false);
31             playerStatsCanvas.SetActive(true);
32             leaderboardButton.SetActive(false);
33             keyBoard.SetActive(false);
34             username.SetActive(false);
35             gameObject.SetActive(false);
36         }
37     }
38 }
39

```

Figure 24: Screen capture of script

3.8.2 Player Statistics

Player statistics include health bar, current score and item indicators (see Figure 25). Health bar is positioned at the top left corner of the view, which monitors your current health in real-time. Current score is positioned at the top right corner, and its purpose is to display your gained points throughout the round so far. Last but not least, the item indicators are located right below the health bar. There are 2 indicators currently, which correspond to firewall and DDOS respectively. When player picks up firewall, the

firewall indicator which looks like a shield will pop up to represent that he/she has very high defense now; whereas when player steps on a DDOS mine, the DDOS indicator will pop up to symbolize that his/her speed is severely lowered.

Contrary to start menu, the canvas for player statistics is rendered in “Screen Space - Camera” mode. This allows the canvas to always follow the player’s camera and be displayed within player’s view, and as players do not need to interact with this canvas, interaction problems will not occur.

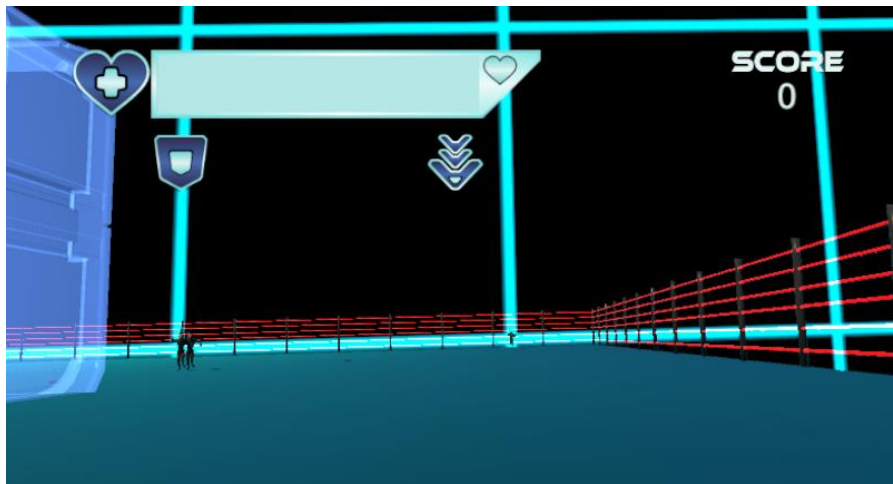


Figure 25: Player statistics

3.9 Database

To support the leaderboard function, a static database class has been implemented to communicate with our backend and MySQL database (see Figure 26). After enemy manager calls the `addRecord()` function of the static class and sends the player name and score as the arguments, an `UnityWebRequest` will be sent to our backend PHP script which handles the logic of inserting records into MySQL table.

On the other hand, before every new round starts, player will call the `retrieveRecord()` function of the static class to get the top 5 scores from MySQL. Same as adding records

to MySQL, an UnityWebRequest will be sent to another backend script that we have deployed, which will help us manage all the logic for retrieving records (see Figure 27). The records will then be encoded as JSON string and passed back to our static class. Our class further parses the string into Unity JSON object such that we could obtain a simpler data structure for subsequent processing.

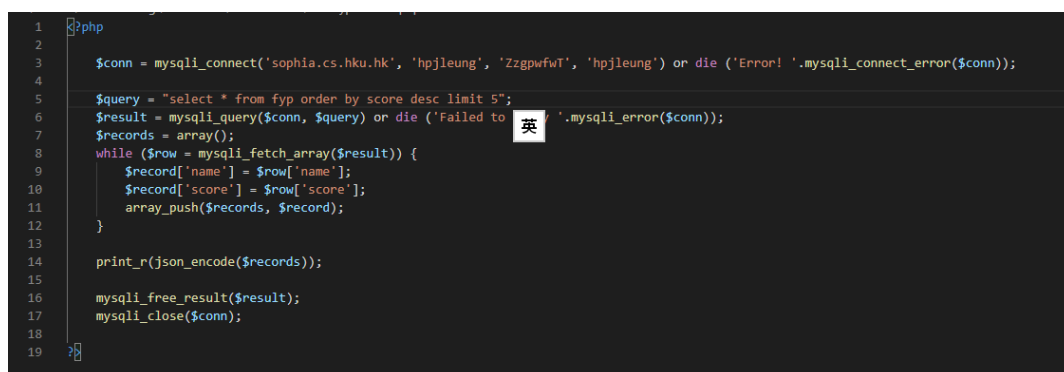


```

1  using SimpleJSON;
2  using System.Collections;
3  using System.Collections.Generic;
4  using UnityEngine;
5  using UnityEngine.Networking;
6
7  public static class Database {
8
9      public static IEnumerator retrieveRecord()
10     {
11         WWWForm form = new WWWForm();
12         using (UnityWebRequest request = UnityWebRequest.Post("https://i.cs.hku.hk/~hpjleung/fypRetrieve.php", form))
13         {
14             yield return request.SendWebRequest();
15             yield return new WaitUntil(() => request.isDone);
16
17             if (request.isNetworkError || request.isHttpError)
18             {
19                 Debug.Log(request.error);
20             }
21             else
22             {
23                 JSONNode result = JSON.Parse(request.downloadHandler.text);
24                 GameObject.FindWithTag(Tags.PLAYER_TAG).GetComponent<Health>().setRecords(result);
25             }
26         }
27     }
28
29     public static IEnumerator addRecord(string username, int score)
30     {
31         WWWForm form = new WWWForm();
32         form.AddField("name", username);
33         form.AddField("score", score);
34
35         using (UnityWebRequest request = UnityWebRequest.Post("https://i.cs.hku.hk/~hpjleung/fypAdd.php", form))
36         {
37             yield return request.SendWebRequest();
38             yield return new WaitUntil(() => request.isDone);
39
40             if (request.isNetworkError || request.isHttpError)
41             {
42                 Debug.Log(request.error);
43             }
44             else
45             {
46                 Debug.Log("finished");
47             }
48         }
49     }
50 }
51
52

```

Figure 26: Screen capture of script



```

1  <?php
2
3  $conn = mysqli_connect('sophia.cs.hku.hk', 'hpjleung', 'ZzgpwfwT', 'hpjleung') or die ('Error! ' .mysqli_connect_error($conn));
4
5  $query = "select * from fyp order by score desc limit 5";
6  $result = mysqli_query($conn, $query) or die ('Failed to ' .mysqli_error($conn));
7  $records = array();
8  while ($row = mysqli_fetch_array($result)) {
9      $record['name'] = $row['name'];
10     $record['score'] = $row['score'];
11     array_push($records, $record);
12 }
13
14 print_r(json_encode($records));
15
16 mysqli_free_result($result);
17 mysqli_close($conn);
18
19

```

Figure 27: Screen capture of script

4 Challenges Encountered

The following section talks about challenges that have been faced during the development of this project. Possible mitigation strategies are also proposed to solve them.

4.1 Character Modelling

With realistic and artistic in-game models, a game will be much more presentable to its player base. However, as a developer, I do not possess adequate artistic sense and skills to create such models. I had limited experience on modelling 3D characters for Unity, therefore creating custom-made characters using software such as Blender significantly increases the workload.

After several trials with unsatisfactory modelling results, I adopted another strategy to minimize workload on modelling and prevent wastefully use up my time resources, which is to make use of existing assets shared on Unity Asset Store or combine them to construct my own suitable models. Besides the asset store, Mixamo website also helped me a lot in finding fitting resources. For instance, the model for man-in-the-browser is constructed by combining horse mask model from Unity Asset Store (see Figure 28) with cybercriminal model from Mixamo. This method aided me in building most of the models inside the game.



Figure 28: Horse mask model

4.2 Room-scale VR Setup

Due to the current difficult situation that prohibited citizens from leaving home too often, I could not test the VR game with well-established VR setup in university's multimedia laboratory. As a result, I needed to borrow the VR equipment back home and setup a room-scale VR environment myself. However, for the room-scale setup to function properly, several requirements need to be fulfilled. First of all, you need to have enough sockets in your room to satisfy the extra VR equipment plugs besides your original devices setup. To accomplish this condition, I had to bring in another extension unit besides the original one into the room, which caused the situation to be a bit messy. Secondly, the base stations needed to be mounted in opposite ends of the room at a height higher than your body height (see Figure 29) to accurately track your position. This proves to be quite difficult as there were no locations in my room that were both higher than my head height and able to be reached by the wires. With no perfect solution to that, I could only place the base stations in opposite ends with one of them lower than my head height, which caused periodic tracking errors during the testing process.

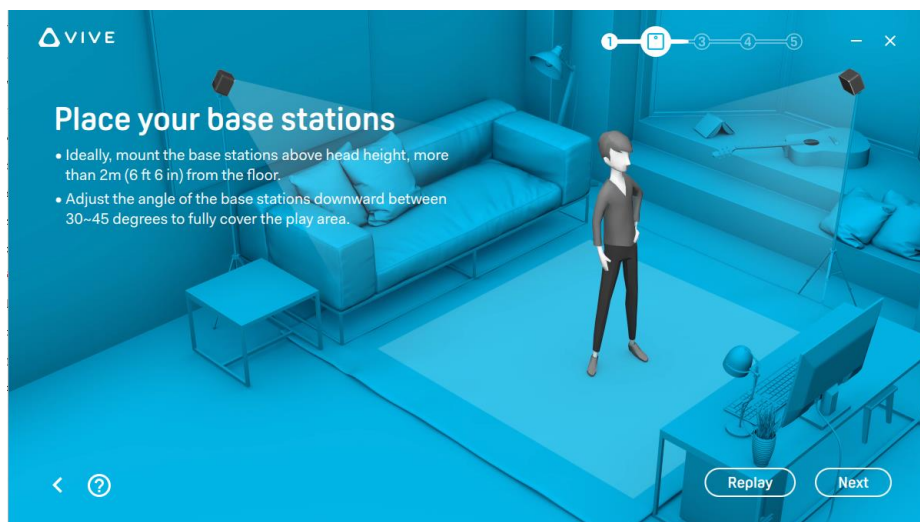


Figure 29: Base stations ideal positioning

5 Future Work

The following section talks about future development that could be continued after the completion of current version, which includes adding more weapons to the player, increasing map choices for the game and giving clear explanation of the cybersecurity concepts through the UI.

5.1 *New Weapons*

For the sake of enhancing diversity of the game and giving more control to the player's playstyle, more weapons could be added into the project. For example, a backup pack could be included into the weapon roster as a tool to replenish health. It works by storing a copy of current remaining health such that whenever player is at critical health, he/she can use the backup pack and replenish the health he/she stored. That means it works the best when player is at full health as the copy of current remaining health will be the amount of full health. This new weapon is aimed at symbolizing the action of periodic backing up your files such that even if your computer system is hacked, your files will not be lost.

5.2 *More Map Choices*

Besides the current cyber map, more maps could be constructed and introduced to increase player's choices. This again enhances the diversity of the game and reinforces the excitement of the project. For instance, a new map focusing on vertical development could be added to build a contrast with the current map, which emphasizes on horizontal movement, tunnels, shelters and cyber atmosphere.

5.3 Clear Explanation of Cybersecurity Concepts

This game has embedded cybersecurity concepts in every design so as to fully pursue the final aim of raising teenagers' awareness on cybersecurity. However, it may be hard for players to understand every meaning behind during gaming without explicit explanation. As a result, clear explanation of every cybersecurity knowledge included in the game through the UI will be added in future stages to ensure that players could fully grasp the meaning and be aware of the importance of cybersecurity.

6 Conclusion

Cybersecurity is an issue that has been proven more important now than ever due to the ever-growing trend of cybercrimes in recent decade. This paper proposes a 3D VR game for learning essential knowledge about cybercrimes in an exciting manner. Different methodologies to realize the game setting has been introduced, followed by a detailed explanation on every aspect of the project implementation. Furthermore, challenges encountered throughout the development process and related mitigation strategies have also been looked into.

In the future, we hope to include more weapons and maps to enhance the diversity of the game. Clear explanation on the cybersecurity concepts behind every design through the UI is also desirable such that players could more easily understand the cybersecurity knowledge we want to convey during gaming. To conclude, we hope that this game may provide a fascinating gaming and more importantly, learning experience to our players.

References

- [1] Bera, Ana. "83 Terrifying Cybercrime Statistics" safeatlast. March 12, 2019. <https://safeatlast.co/blog/cybercrime-statistics/>
- [2] Morgan, Steve. "The 2019 Official Annual Cybercrime Report" HERJAVEC GROUP. June 21, 2019. <https://www.herjavecgroup.com/the-2019-official-annual-cybercrime-report/>
- [3] Norton by Symantec. "Youngsters are more vulnerable to cyber crime: Norton study" BGR. August 4, 2016. <https://www.bgr.in/news/youngsters-are-more-vulnerable-to-cyber-crime-norton-study/>
- [4] Kapp, Karl. "Advantages of 3D for Learning" Kapp Notes. February 9, 2019. <http://karlkapp.com/advantages-of-3d-for-learning/>
- [5] Rogers, Sol. "Virtual Reality: THE Learning Aid Of The 21st Century" Forbes. March 15, 2019. <https://www.forbes.com/sites/solrogers/2019/03/15/virtual-reality-the-learning-aid-of-the-21st-century/#451d501139b6>
- [6] HTC Corporation. "What are the system requirements?" VIVE. September 28, 2018. https://www.vive.com/hk/support/vive/category_howto/what-are-the-system-requirements.html
- [7] Rouse, Margaret. "man in the browser" TechTarget. August, 2006. <https://searchsecurity.techtarget.com/definition/man-in-the-browser>

[8] Banach, Zbigniew. “What is Code Injection and How to Avoid It” netsparker. September 27, 2019. <https://www.netsparker.com/blog/web-security/code-injection/>

[9] Malwarebytes Labs. “Ransomware” Malwarebytes. June 15, 2018. <https://www.malwarebytes.com/ransomware/>

[10] Weisman, Steve. “What is a distributed denial of service attack (DDoS) and what can you about them?” Norton. February 19, 2019. <https://us.norton.com/internetsecurity-emerging-threats-what-is-a-ddos-attack-30sectech-by-norton.html>