Don't Look Back: Horror in Virtual Reality

A Major Qualifying Project
Submitted to the Faculty of
Worcester Polytechnic Institute
in Partial Fulfillment of the Requirements for the
Degrees of Bachelor of Science
in
Computer Science
and in
Interactive Media and Game Development

by
Andrew Fernandes
Mingxi Liu
Zhecheng Song
Jingyu Xie

Advisors
Brian Moriarty, IMGD
Therese Smith, CS

16 December 2021

Abstract

Don’t Look Back is a VR (virtual reality) application which simulates the experience of an office worker being menaced by a supernatural entity. The approach of this visually elusive presence from behind the player’s seat is signaled by environmental cues. Attempts to look directly at the entity make it disappear. The player’s virtual sanity depends on maintaining a balance between their mundane work responsibilities and their anxious curiosity about the advancing horror. The application was built using the Unity engine and XR framework plugin. Playtesting suggested a need for additional design refinement.

Acknowledgements

Our team would like to express our special thanks and gratitude to our advisors Brian Moriarty and Therese Smith for their kind support and helpful guidance throughout the entire project.

We are also grateful for the many people who tested our project and shared their subjective experiences and opinions with us.

Finally, we would also like to thank our teammates for the contributions they made to the completion of our project.

Contents

Figures

1. Introduction

Don’t Look Back is a suspenseful horror game for virtual reality (VR) headsets. In it, the player assumes the role of an office clerk slowly being driven insane by a phantom presence lurking behind their workspace. While sorting folders, the player must keep track of where the phantom is located using audio cues and other information. They lose a lot of sanity if it manages to reach them, but turning around and looking straight at it when it appears makes it vanish and return to the back of the room. However, merely looking behind you also drains your sanity. Balancing your work with a vigilant watch for the supernatural menace is the foundation of this tense experience.

The development process, and the ideas behind it, and an evaluation of the result are presented in this report. Another significant portion of the project involved repeated playtesting of different iterations of our design. This includes the methods and results used for early prototyping to solidify our direction for the game and late betas that helped tune our mechanics. The degree to which we considered and planned multiple stages of playtesting was a key aspect of our team’s production strategy.

2. Background

2.1. Virtual reality

Virtual reality is a field of technologies in which users interact with a digital 3-dimensional space that is simulated around them. This immersive environment is usually implemented with head-mounted screens directly in the user's line of sight, as well as handheld controllers as the main form of input and interaction with the space. These devices have their movement, positions, and rotation tracked so that they can be mapped to the virtual space; the same movements a user would perform to walk around a table or use a door can be practically the same in physical and virtual realities. The field is rapidly growing due to its broad potential as an upcoming generation of human-computer interfaces.

So far, the technology has mostly found use in the game industry. Even in this limited scope, however, the boundaries of the hardware and genre are being pushed. One demonstration of this is *Job Simulator*, a comedic VR game from Owlchemy Labs (2016). In it, players choose one of four levels modeled after specific jobs and workplaces to perform a series of tasks. Activity is filtered through the perspective of a CRT-monitor robots' research into how the jobs are supposed to work.

Combined with simple, colorful graphics and witty writing, *Job Simulator* serves as a casual introduction to VR. In addition to this, the game brings up a potential use of VR in the future; the levels are framed as digital, solo museum exhibits on those jobs for humans that have long forgotten them. While a VR museum might not be economically or technologically reasonable at this time, drastic improvements in VR equipment or AI sentience would not be required.

Another notable example of VR gaming is *SUPERHOT VR* by the SUPERHOT Team (2016). It is somewhere between a sequel and spinoff of the original *SUPERHOT*. What would normally be rapid run-and-gun gameplay is turned into a puzzling, deliberate experience by its central mechanic; time moves only when the player moves, whether it is walking, shooting, turning, or picking up objects. This is built upon with minimalistic graphics, short and quickly resettable levels, and one-shot-kill combat to draw the most out of its gameplay. The VR version puts the player in the character's head with all that entails; time moves when the player literally moves, weapons can and must be precisely aimed, and bullets can be dodged by crouching or leaning out of the way. This layer of physical, focused movement and stillness created an experience as unique and award-winning as the original.

2.2. Unity

Unity is a free-to-use game engine developed by Unity Technologies. It was first published for use in Apple Worldwide Developers Conference 2005, and it could initially only be used in Mac OS X (Brodkin, 2013). Its goal is to make game development accessible to more developers (McWhertor, 2014). In 2007, Unity 2.0 launched with improvements in 3D environment development, real-time shadow, video playback, and more (Cohen, 2007). Unity 3.0 launched in 2010 expanded its support to Android development (Girard, 2010). With the use of WebGL, Unity 5 allows developers to create games that can run on any browser that supports WebGL without the need for players to install any plug-ins (Kumparak, 2014). Unity continued to improve its graphic quality, introducing different render pipelines and allowing developers to modify shaders. In 2020, Unity introduced Mixed and Augmented Reality Studio (MARS) (Sprigg, 2018), which extended developers' abilities to create augmented reality (AR) applications. The tool's VR development support has also been significantly refined.

Unarguably, Unity has become one of the most-used game engines, and currently offers support for 27 release platforms. Its programming API based on C# makes it easy to learn and use by experienced programmers (Fine, 2017). With different tools introduced like tilemap, sprites for 2D games, and texture mipmaps for 3D games, Unity has become a standard tool for making both 2D and 3D games. It has also become a good choice for making VR games.

3. Initial prototyping

Before starting work on a detailed game design, we made two prototype designs with different core mechanics and tested them with volunteers.

3.1. Design 1

In the first design, the phantom behind the player wandered into the room and occasionally moved towards the player. The player was required to work on assigned tasks while avoiding being touched by the phantom. To accomplish this, the player needed to carefully listen to the sounds caused by the phantom to estimate its position. If the player felt the phantom was near enough to pose a threat, they could take an anti-anxiety pill to dissipate the phantom and protect themself from being approached again for a short period of time. The player was also allowed to look back and visually confirm where the phantom was lurking, but doing so shortened the protection time provided by the next pill. The game was lost if the player was touched by the phantom or ran out of pills.

The goal of this design was to survive long enough to complete all required office work.

3.2. Design 2

Design 2 was similar to the first design in the assigned tasks, but differed in how the player could protect themself. Instead of a limited supply of pills, the second design introduced “sanity” as a restorable resource. The player’s sanity level decreased as time passed, and was drastically reduced if the phantom reached them. The player could still turn around and observe the phantom to delay its approach, but doing so decreased their sanity level at a faster rate.

Similar to the first design, sounds created by the phantom offered the player a hint of its position, potentially reducing the number of times the player needed to look back. The goal of this design was to complete all required office work while maintaining at least a shred of sanity. The game was lost if the player’s sanity was reduced to zero.

3.3. Testing methods

After establishing these basic ideas of our game, we created non-digital prototyping tests to help us select the best design choices and make sure they worked well.

3.3.1. Design 1

To simulate Design 1 in the real world, we implemented the following play mechanics:

1. The player was required to focus on reading a given article, which corresponded to their assigned working task. One group member acted as the phantom, and started approaching the player from behind while making odd noises.
2. The player was allowed to clap up to five times during a test session. Each clap represented the action of taking a pill.
3. When a player clapped, the phantom returned to the back of the room and stopped moving for ten seconds.
4. The player was allowed to look back and see where the phantom was to decide when to clap, but the phantom would only pause for seven seconds after subsequent claps.
5. The player’s win condition was to survive for two minutes in this situation while completing their reading assignment.

3.3.2. Design 2

1. The setup of this design was similar to the first. The player had the same reading assignment, and a team member acted as the phantom.
2. To simulate the player's sanity level, an assistant maintained two timers, one for "sanity," and another for "used sanity." The sanity timer was initialized to twenty seconds. The used sanity timer was initialized to zero.
3. If the player focused on reading their assigned article, their sanity timer would steadily decrease. If the player turned back, the phantom stopped moving, and their sanity timer was paused, but their used sanity timer would advance.
4. If the player was touched by the phantom, the sanity timer was stopped, and the used sanity timer was increased by ten seconds.
5. Once the value of the used sanity timer reached the same value as the sanity timer, the game ended.

3.4. Prototype test results

Since these initial tests were informal, with volunteers drawn from our friends and relatives, we didn't provide any post-test surveys, but simply asked for casual feedback and suggestions. Key responses are listed below.

Player 1: Both designs are easy to get the point and find the strategy, but not interesting. For Design 1, it seems players are hard to win because they are unsure about the location of the phantom and the time.

Player 2: Probably hard to balance the number of pills and the time from a designing perspective. Personally suggest adding more items except for pills in Design 1, and adding both buff and debuff to “turning back” action in Design 2.

Player 3: The second idea is more interesting to me, because it adds a new game mechanic named “sanity.”

Based on this feedback, we decided to base our digital prototypes on Design 2. The “sanity” metric seemed less precise and therefore more interesting than a supply of countable pills.

4. Design

Our design process, together with a detailed explanation of game mechanics, are explained below.

Before beginning the implementation of a digital prototype, we integrated our thoughts and ideas into a formal design document (Appendix A).

4.1. Core mechanics

4.1.1. Overview

Don't Look Back takes place in a virtual environment representing an office workspace at night. The player serves as an archivist who needs to classify a pile of files based on their different seals. The archivist is alone, and always remains seated at their desk; no walking is involved.

Throughout the experience, the player is being approached from behind by an elusive, menacing phantom. The player's goal is to complete all assigned tasks while maintaining their virtual sanity.

4.1.2. Game controls

In the game, a player is able to classify files by interacting with the game controllers. The player can grab an object by pressing the button behind the controller, and drop it by releasing the button.

The player can delay the phantom's approach by turning around to look at it. To do this, the player must physically turn their head; the tracking system in the VR headset detects the action in real time and reproduces it in the virtual world.

4.2. Game elements

4.2.1. Phantom movement

The movement mechanic for the phantom was inspired by the children's game “Red Light, Green Light” (also known as “Statues”) in which a selected person stands at the end of a playing field, turning their back to the remaining players (Mann Susan, 2019). On cue, these players begin running as fast as they can toward the target player, but if the target turns to face them, all runners must immediately freeze in place.

In *Don’t Look Back*, the phantom appears in a door located in the back wall of a 10 x 10 unit room, and moves at a speed of 0.5 units per second. In the current game, the phantom can only move along one of several predetermined paths (see Section 4.5). Paths are randomly chosen for each play session, and not all paths lead directly to the player, increasing the variety and uncertainty of its progress.

Players can hear the breath and footsteps of the phantom, indicating its presence and position. We hoped to produce a suspenseful experience by forcing players to turn and face the phantom, although their rate of sanity loss is doubled if they do so.

4.2.2. Design modifications

To ease the challenge faced by players, we added the ability to eliminate the phantom for a short period of time. When players turn back and stare at the phantom, it vanishes and grants a five-second “grace period,” allowing the player to focus on working until phantom reappears. However, staring at the phantom increases the rate of sanity loss by four times, so players still have to carefully decide the right moment to use this ability.

While developing the game, we noticed several design issues that needed to be addressed:

1. In the previous design, players could see the phantom every time they turned to look. This approach seemed ineffective because no one would remain scared by something normally visible. To increase the tension, we added an “invisibility” mechanic that makes the phantom appear only when it collides with objects in the scene, or draws very close to the player.
2. Previously, players only needed to look back once to make the phantom retreat. This easy tactic worked against the stressful atmosphere we hoped to create. Therefore, each successive round increases the amount of “staring” needed to delay the phantom from zero to three seconds. This change will force players to look back more frequently when hearing its sound, since it will only appear during a collision.

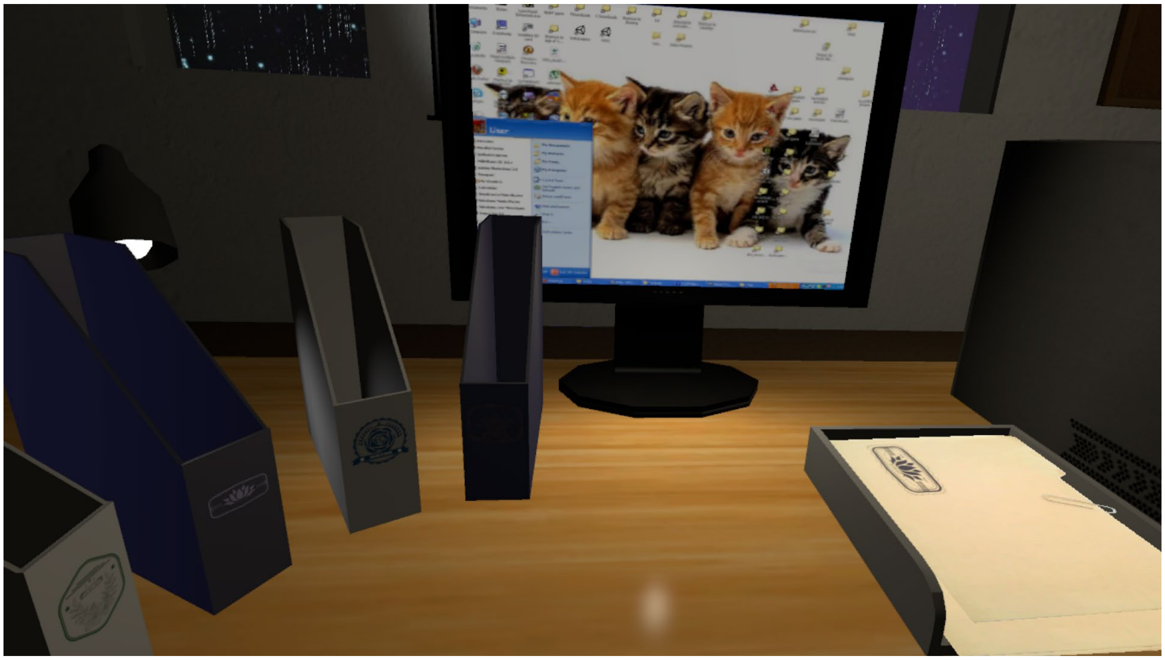


Figure 1. Player's virtual workspace. Source: Screen capture.

4.2.3. Work

In the game, players play the role of a night-shift archivist who's trying to classify files while seated at a desk (see Figure 1). The folder on the right-hand side of players generates files every 4.5 to 5.5 seconds, and players must classify them based on the style of seal on the file. Each of the four types of seals may appear on any of three randomly selected locations on the file. During the game, players only need to pick up one file, find the seal on it, and put the file into the correct folder. Each correctly sorted file recovers 6% of the player's sanity. Levels can only be survived by correctly sorting a specific number of files, depending on the difficulty of the level. In the first level, the target number of correct classifications is twenty.

4.2.4. Sanity bar

Many games use the concept of a “sanity bar” to give players an insight of how close the character they are controlling is to losing their mind (Richter, 2021). It serves as a measure of fear, and we have incorporated this mechanic to inform players about their status.

The player’s initial sanity is 100% and decreases at a steady rate of 1% per second. If the phantom reaches the player, they immediately lose 50% of their initial sanity. Sanity increases by 6% each time the player correctly sorts a file. If the player looks back, their sanity decreases at a rate of 2% per second, or at 4% per second if they stare continuously at the phantom.

4.2.5. Distractions

To diversify the game experience, we added environmental distractions.

If sanity falls below 80%, a lamp in the office environment begins blinking three times per second, increasing to five times per second if sanity falls lower than 60%. If sanity falls below 40%, the lamps behind and in front of the player turn off and an ominous heartbeat sound begins to play, making it harder to concentrate on file classification.

We also added sound distractions that prevent the player from precisely locating the phantom. Sound effects like thunder and ringing phones increase the difficulty of locating the phantom’s position.

4.3. Level design

We plan to design three levels for the game, which increase in difficulty. Players can complete the whole game only if they survive all three levels. Our project plan focused on developing and testing the second level. We hoped to make changes based on player feedback and complete the development of the remaining of the levels if time allowed.

In the first level, the difficulty is easy. If players look back, they should be able to see the phantom itself when it approaches them. Once the phantom goes away new ones will appear at a low frequency. During the game there won't be any environmental distractions for players and the sanity bar is always visible for players.

In the next level, the difficulty is medium. The phantom won't always be visible, but other signs of them can still be seen in the environment. During the game there will be some sound or visual distractions that can interfere in players' judgements or their working progress. In this level, the sanity bar is still visible to players but phantoms will appear slightly more often than they do in the first level.

In the last level, the difficulty is hard. The phantom is mostly invisible, so players can only see indirect signs of its locations. Environmental distractions will appear frequently. The sanity bar is not visible to players and they need to track it themselves. Once the phantom disappears, new phantoms will come out much more often.

4.4. Room layout

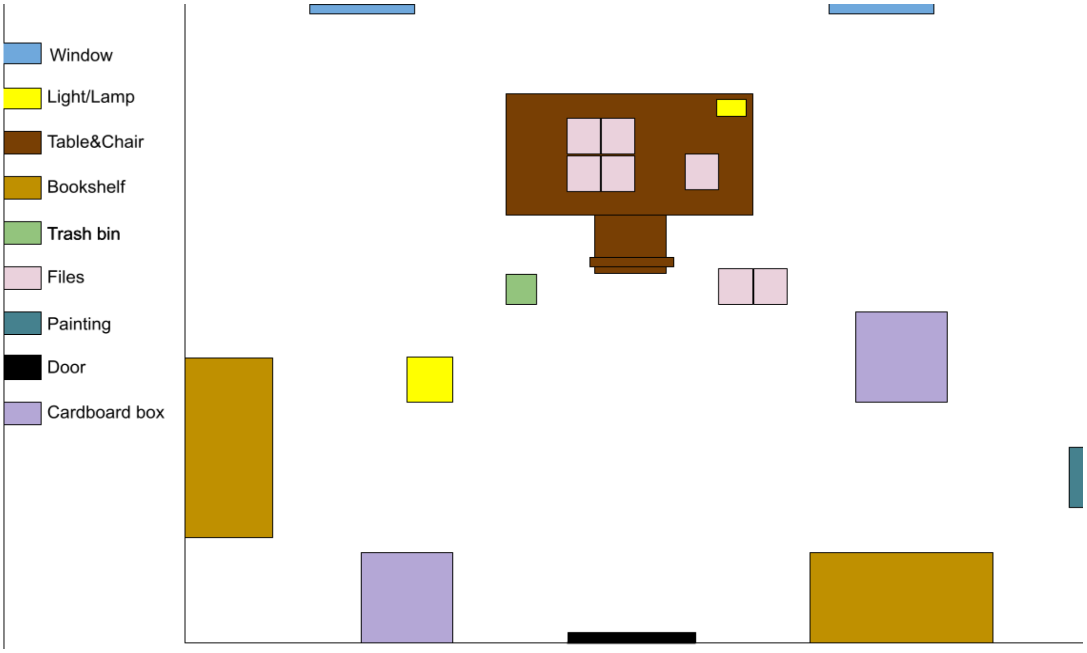


Figure 2. Initial room layout. Source: Google Drawings screen capture.

4.4.1. Initial design

As mentioned previously, the phantom follows the pre-designed paths to approach the player. Therefore, we first designed a room layout with 10 x 10 square units large as shown in Figure 2, with different objects placed behind the player, then made paths based on the room layout.

In the first version of our room layout design, the room consisted of nine types of objects: windows, lamps, the table with a chair, bookshelves, the trash bin, files, the painting, card boxes, and the door. Among all these objects, the table with a chair represents the location of the player, and the door is the spawn point of the phantom. Additionally, there are hint objects (lamps, bookshelves, files, and cardboard boxes) that the phantom can collide with to play sound effects. The painting and the trash bin are objects for decoration only. To avoid the confusion caused by audio from directly behind the player, all objects are distributed on average on either the left or right back side of the player. Thus, it can provide sufficient chances for the player to locate where the phantom is.

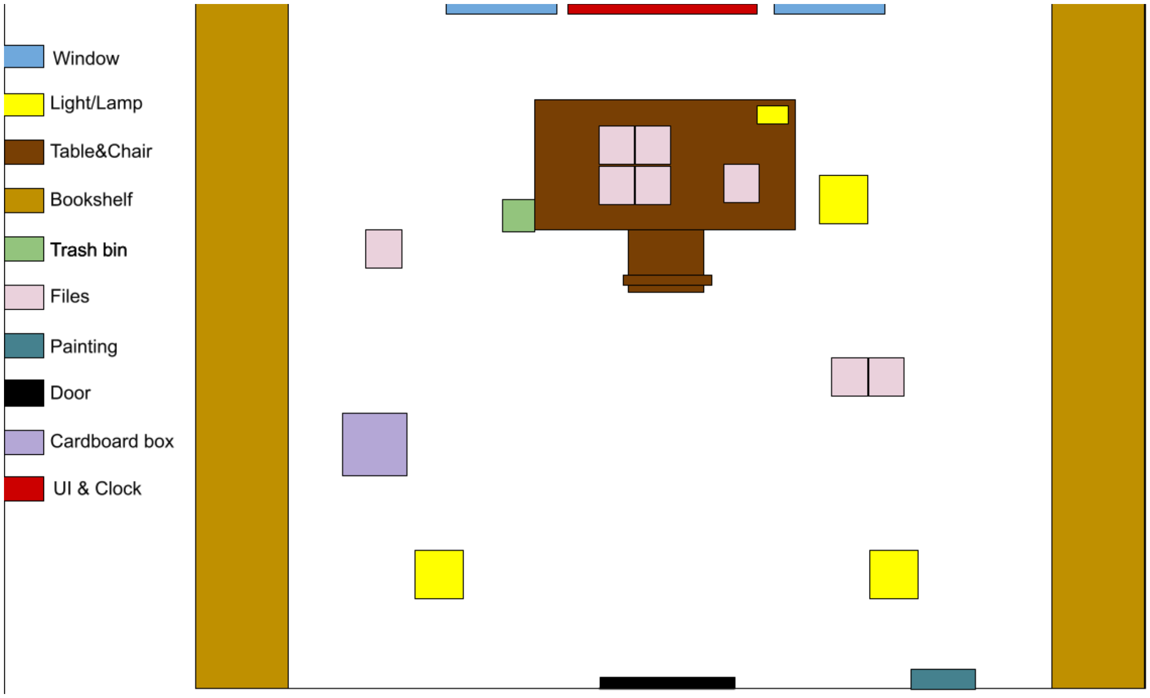


Figure 3. Edited room layout. Source: Google Drawings screen capture.

4.4.2. Edited layout

The design has been revised after the phantom mechanic revision. As previously mentioned, the phantom will appear only during collision, so chances for players to eliminate the phantom are too easy to catch it in for the previous room layout. Also, too many collision sounds mitigate the horrible atmosphere. Thus, we extended the distance between objects, as shown in Figure 3, so players only have a limited number of chances to eliminate the phantom. Doing so also helps us stress players with unusual and unpredictable sound effects. Furthermore, we added the UI whiteboard into our new room layout, so players can see their current sanity bar, the current number of completed works and the total number of works to complete. Above the UI whiteboard is a clock with loop sound effects.

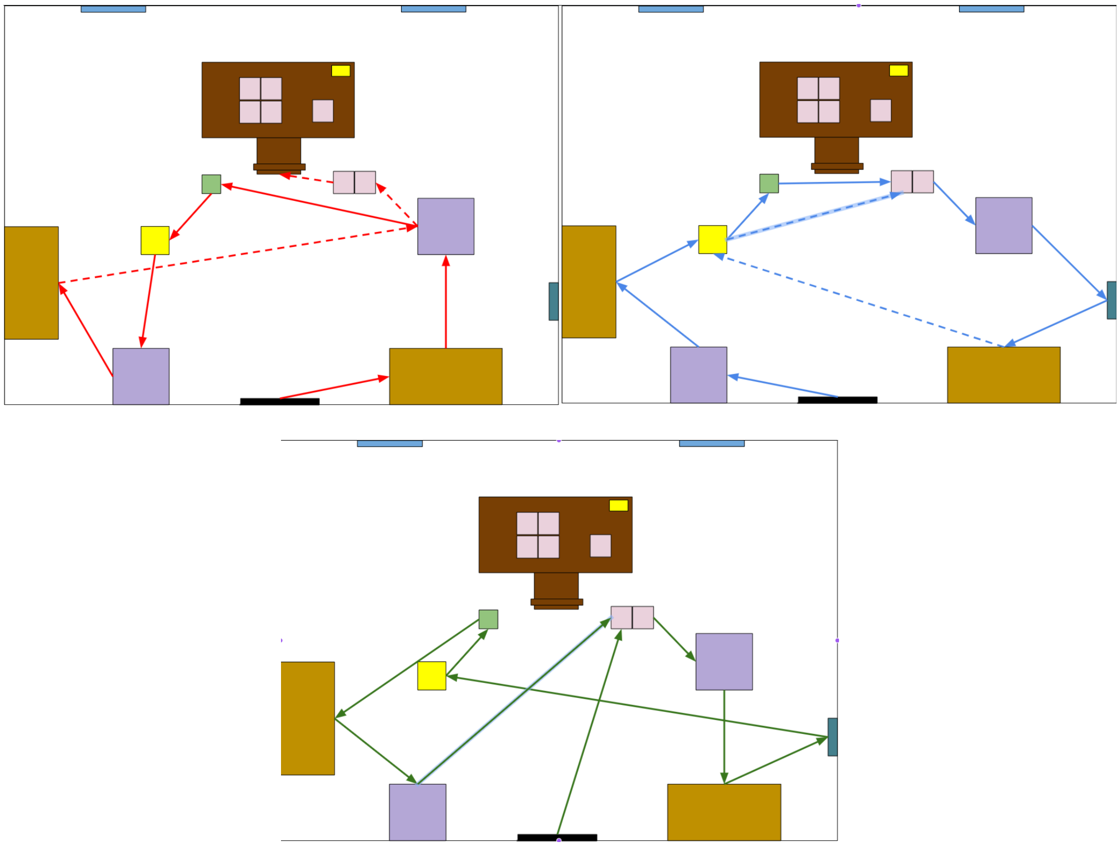


Figure 4. Initial path designs. Source: Google Drawings screen capture.

4.5. Phantom path designs

For phantom paths design, we initially made three different paths based on room layout 1 as shown in Figure 4. However, since the room layout was redesigned, we designed a newer version of paths based on room layout 2. Besides, the long paths above take more time to get near the player than we expected, so we shorten every path length to four or five turning points in total. Thus, players need to listen to the sound effects behind more frequently, otherwise they will be touched by the phantom in a short time.

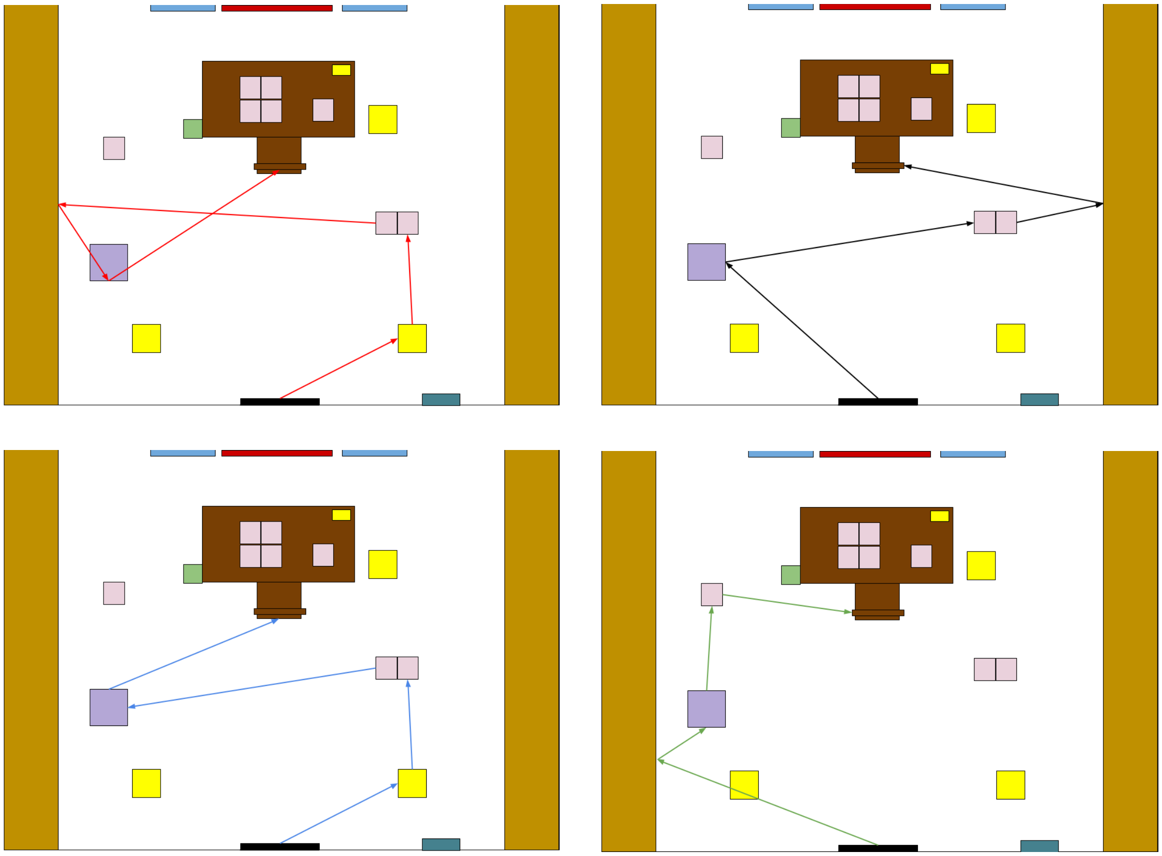
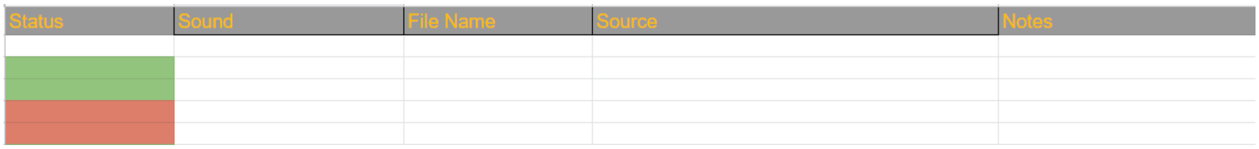


Figure 5. Updated path designs. Source: Google Drawings screen capture.

We also increased the number of paths from three to four to make sure that the variability of paths won't decrease with shorter length. All paths are shown in Figure 5.

4.6. Audio design



Status	Sound	File Name	Source	Notes
🟩				
🟩				
🟥				
🟥				

Figure 6. Sound asset list format. Source: Google Sheets screen capture.

4.6.1. Audio asset list

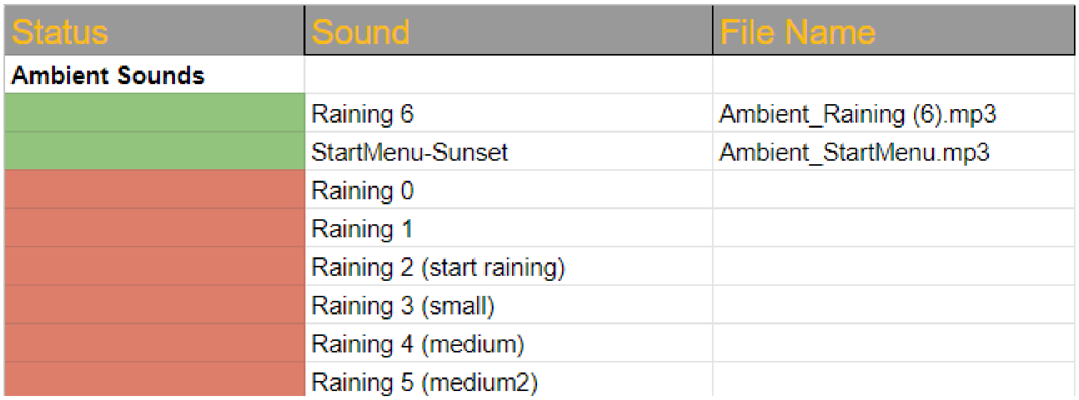
Before starting work on searching audio, we made a sound asset list to track our current sound demand status and references. Its structure is shown in Figure 6. See Appendix F for the complete list and sources.

The list is divided into five columns: Status, Sound, Source, Filename, and Notes. Status means whether the sound is already found and completed. The green color means we have finished all search, edit, and import steps of the sound, so it's complete. The red color means the sound is either not found, not edited, or not imported, and specifically which step is incomplete will be marked in Notes. The Sound is mainly the description of the sounds we are looking for. The Source means the link to where the sound is found, which benefits our reference work later. The Filename represents the current filename of the sound in google drive, so it's easier for us to find where the sound is. The Notes contains other notifications about the sound, like which part of the sound needs to be edited.

4.6.2. Sound categories

After finishing the audio asset list set up, we separated all audio demands into three categories based on their attributes and started to seek open-source sound effects from freesound.org and www.ear0.com.

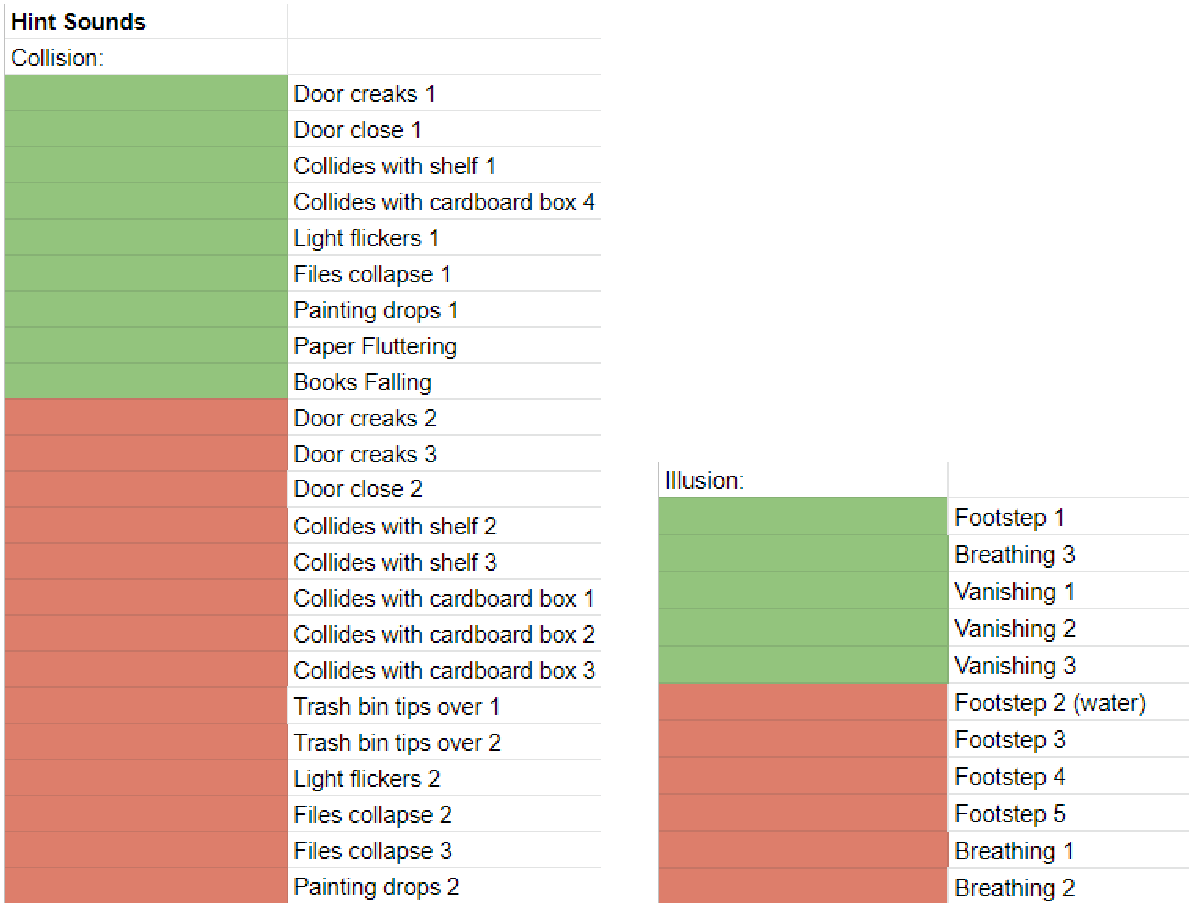
The first category is the ambient sounds that affect the atmosphere of the game. Since any one scene only needs one ambient sound to play on a loop, we only need two in total, one for the start menu, and another for in-game play. The ambient sound of the start menu is targeted in providing a relaxing mood to players before the game, so we decided to use the sound that represents the atmosphere of sunset.



Status	Sound	File Name
Ambient Sounds		
🟩	Raining 6	Ambient_Raining (6).mp3
🟩	StartMenu-Sunset	Ambient_StartMenu.mp3
🟥	Raining 0	
🟥	Raining 1	
🟥	Raining 2 (start raining)	
🟥	Raining 3 (small)	
🟥	Raining 4 (medium)	
🟥	Raining 5 (medium2)	

Figure 7. Ambient sounds category sample. Source: Google Sheets screen capture.

For the in-game play, we chose the pure raining ambient sound to maximize the horrible atmosphere in the game. To provide enough choices for selection, we found several sounds for each need (Figure 7) and then selected the appropriate one from them. The final selected and imported sound will be marked green, so it’s clear which sound has been used.

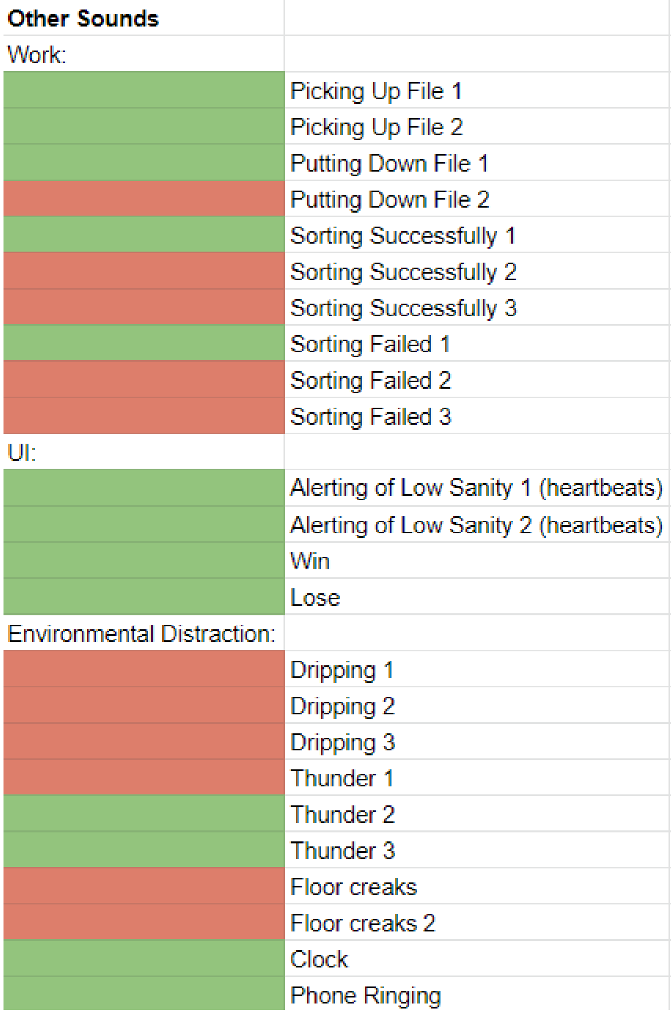


Hint Sounds	
Collision:	
🟩	Door creaks 1
🟩	Door close 1
🟩	Collides with shelf 1
🟩	Collides with cardboard box 4
🟩	Light flickers 1
🟩	Files collapse 1
🟩	Painting drops 1
🟩	Paper Fluttering
🟩	Books Falling
🟥	Door creaks 2
🟥	Door creaks 3
🟥	Door close 2
🟥	Collides with shelf 2
🟥	Collides with shelf 3
🟥	Collides with cardboard box 1
🟥	Collides with cardboard box 2
🟥	Collides with cardboard box 3
🟥	Trash bin tips over 1
🟥	Trash bin tips over 2
🟥	Light flickers 2
🟥	Files collapse 2
🟥	Files collapse 3
🟥	Painting drops 2

Illusion:	
🟩	Footstep 1
🟩	Breathing 3
🟩	Vanishing 1
🟩	Vanishing 2
🟩	Vanishing 3
🟥	Footstep 2 (water)
🟥	Footstep 3
🟥	Footstep 4
🟥	Footstep 5
🟥	Breathing 1
🟥	Breathing 2

Figure 8. Hint sounds category sample. Source: Google Sheets screen capture.

The second category is the hint sounds that provide location information of the phantom. Based on whether that sound is triggered by the collision, or it went with the phantom, we classify them to collision sounds and phantom sounds. The collision sounds will be played each time the phantom collides with an object behind the player, so it depends on the object under collision. Based on our room layout design, we made the collision sound effects demand list shown in Figure 8. Another type, named phantom sounds, will be played in time with the movement of the phantom, like the breath or vanish sounds. In our first version of the audio demand for phantom, footstep noises were a significant part. However, since our art design for the phantom changed from moving with feet to floating in the air, the footstep was abandoned.



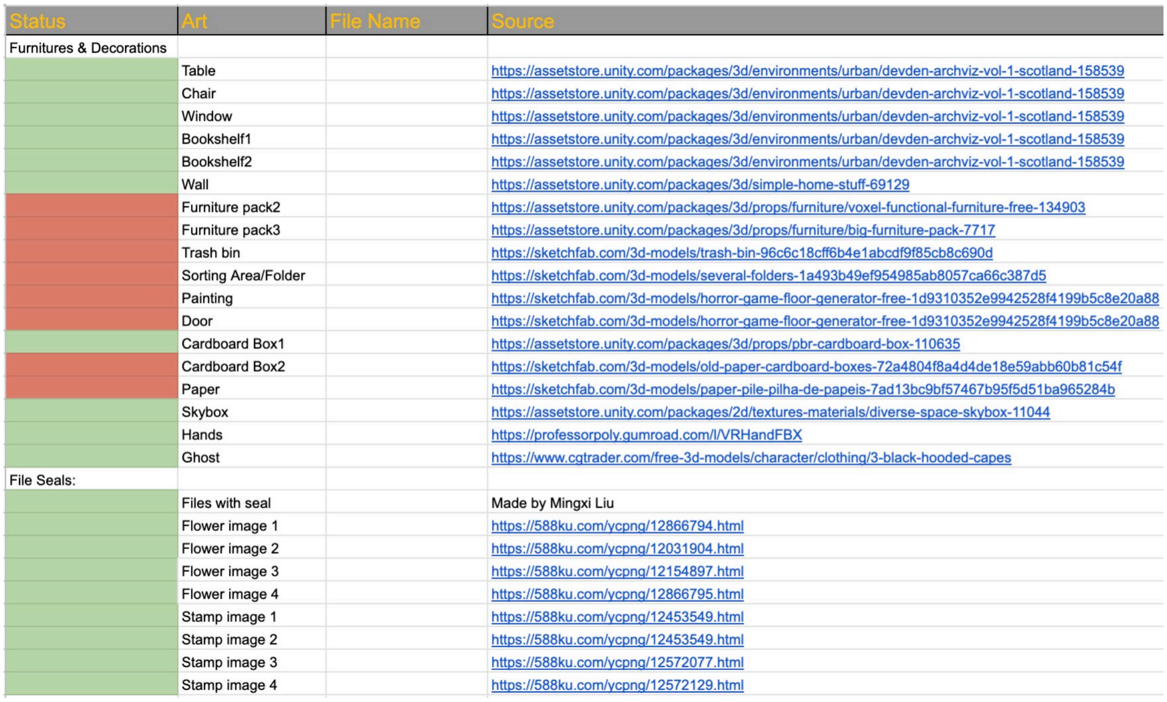
Other Sounds	
Work:	
🟩	Picking Up File 1
🟩	Picking Up File 2
🟩	Putting Down File 1
🟥	Putting Down File 2
🟩	Sorting Successfully 1
🟥	Sorting Successfully 2
🟥	Sorting Successfully 3
🟩	Sorting Failed 1
🟥	Sorting Failed 2
🟥	Sorting Failed 3
UI:	
🟩	Alerting of Low Sanity 1 (heartbeats)
🟩	Alerting of Low Sanity 2 (heartbeats)
🟩	Win
🟩	Lose
Environmental Distraction:	
🟥	Dripping 1
🟥	Dripping 2
🟥	Dripping 3
🟥	Thunder 1
🟩	Thunder 2
🟩	Thunder 3
🟥	Floor creaks
🟥	Floor creaks 2
🟩	Clock
🟩	Phone Ringing

Figure 9. Other Sounds category sample. Source: Google Sheets screen capture.

The third sounds category is miscellaneous sounds that we need in our game as shown in Figure 9. In this category, we listed all other needed sounds for UI, interaction, alerting, and distraction. For UI, we need a heartbeat sound to stress the atmosphere when the player has a sanity level lower than 45%, the sound of winning and losing the game. In addition, we need sound effects triggered when picking up and putting down files, as well as the environmental distraction sounds like thunder that will be useful in future development. See Appendix G for the complete audio asset list with sources.

However, not all of the sounds listed above can be found and used directly. Two sounds for the ambient sound category need to be played on a loop, so the starting and ending should fluently connect with each other without any disruption. Therefore, we clipped a part of the ambient sound through using version 3.1.1 of Audacity(R) recording and editing software (The Audacity Team, 1999-2021) if we found it abrupt, and reconnected the sifted sounds. For the hint sound and other sound categories, we successfully found sounds like books falling, paper fluttering, and monster breathing in freesound.org and ear0.com, but most of the sounds failed to meet our expectations.

To solve the problem, we used three strategies. To begin with, we tried clipping the existing sounds and then connecting them to make a new sound effect. For example, we clipped and connected sounds of door creaking and suddenly closing to inform the appearance of the phantom, because the spawn point of the phantom is at the door. Furthermore, we shorten the length of the sound to make it appropriate for our game. This strategy is used in the vanishing sound of the phantom. We shorten the length of the sound for phantom vanishing from five seconds to two seconds to avoid the discrepancy between audio and performance. Additionally, we tried using clips from other sound effects to fulfill our demand if we are unable to find the suitable sound. This strategy has been used in making the sound of colliding with a cardboard box. Since all found sounds of that failed to create a sense of something falling downstairs, we clipped the sound of a tipping over trash bin and used part of it.



Status	Art	File Name	Source
Furnitures & Decorations			
🟩	Table		https://assetstore.unity.com/packages/3d/environments/urban/devden-archviz-vol-1-scotland-158539
🟩	Chair		https://assetstore.unity.com/packages/3d/environments/urban/devden-archviz-vol-1-scotland-158539
🟩	Window		https://assetstore.unity.com/packages/3d/environments/urban/devden-archviz-vol-1-scotland-158539
🟩	Bookshelf1		https://assetstore.unity.com/packages/3d/environments/urban/devden-archviz-vol-1-scotland-158539
🟩	Bookshelf2		https://assetstore.unity.com/packages/3d/environments/urban/devden-archviz-vol-1-scotland-158539
🟩	Wall		https://assetstore.unity.com/packages/3d/simple-home-stuff-69129
🟥	Furniture pack2		https://assetstore.unity.com/packages/3d/props/furniture/voxel-functional-furniture-free-134903
🟥	Furniture pack3		https://assetstore.unity.com/packages/3d/props/furniture/big-furniture-pack-7717
🟥	Trash bin		https://sketchfab.com/3d-models/trash-bin-96c6c18cff6b4e1abcdf9f85cb8c690d
🟥	Sorting Area/Folder		https://sketchfab.com/3d-models/several-folders-1a493b49ef954985ab8057ca66c387d5
🟥	Painting		https://sketchfab.com/3d-models/horror-game-floor-generator-free-1d9310352e9942528f4199b5c8e20a88
🟥	Door		https://sketchfab.com/3d-models/horror-game-floor-generator-free-1d9310352e9942528f4199b5c8e20a88
🟩	Cardboard Box1		https://assetstore.unity.com/packages/3d/props/pbr-cardboard-box-110635
🟥	Cardboard Box2		https://sketchfab.com/3d-models/old-paper-cardboard-boxes-72a4804f8a4d4de18e59abb60b81c54f
🟥	Paper		https://sketchfab.com/3d-models/paper-pile-pilha-de-papeis-7ad13bc9bf57467b95f5d51ba965284b
🟩	Skybox		https://assetstore.unity.com/packages/2d/textures-materials/diverse-space-skybox-11044
🟩	Hands		https://professorpoly.gumroad.com/l/VRHandFBX
🟩	Ghost		https://www.cgtrader.com/free-3d-models/character/clothing/3-black-hooded-capes
File Seals:			
🟩	Files with seal		Made by Mingxi Liu
🟩	Flower image 1		https://588ku.com/ycpng/12866794.html
🟩	Flower image 2		https://588ku.com/ycpng/12031904.html
🟩	Flower image 3		https://588ku.com/ycpng/12154897.html
🟩	Flower image 4		https://588ku.com/ycpng/12866795.html
🟩	Stamp image 1		https://588ku.com/ycpng/12453549.html
🟩	Stamp image 2		https://588ku.com/ycpng/12453549.html
🟩	Stamp image 3		https://588ku.com/ycpng/12572077.html
🟩	Stamp image 4		https://588ku.com/ycpng/12572129.html

Figure 10. Art asset list sample. Source: Google Sheets screen capture.

4.7. Art design

4.7.1. Art asset list

In addition to the asset list of required audio, we created an asset list for visual art in a similar format (Figure 10). See Appendix G for the full list and source references.

4.7.2. Art assets

During the game play, one of the core art demands is four file seal types. Players need to classify files based on the seal type appearing on each of them. In our initial design of seals, we wanted to make the classification as simple as possible, so we were looking for graphic symbols that are easy to identify and are aesthetic at the same time. We finally decided to use different flowers as the graphic symbol to differentiate different seals. We found four copyright-free flower images and four seal shapes online. To imitate the seal style in the real world, we also added dates on each seal.

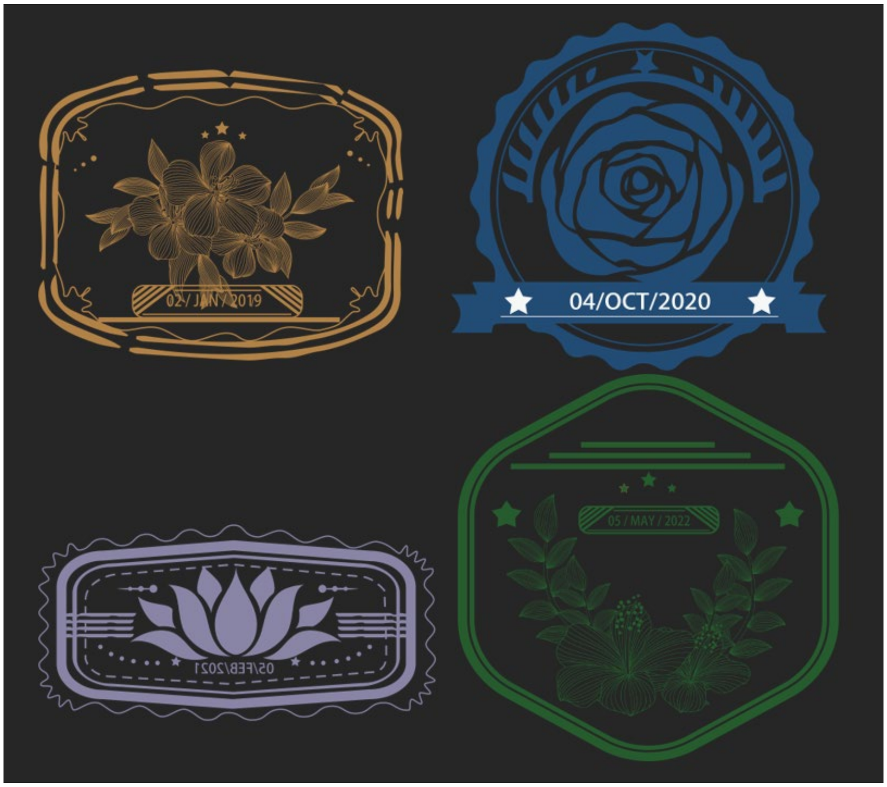


Figure 11. Four seals. Source: Original concept art.

The next step was to combine all assets together with Adobe Illustrator and put seal style onto the file model. Each seal, as shown in Figure 11, has a unique color so that players can differentiate them quickly.

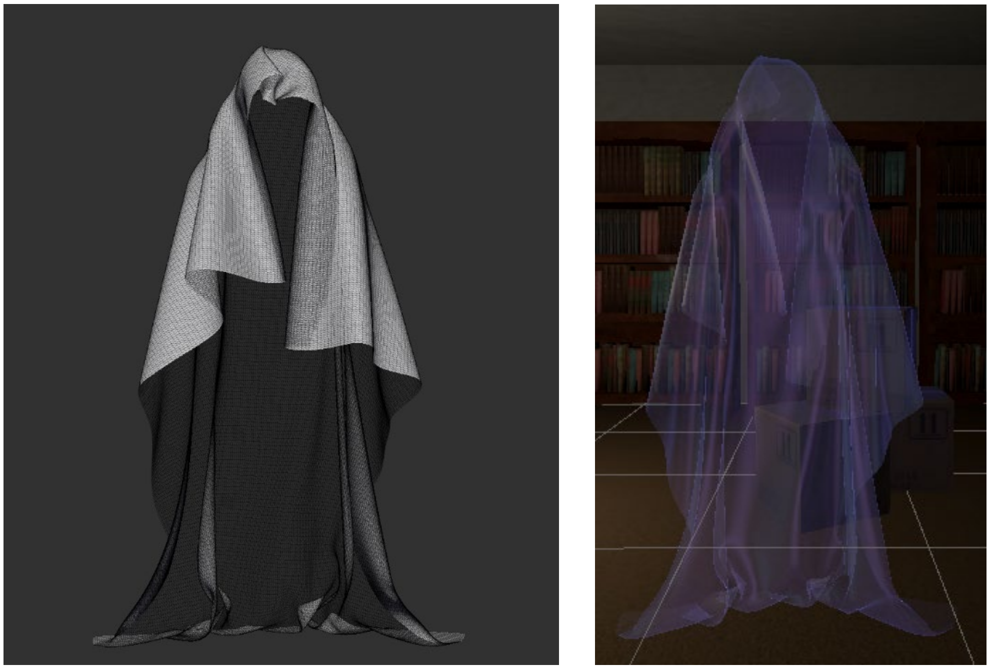


Figure 12. Phantom model and prefab. Sources: Unity screen capture.

Another core art demand is the prefab of the phantom. Based on our design, the phantom is invisible most of the time except for colliding with something. Therefore, the phantom prefab should give players a sense of being haunted by something. We chose the ghost model as shown on the left side of Figure 12, and added a half-opaque material to form the phantom prefab shown on the right.

5. Development

For our VR game, we started with a URP template in Unity and then continued by adding a XR plug-in that works as an API for VR development in 3D.

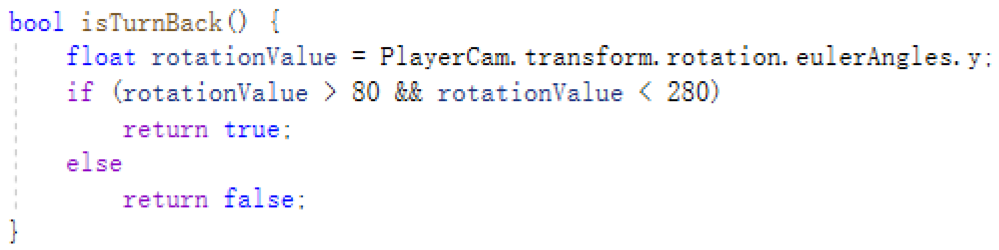
5.1. Plug-ins

To get VR working in Unity, we need a plug-in that allows receiving inputs from VR devices. In the past, the plug-in provided by a specific VR manufacturer could only work with the VR devices produced by that company. However, over years of development in the VR industry, a universal API for VR development, XR Framework plug-in, has been created. XR Framework supports most of the current VR devices like HTC Vive, and Oculus. You only need to choose which platform you want to develop on after installing the XR plug-in, and it will install the necessary plug-in for you. You can also switch between platforms very easily without the need of changing your code.

We also use some supportive plug-ins in our game. For example, one plug-in we get from Unity asset store can simulate rain using particle effects.

5.2. Player controls

We used the API provided by XR plug-in for our player control. It allows basic movements for the player like turning their head, or grabbing and using objects by their controller. It also has an event system for reporting different inputs like the main trigger, or grab trigger. Therefore, we can easily allow players to grab any object by adding a script called “XRGrabInteractable” provided by the XR plug-in. It also contains the functionality that allows us to trigger the function we want when the object is being grabbed or used. We also used the raycasting function provided by the plug-in to interact with UI components.



```
bool isTurnBack() {
    float rotationValue = PlayerCam.transform.rotation.eulerAngles.y;
    if (rotationValue > 80 && rotationValue < 280)
        return true;
    else
        return false;
}
```

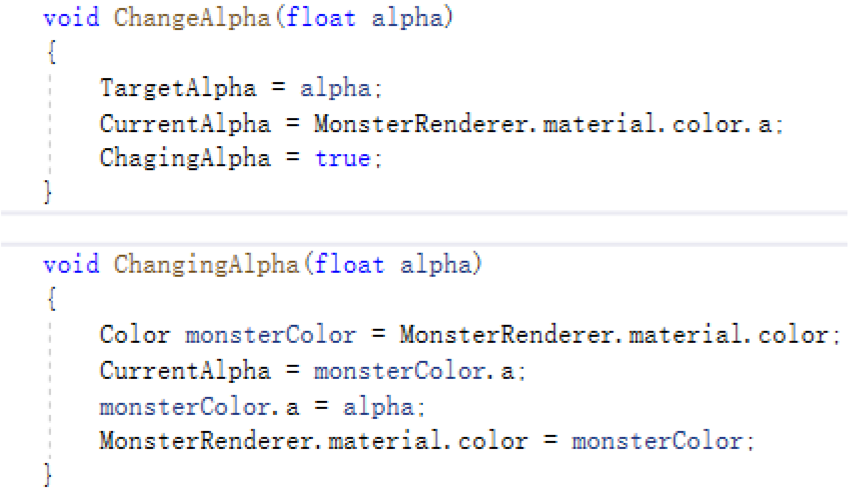
Figure 13. Script to check turning back. Source: Unity screen capture.

5.3. Sanity

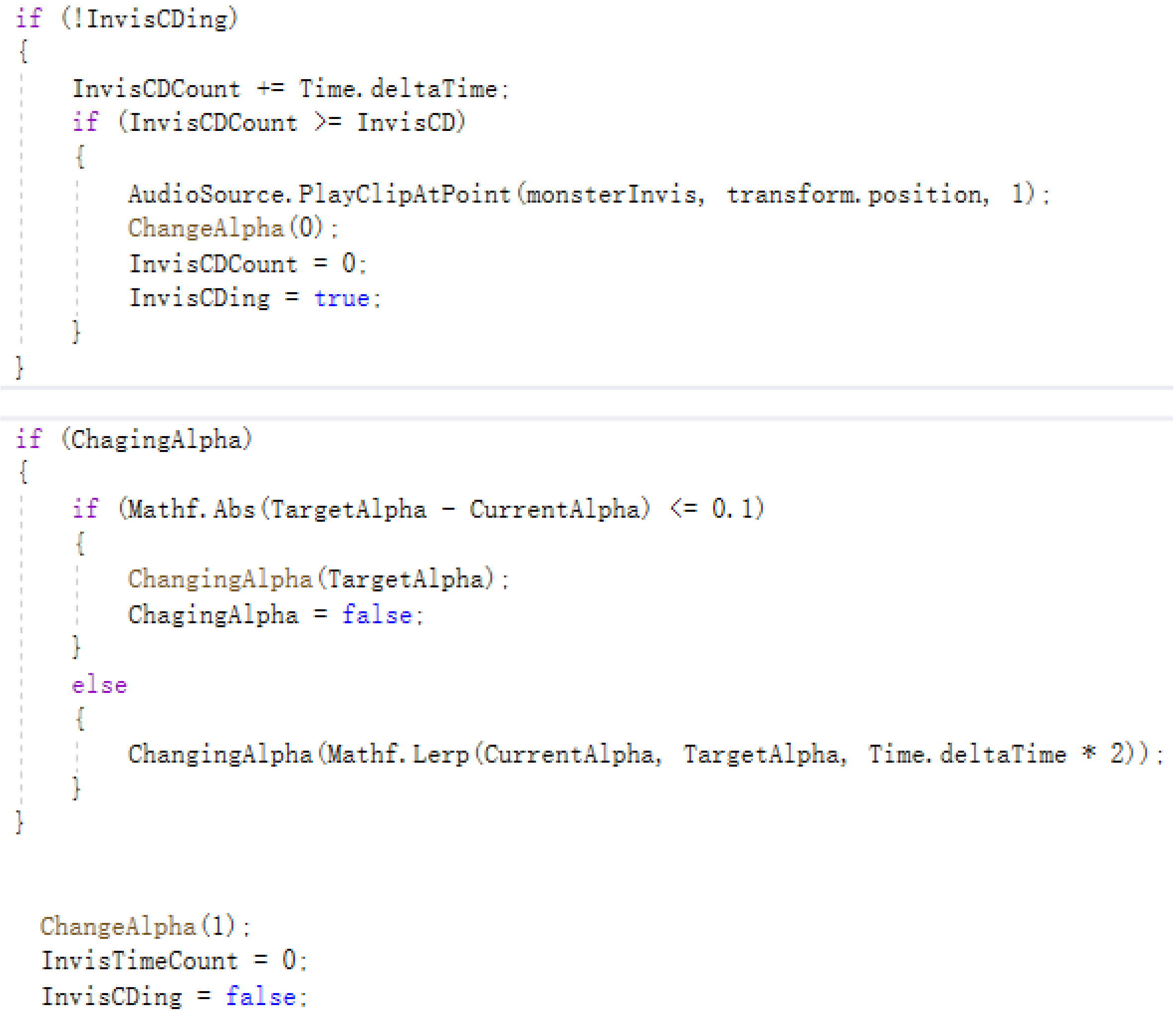
Sanity is a system in our game that works in a way similar to health. It decreases slightly as time passes, and decreases greatly if the player is attacked by the enemy. A sanity object is linked to all other objects that are able to increase or decrease sanity. Those objects can call the public function “UpdateSanity(float)” with a given positive or negative float number. If the float number is negative, the sanity will decrease, and the sanity will increase if it’s positive. Additionally, the sanity decreases at a faster rate if the player is looking back. In the “Update()” function of the player’s logic, we detect the rotation angle of the camera to check if the player is turning back as shown in Figure 13. Furthermore, the UI of sanity is displayed on a screen in front of the player, so the player can know how much sanity is left. Instead of using the UI settings provided by Unity, which are based on screen position, we decided to base it on the world position. We will discuss the UI in detail in section 5.6 later.

5.4. Phantom logic

The enemy in our game is an phantom, a ghost-like monster, which means it's invisible all the time except during collision. To make an appearing and disappearing effect of the phantom, we gradually change the opacity of the phantom object from 0 to 1 when it collides with other objects, and gradually change it back to 0 after a short period of time as shown in Figure 14 below.



```
void ChangeAlpha(float alpha)
{
    TargetAlpha = alpha;
    CurrentAlpha = MonsterRenderer.material.color.a;
    ChagingAlpha = true;
}

void ChangingAlpha(float alpha)
{
    Color monsterColor = MonsterRenderer.material.color;
    CurrentAlpha = monsterColor.a;
    monsterColor.a = alpha;
    MonsterRenderer.material.color = monsterColor;
}
```



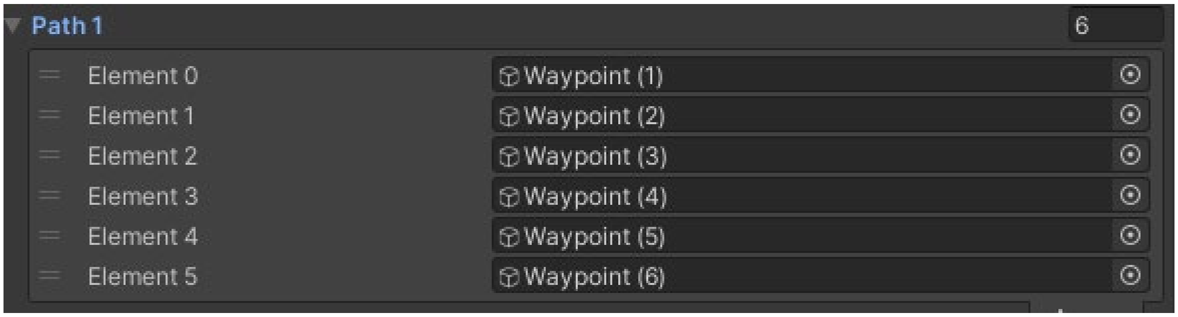
```
if (!InvisCDing)
{
    InvisCDCount += Time.deltaTime;
    if (InvisCDCount >= InvisCD)
    {
        AudioSource.PlayClipAtPoint(monsterInvis, transform.position, 1);
        ChangeAlpha(0);
        InvisCDCount = 0;
        InvisCDing = true;
    }
}
```

```
if (ChagingAlpha)
{
    if (Mathf.Abs(TargetAlpha - CurrentAlpha) <= 0.1)
    {
        ChangingAlpha(TargetAlpha);
        ChagingAlpha = false;
    }
    else
    {
        ChangingAlpha(Mathf.Lerp(CurrentAlpha, TargetAlpha, Time.deltaTime * 2));
    }
}
```

```
ChangeAlpha(1);
InvisTimeCount = 0;
InvisCDing = false;
```

Figure 14. Script of opacity change. Source: Unity screen capture.

The AI of our phantom is simple but effective. We decided to focus on making the environment scarier to immerse players in our game. At the start of the game, the phantom would randomly choose a path from pre-designed paths mentioned in section 4.5 by using the "Random.Range()" function provided by Unity. This random generator can generate either an int or a float over a given range. The range is from 0 to the number of paths we have, which is 4 right now. After picking up the path, our phantom has two kinds of action in the game — navigation and collision.



Path 1 6

Element 0	Waypoint (1)
Element 1	Waypoint (2)
Element 2	Waypoint (3)
Element 3	Waypoint (4)
Element 4	Waypoint (5)
Element 5	Waypoint (6)

Figure 15. Path consisting of waypoints. Source: Unity screen capture.

First, it follows the given path to patrol around the room. One specific path consists of many stopping points that are pre-stored in a list in the phantom’s script shown in Figure 15.

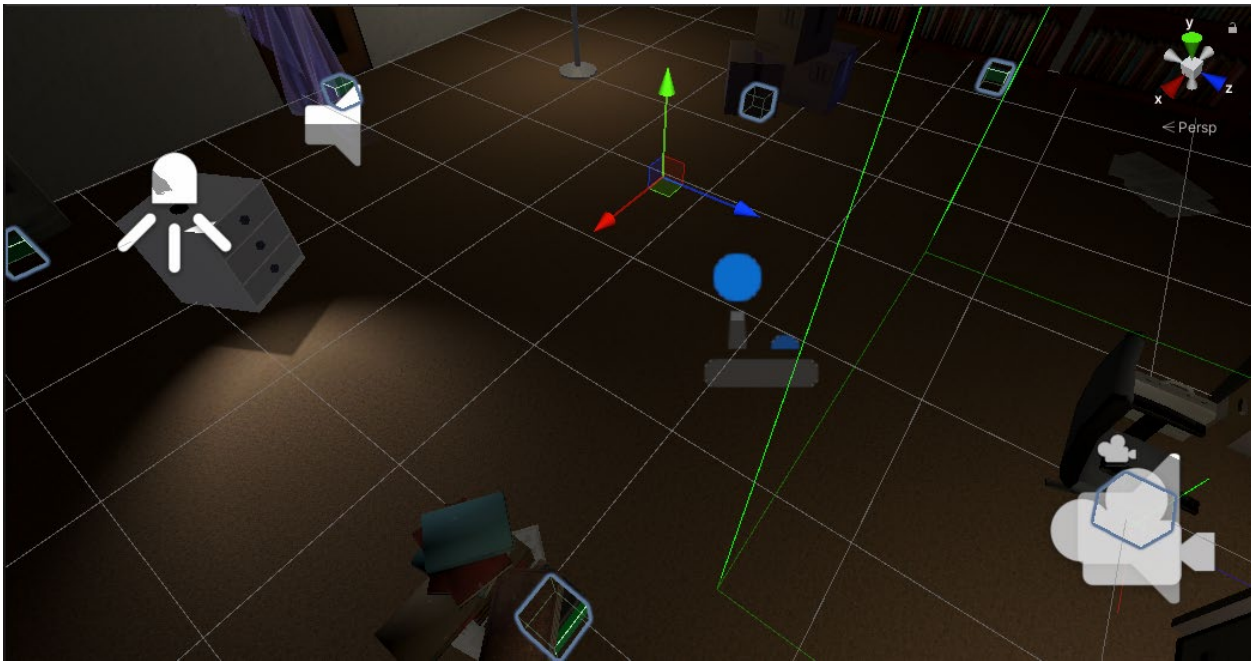


Figure 16. Point objects created in game scene. Source: Unity screen capture.

The stopping points are created in the scene shown in Figure 16.

The phantom starts its path navigation by teleporting to the first point in the list and then follows the path to the next point. During the navigation between points, the phantom first turns towards the next stopping point, then walks to it at a given speed. After reaching that stopping point, it repeats the process until reaching the end point of the path.

In order to turn the phantom smoothly, we use the function “Mathf.lerp(a,b,t)” which returns a value interpolated between “a” and “b” by the function “a + (b - a) * t”. In this way, we can control how much it turns per frame and make it smooth. When the phantom reaches the end of the path, or is stared at by the player for 3 seconds in total, the phantom enters a cooldown for 5 seconds before restarting at a random path again. Instead of deleting the phantom and respawning it, we decided to move the phantom to a place where the player can't see it, and have it count the cooldown for itself.

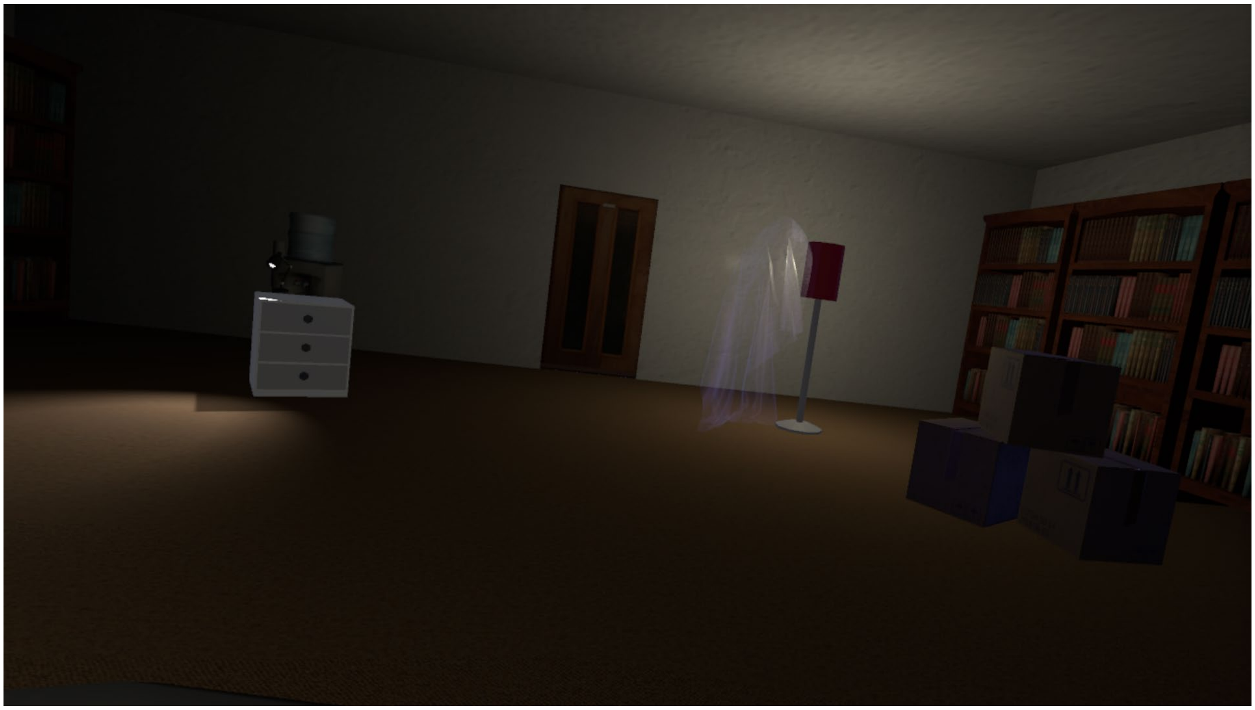
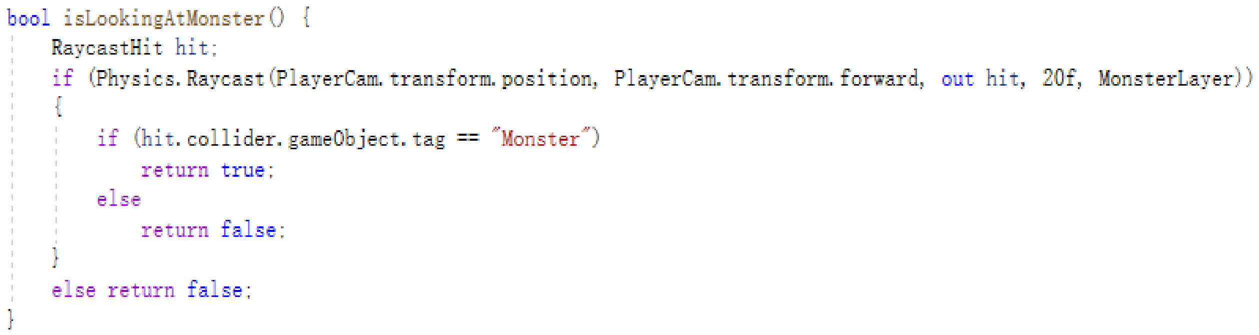


Figure 16. Phantom collision. Source: Screen capture.

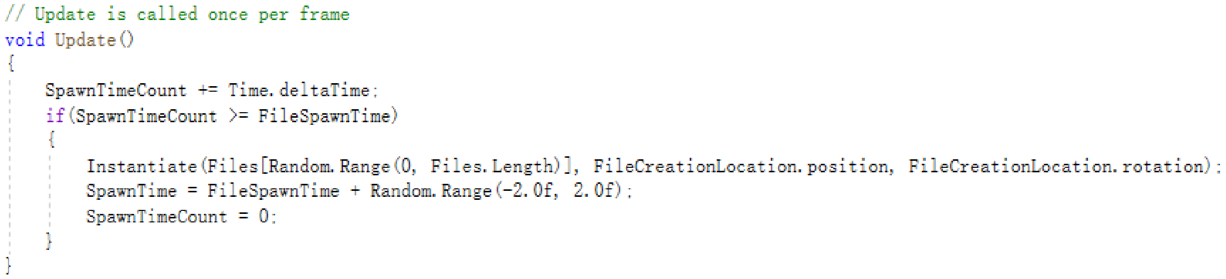
Second, the phantom interacts with the objects in the environment. It plays the designed sound effect whenever it touches an interactable object or makes the lamp flash as shown in Figure 16. To have the light flash, we attach a script on the lamp that repeatedly turns the light source off and on. We use the "OnTriggerEnter()" function to determine whether or not the phantom touches the lamp, or any other interactable object. Then, we examine the tag of the object it collides with to see if that's an interactable object. Finally, we check the name of the object and perform an phantom act accordingly. The phantom also turns visible for a very short time during that interaction. The player needs to listen carefully to be able to identify the location of the phantom and stare at it to eliminate it.



```
bool isLookingAtMonster() {
    RaycastHit hit;
    if (Physics.Raycast(PlayerCam.transform.position, PlayerCam.transform.forward, out hit, 20f, MonsterLayer))
    {
        if (hit.collider.gameObject.tag == "Monster")
            return true;
        else
            return false;
    }
    else return false;
}
```

Figure 17. Partial script of raycast check. Source: Unity screen capture.

Besides, for determining whether the player is staring at the phantom, we use the function “Physics.Raycast” provided by Unity through specifying the starting point, direction, length, mask, etc. If it hits anything, the function will reference that object in the result. The partial script is listed in Figure 17.



```
// Update is called once per frame
void Update()
{
    SpawnTimeCount += Time.deltaTime;
    if(SpawnTimeCount >= FileSpawnTime)
    {
        Instantiate(Files[Random.Range(0, Files.Length)], FileCreationLocation.position, FileCreationLocation.rotation);
        SpawnTime = FileSpawnTime + Random.Range(-2.0f, 2.0f);
        SpawnTimeCount = 0;
    }
}
```

Figure 18. File generation check code. Source: Unity screen capture.

5.5. Work

The only work that the player needs to do other than surviving from the phantom's attack is classifying files. The file holder will generate a file for every 4.5 to 5.5 seconds as mentioned in section 4.2.3, and the player needs to put the file into the corresponding folder box based on its seal type. The grab and drop function is implemented by the use of the XR plug-in as mentioned in 5.2. For the file holder, we use the Update() function to check whether the time count to generate a file is larger than the spawn time required as shown in Figure 18.

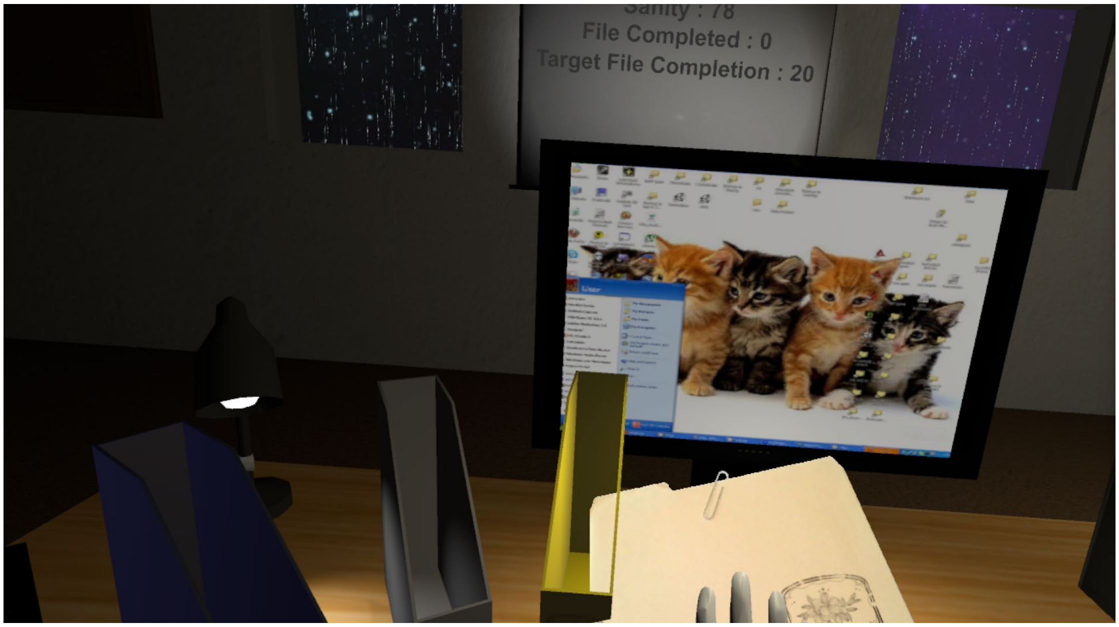
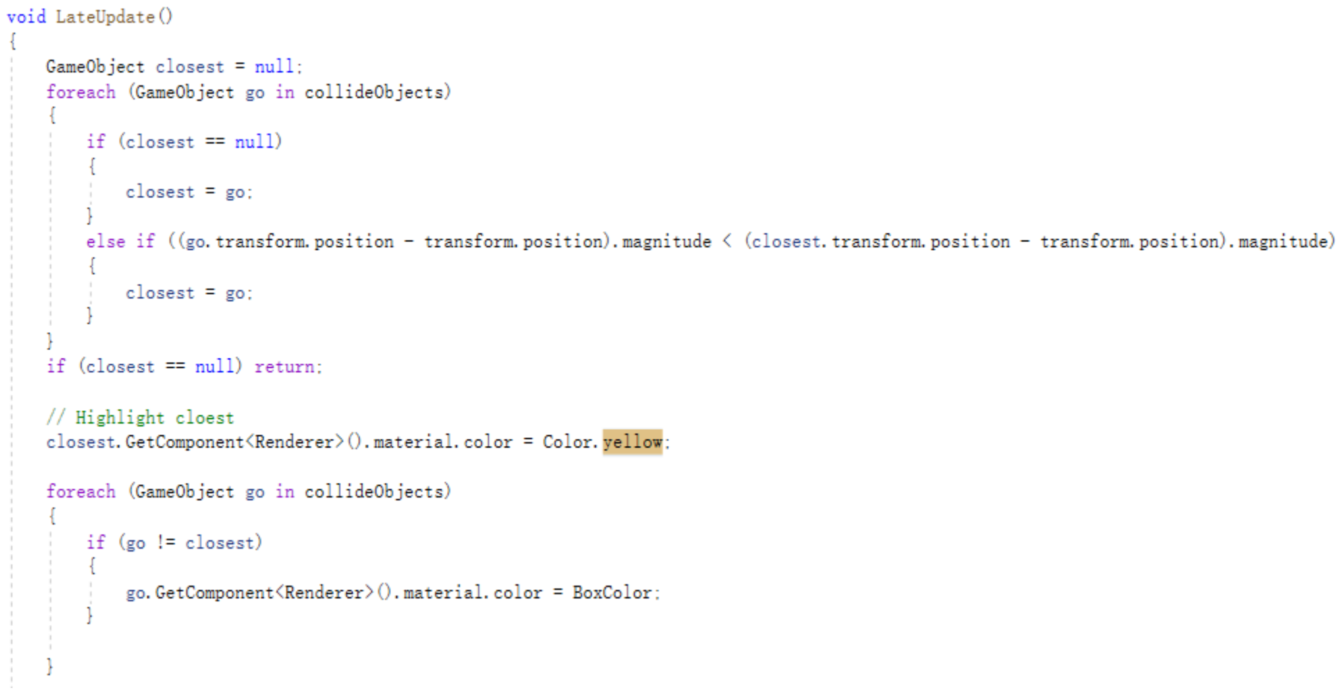


Figure 19. Highlighted folder box. Source: Screen capture.

Besides, the folder box would be highlighted when the player grabs and holds a file near it as shown in Figure 19. The highlighting is done by changing the material's color in the "Renderer" component to yellow, and reset to white when unhighlighted. However, we encountered a problem that multiple boxes might be highlighted at the same time because they are really close to each other. We find a solution by having the file object saving all the boxes it collides with. Then, at the end of the frame, determine which box is the closest one and highlight only that box as shown in Figure 20 below.



```
void LateUpdate()
{
    GameObject closest = null;
    foreach (GameObject go in collideObjects)
    {
        if (closest == null)
        {
            closest = go;
        }
        else if ((go.transform.position - transform.position).magnitude < (closest.transform.position - transform.position).magnitude)
        {
            closest = go;
        }
    }
    if (closest == null) return;

    // Highlight cloest
    closest.GetComponent<Renderer>().material.color = Color.yellow;

    foreach (GameObject go in collideObjects)
    {
        if (go != closest)
        {
            go.GetComponent<Renderer>().material.color = BoxColor;
        }

    }
```

Figure 20. Script of box highlight. Source: Unity screen capture.

The function that checks the correctness of file classification would only be triggered when the file is not held by the player. We detect this with the use of the "Select Entered" and "Select Exited" events in the event system provided by XR plug-in.

To add some random elements to the game, we created different prefabs with files with seals on different locations of the file. The file box that spawns files in it would choose randomly using the "Random.Range()" function.

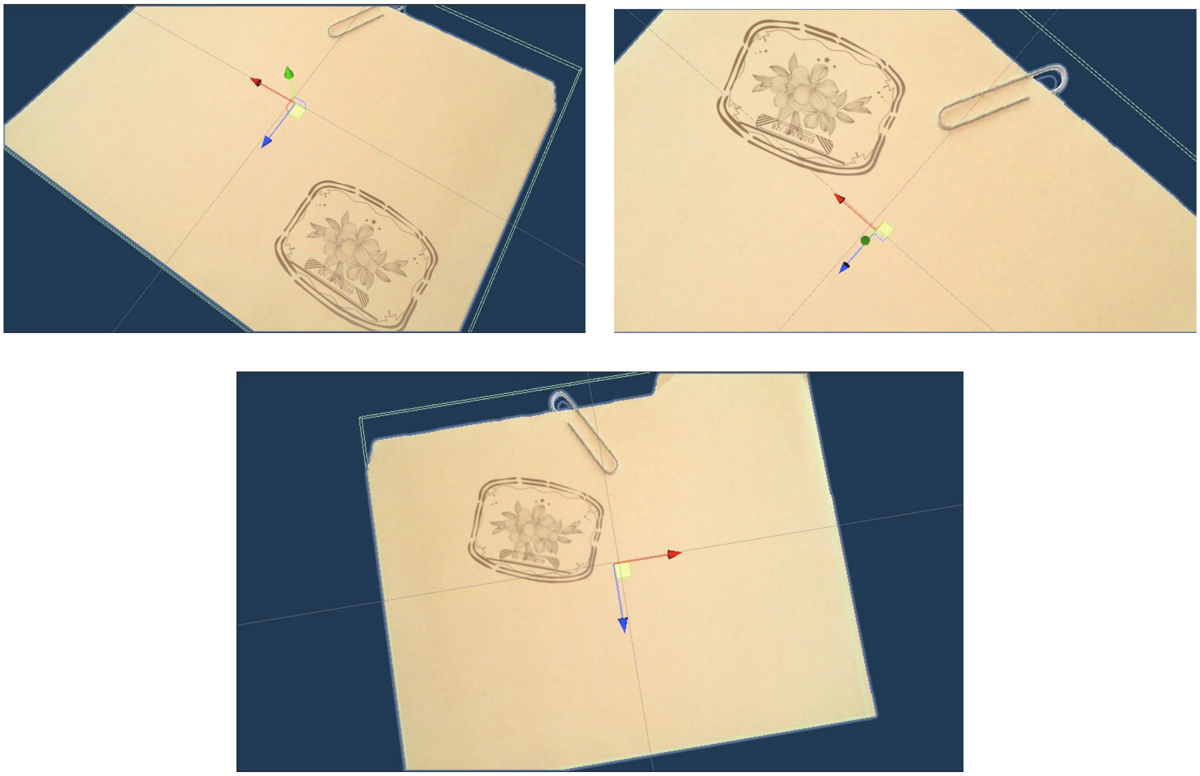


Figure 21. Three seal locations. Source: Screen capture.

An example of three files with different seal locations is shown in Figure 21.

5.6. User interface

The user interface (UI) components in VR games are slightly different from usual 3D games. In normal 3D games, we choose to have a health bar, or items displayed at a fixed location on screen. However, it’s unwise to have that in VR games because players tend to turn their head rather than just rolling their eyes to look at things. If we place the UI at the edge of the screen, players can hardly see it, and it may interfere with the gameplay if we place it at the center. Therefore, the best way to place UI in VR games is to put them inside the game with a fixed world position, and to use the canvas as the object for showing the UI.

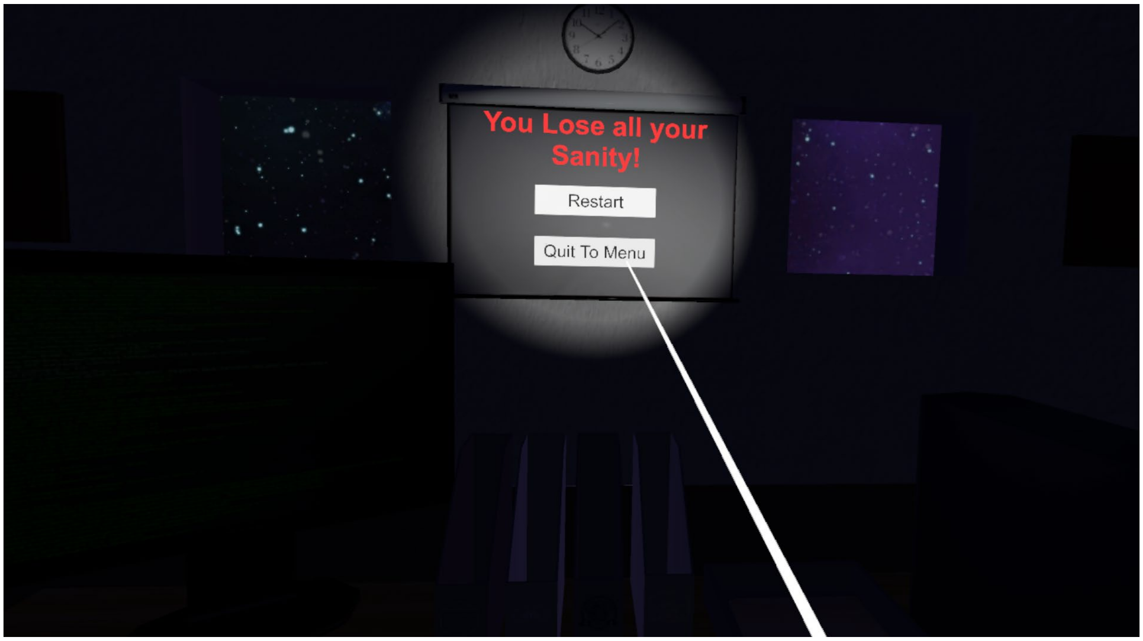


Figure 22. Raycasting from hand model. Source: Screen capture.

Additionally, we need the player to be able to "click" on the VR menu UI from a long distance. Again, we use the prefab in XR plug-in, which casts and draws a ray cast from the hand model in Figure 22. The player can aim that ray on a button and "click" on it by pulling the main trigger. The UI system is the same as normal UI in 3D games, except for a few things to set up.

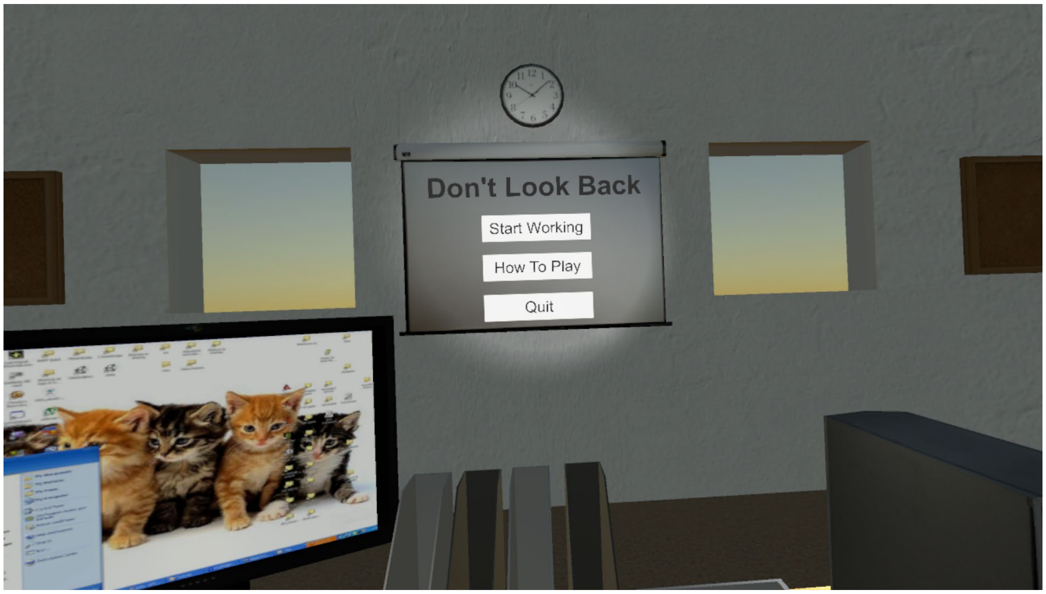
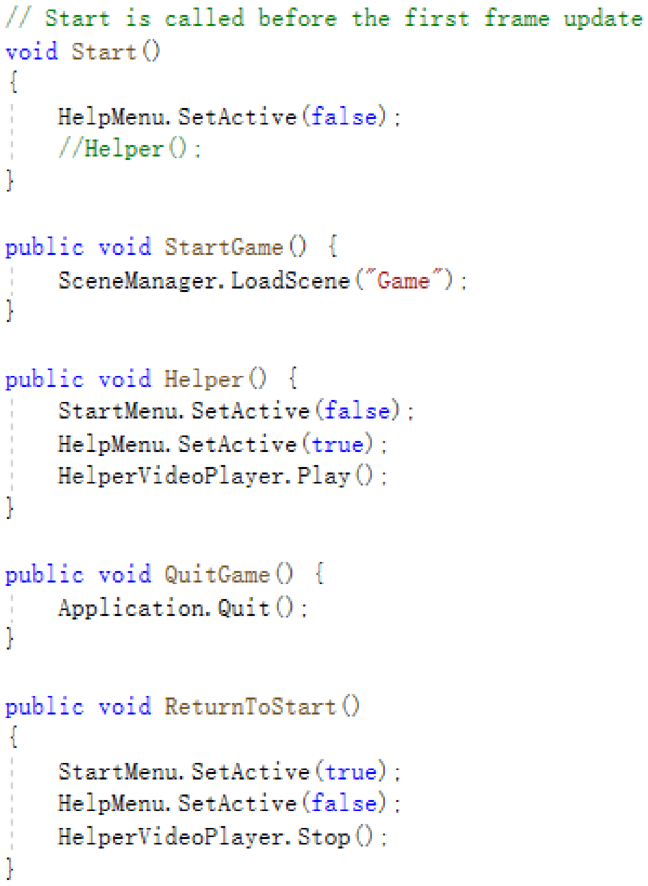


Figure 23. Start menu. Source: Screen capture.

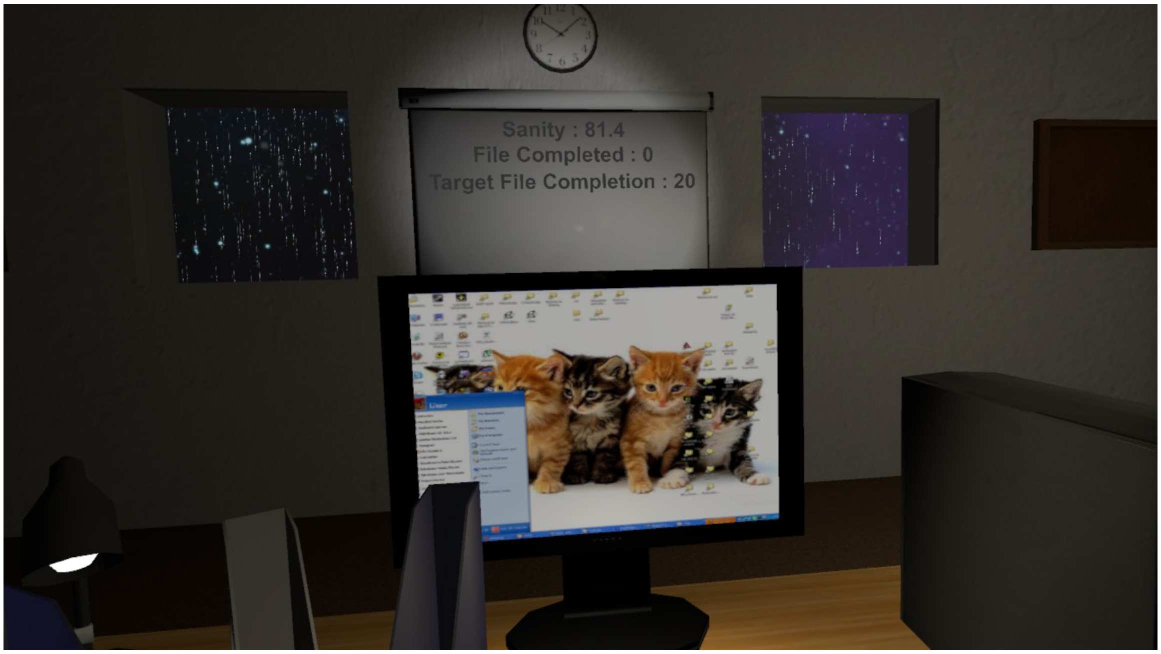
For our start menu, we have three buttons in total, which are “Start Working”, “How To Play”, and “Quit” in Figure 23. By clicking these buttons, players can either start the game, watch the tutorial video, or quit the game.



```
// Start is called before the first frame update
void Start()
{
    HelpMenu.SetActive(false);
    //Helper();
}

public void StartGame() {
    SceneManager.LoadScene("Game");
}

public void Helper() {
    StartMenu.SetActive(false);
    HelpMenu.SetActive(true);
    HelperVideoPlayer.Play();
}

public void QuitGame() {
    Application.Quit();
}

public void ReturnToStart()
{
    StartMenu.SetActive(true);
    HelpMenu.SetActive(false);
    HelperVideoPlayer.Stop();
}
```

Figure 24. Start menu script for buttons. Source: Unity screen capture.

All button functions are controlled by the StartUI.cs script bound with the canvas as shown in Figure 24.



Sanity : 81.4

File Completed : 0

Target File Completion : 20

Figure 25. In-game UI. Source: Screen capture.

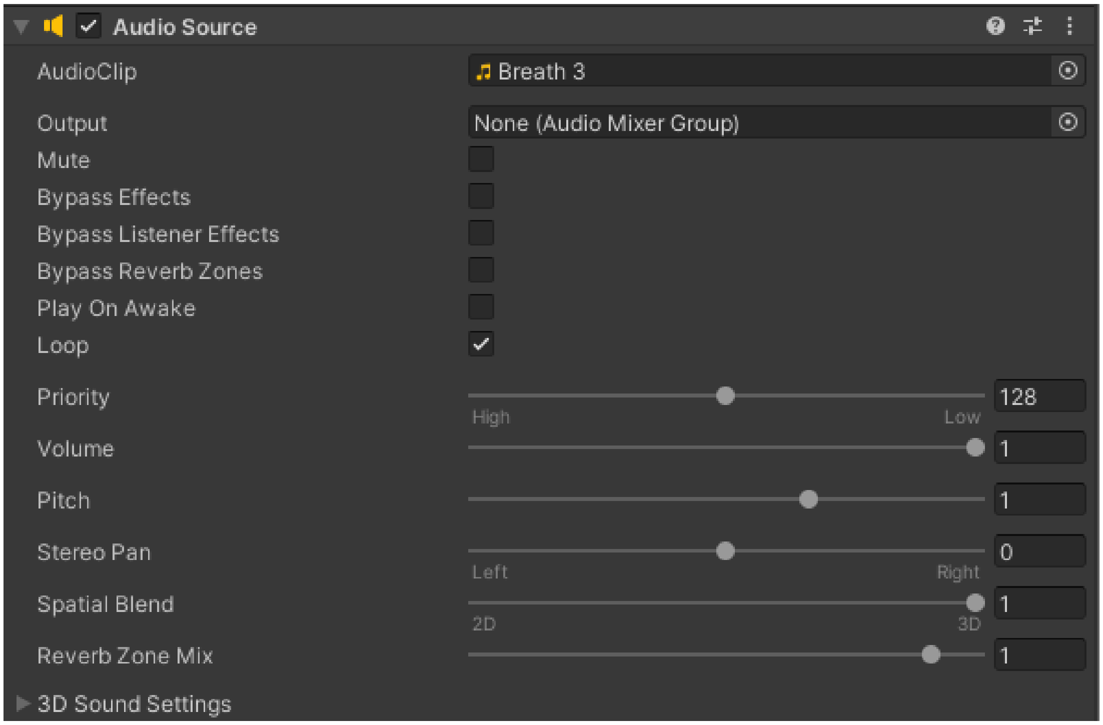
For the in-game UI shown in Figure 25, we list “Sanity” for the player’s current Sanity, “File Completed” for the number of assignments completed by the player, and “Target File Completion” for the number of assignments in total to win the game. Compared to the start menu UI, we control these shown values through PlayerLogic.cs in the Player Object directly. Whenever the sanity level of the player changes, the “Sanity” will also change in the Canvas.

5.7. Audio

5.7.1. Audio development

Audio, as part of the game, is important in creating an immersive experience. It becomes even more significant in our VR game, because it can be used to establish a sense of space through the stereo system. Thus, players can locate the phantom through listening and decide whether they need to continue working or to turn back. From all applicable methods provided by Unity, we chose the two most appropriate methods, which are audio source play and function PlayClipAtPoint.

The first method plays sounds through audio sources directly, and the developer can modify audio attributes in the inspector. In this method, the sound effects played will follow the movement of the object, and automatically change the volume and stereo pan. Therefore, we implement this method with the phantom’s breath sound, which needs to be played in a loop once the phantom appears.



Audio Source

AudioClip: Breath 3

Output: None (Audio Mixer Group)

Mute □

Bypass Effects □

Bypass Listener Effects □

Bypass Reverb Zones □

Play On Awake □

Loop ☑

Priority: 128

Volume: 1

Pitch: 1

Stereo Pan: 0

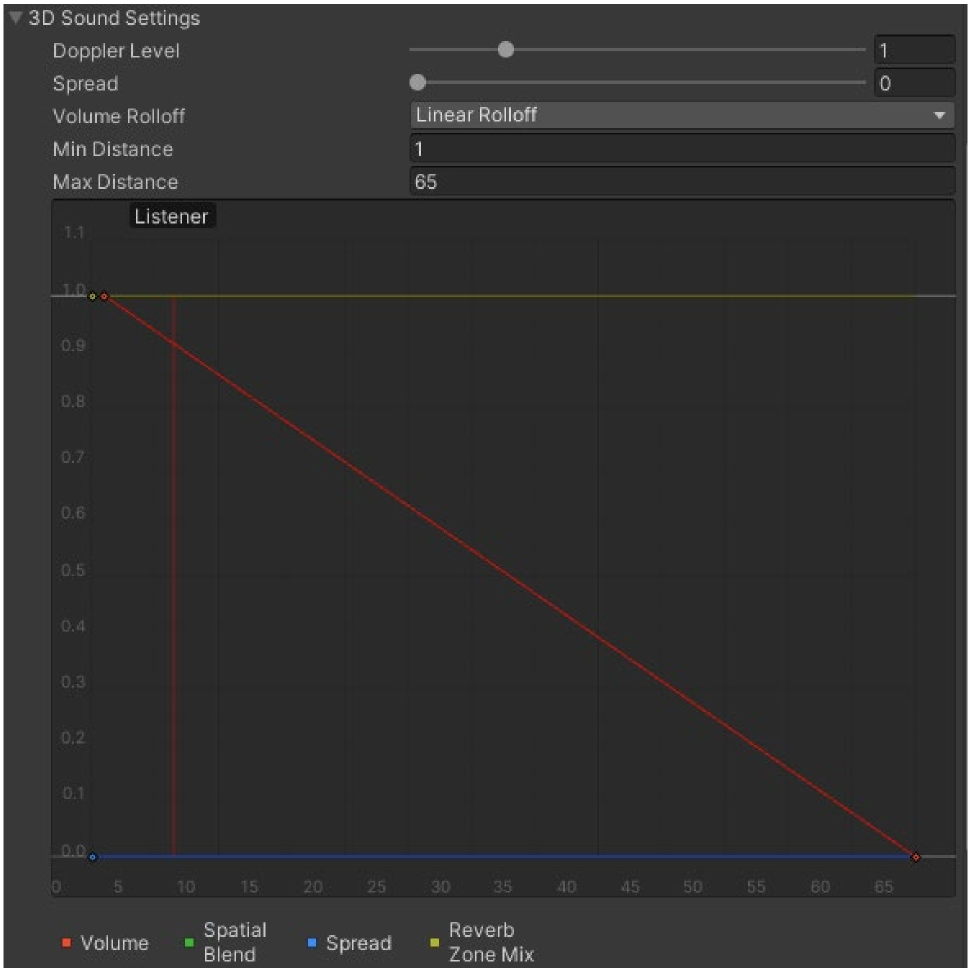
Spatial Blend: 1

Reverb Zone Mix: 1

▸ 3D Sound Settings

Figure 26. Audio source attributes. Source: Unity screen capture.

The first step of method one is to add a component named Audio Source into the phantom object, then edit the attribute of the Audio Source component in the inspector as shown in Figure 26. To create stereo, the Spatial Blend attribute and 3D Sound Settings need to be modified. For Spatial Blend, the value should be set to 3D, so we can adjust the detailed sound attributes in 3D Sound Settings.



3D Sound Settings

Doppler Level 1

Spread 0

Volume Rolloff Linear Rolloff

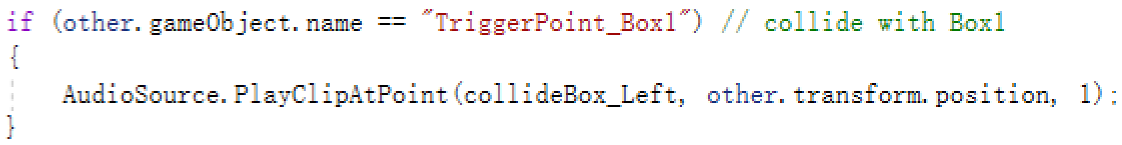
Min Distance 1

Max Distance 65

Figure 27. 3D sound settings. Source: Unity Screen capture.

In 3D Sound Settings, shown in Figure 27, the sound will have the largest volume at Min Distance, and the smallest volume at Max Distance. If the Max Distance value is set too high, the sound will take a longer time to attenuate from Min Distance until it reaches the highest Max Distance. The Volume Rolloff has three different modes, which are Logarithmic Rolloff, Linear Rolloff, and Custom Rolloff. In order to make the audio change much clearer, we utilize the Linear Rolloff with 65 distance, which changes the volume each time when the phantom moves.

The second method is to call the AudioSource.PlayClipAtPoint (AudioClip, Vector, Volume) function in the script each time when a specific situation is achieved. That will automatically change the volume and stereo pan of the sound based on the location of the player and that of the sound. It also helps avoid creating many Audio Source components to hold several Audio Clips as the first method. However, it can only play the sound once, which cannot be utilized if the sound needs to loop. Therefore, we use this method for playing sounds triggered by collision, because that type of sound only has to be played once during each collision.



```
if (other.gameObject.name == "TriggerPoint_Box1") // collide with Box1
{
    AudioSource.PlayClipAtPoint(collideBox_Left, other.transform.position, 1);
}
```

Figure 28. PlayClipAtPoint script. Source: Unity Screen capture.

A template of the function is shown in Figure 28. Once the collision is caught, the PlayClipAtPoint will be called with values of the audio clip name, the position of the phantom, and the max volume.

5.7.2. Solution to binaural effect

Even though Unity provides these methods, sometimes players still fail to successfully locate the phantom when sounds are triggered from directly behind them because of the binaural effect. According to the binaural effect (Strutt, 1876), humans locate sounds through the difference in arrival times and the sound levels heard by two ears. If the audio source is on the right side, the sound will arrive in the player's right ear first, then move to the left ear. By moving a longer distance in the air, the sound level of the audio wave attenuates, so the right ear receives a higher sound level than the left ear. However, the ears fail to locate precisely when the sound is from straight ahead or straight behind, because there's no difference in arrival time and sound levels. To solve this problem, we use Audacity again to edit the stereo pan of imported sounds to 3:7 if that comes from the right side, and to 7:3 if that comes from the left side. Accompanying this edition, the players are able to receive precise location information through sounds even though the phantom is from behind.

6. Evaluation

We participated in IMGD's AlphaFest event on 19 November 2021 to share our work in progress with the IMGD community and collect feedback from playtesting. For each playtest, we required volunteers to read and sign our IRB-approved Informed Consent Agreement (Appendix D) and COVID-19 Mitigation Protocol (Appendix E) which indicated that everyone participating in the playtest should wear a mask, and that we were sanitizing the devices used for testing for each session.

After each playtest session, we collected anonymous feedback from the player by having them fill out our IRB-approved Survey Instrument (Appendix F).

Due to our tight project schedule, we were only able to formally test our project at AlphaFest. We will conduct additional playtesting if possible.

6.1. AlphaFest

Our AlphaFest build of *Don't Look Back* had the game scene set up with complete working logic, allowing volunteers to play multiple rounds of the game if they wished. Since our game is a seated VR experience, only a single chair space was needed for a player. The only problem we did not expect is that one of the most essential mechanics of our game is based on sound effects. However, since there were multiple groups testing their works, the environment was too noisy for players to hear sound effects clearly. So the playtest result regarding the sound effects might not be ideal. If we have a chance to perform a second playtest in the future, we will choose a quiet environment or provide players with headphones.

6.2. Methodology

We built and distributed our IRB survey instrument using Qualtrics (Qualtrics, Provo, UT). The survey structure and questions and were based on standard questionnaire design methodologies.

We included thirteen questions in the survey instrument, ten of which were close-ended questions while the rest were open-ended questions. Close-ended questions are more efficient to use, and more reliable than other question types due to the uniform data they provide (Fink, p.54).

The ten close-ended questions included eight Likert-scale questions requiring a rating of 1-6 regarding effectiveness of elements used in game and overall experience. The two close-ended inquiries were multiple-choice questions asking players to identify the game genre and characterize their feelings while playing the game.

We also wanted to get an insight into what strategies players devised to survive the game, and gather more suggestions and/or opinions. Therefore, we posed the last three questions as open-ended (all optional), with one asking for strategies the player used and another soliciting general suggestions for our project. The final question was used to collect the email address of any IMGD volunteers who wanted to receive playtesting credit for their participation.

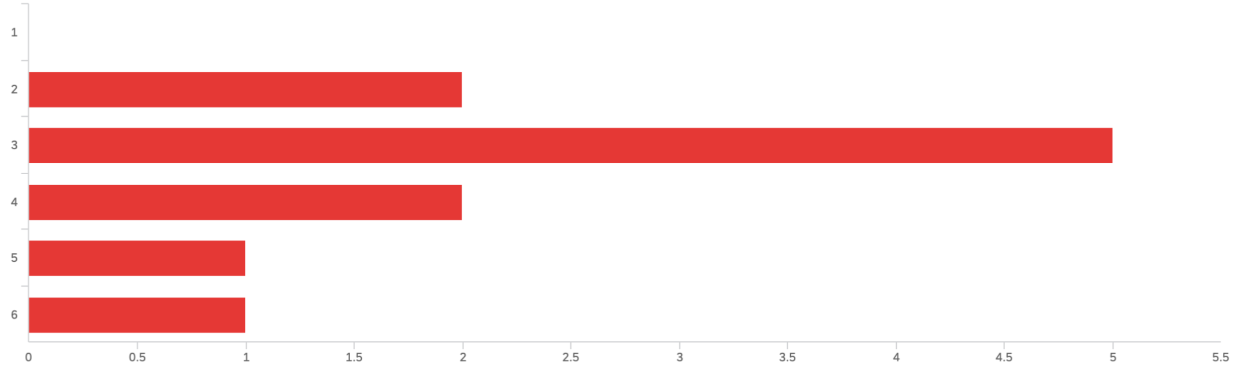


Figure 29. How would you rate the difficulty of understanding the game's rules and instructions? Lickert scale, 1=Difficult, 6=Easy. Source: Data screen capture.

6.3. Survey results

We have collected eleven responses in total from the AlphaFest. The full test report can be found in Appendix F. As seen in Figure 6.1, most players might not think it's relatively easy to understand the game rules. The way we explain the rules is by showing them a tutorial video inside the game. We are considering improving this by designing a tutorial level and having players actually play through one level instead of just watching a video. In this way, they might be able to understand game rules better. However, due to the intense schedule of our development, we might not be able to hold a second round of playtest and assess the performance of our tutorial level.

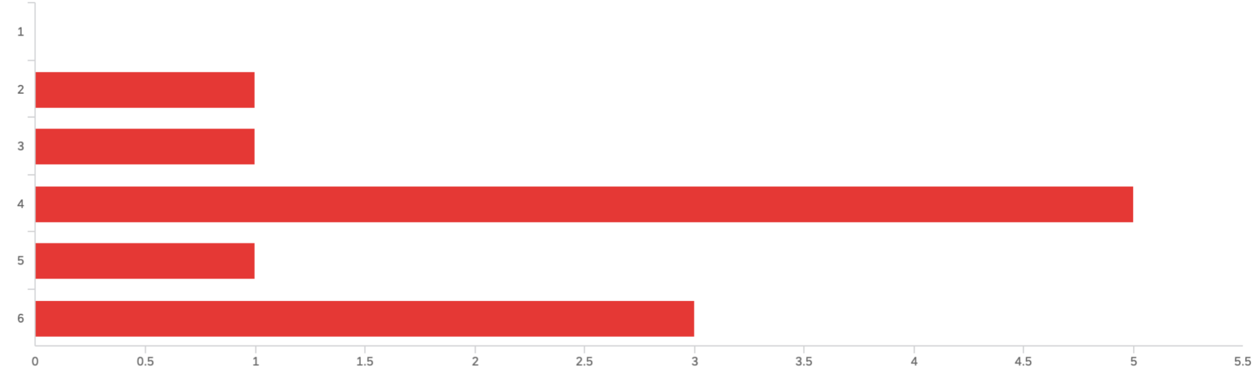


Figure 30. How would you rate the difficulty of operating the game? Lickert scale, 1=Difficult, 6=Easy. Source: Data screen capture.

Figure 6.2 shows results for how players think of the difficulty of game operation. As we can see from the graph, the majority of players think the game is easy to operate, thus we will not plan to make any large adjustments on game control designs.

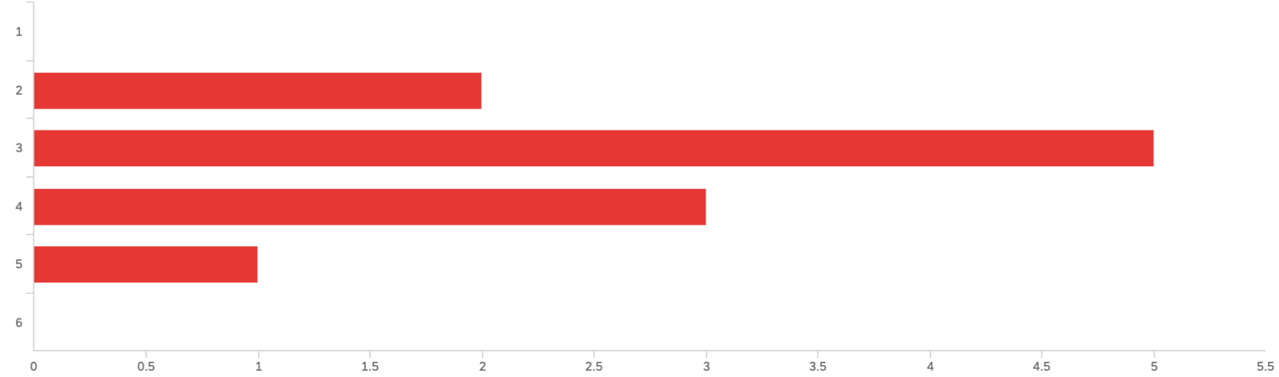


Figure 31. How would you rate the total duration of the game experience? Lickert scale, 1=Too Short, 6=Too Long. Source: Data screen capture.

Figure 6.3 shows the response to how players would think of the overall game length. No one thinks the game process is too short nor too long, while most of them chose a rate of 3 or 4, which indicates our game length is just right.

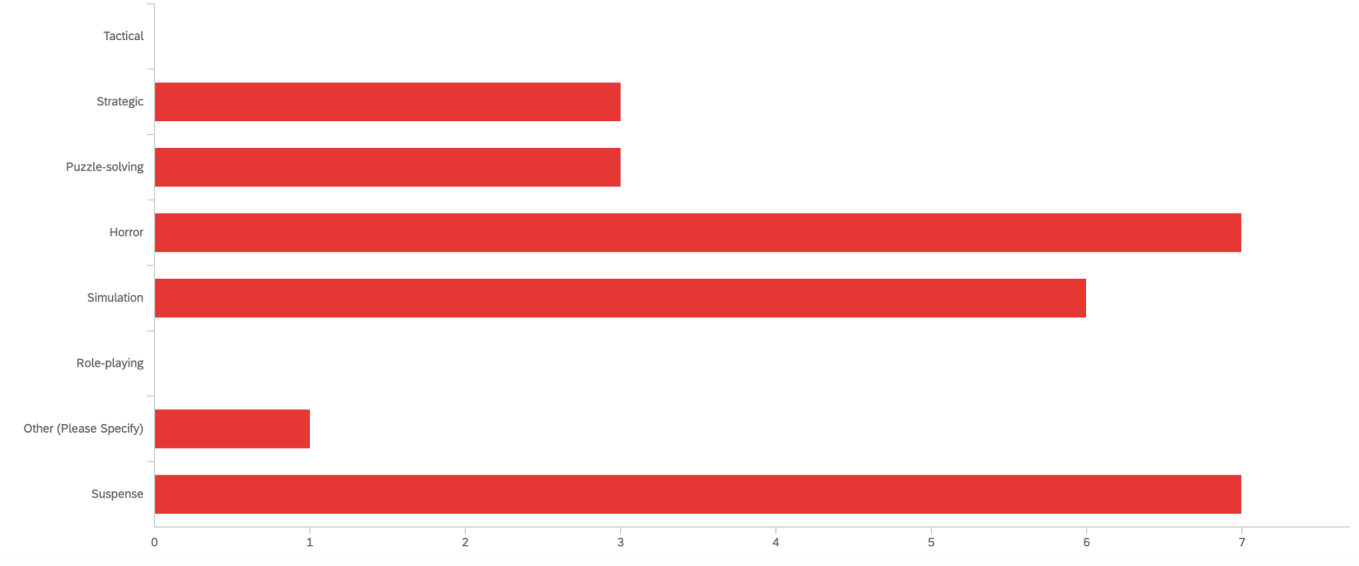
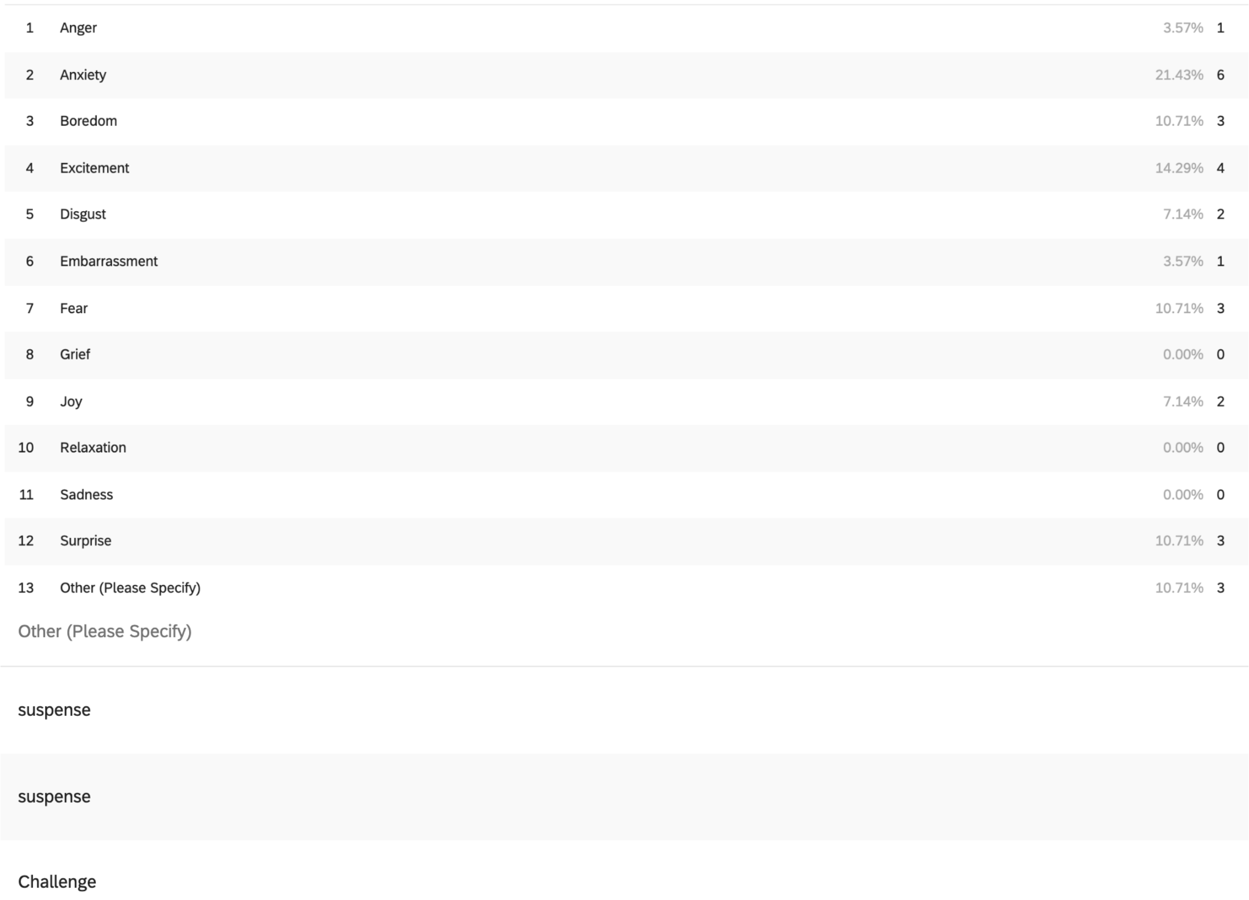


Figure 32. Which of the following terms would you use to describe the game? Please check all that apply. Multiple checkboxes. Source: Data screen capture.

The next question asks players to identify the game genre. We self-identify the game as horror and suspenseful. According to the survey result in Figure 6.4, we would say most of the players would agree with us. However, it's also possible that the result is influenced by the fact that, before players tried our game, we gave them a brief introduction to our game. Some of them might be aware in advance that the game they were experiencing is a horror game. What we didn't expect is that there are six responses that think our game genre is simulation. It's still understandable since we designed a classification task for the player to work on and classifying files is the only interactive activity using the controller.



1	Anger	3.57%	1
2	Anxiety	21.43%	6
3	Boredom	10.71%	3
4	Excitement	14.29%	4
5	Disgust	7.14%	2
6	Embarrassment	3.57%	1
7	Fear	10.71%	3
8	Grief	0.00%	0
9	Joy	7.14%	2
10	Relaxation	0.00%	0
11	Sadness	0.00%	0
12	Surprise	10.71%	3
13	Other (Please Specify)	10.71%	3

Other (Please Specify)

suspense

suspense

Challenge

Figure 33. Which of the following emotions did you experience while playing the game? Please check all that apply. Multiple checkboxes. Source: Data screen capture.

As seen in Figure 6.5, we asked players to identify emotions they experienced during the game. We wanted to find out what the player was feeling like when playing our game. In this way we can have a better understanding of whether we have successfully created a horror atmosphere through visual and acoustic effects. Based on the results below, most respondents said they experienced anxiety during the game, while only three responses were "fear" and two respondents specified "suspense" as the emotion they felt. Followed by anxiety and excitement, the third most chosen emotion is boredom. In our early brainstorming stage, we intended to design a boring but simple work for players to complete as we didn't want the work to take away too much attention from players. The core mechanic is to identify the phantom's location by sound effects and players are not supposed to focus too much on the work. In conclusion, it's still expected that some players would feel bored dealing with the file classification. However, we didn't get as many responses saying they felt scared during the game play as we initially expected. It's possible that, since we tested our game in a public space while many people were passing by, the noisy background reduced the effectiveness of sound effects. In the future playtest, we will choose a private and quiet place to exclude potential environmental factors.

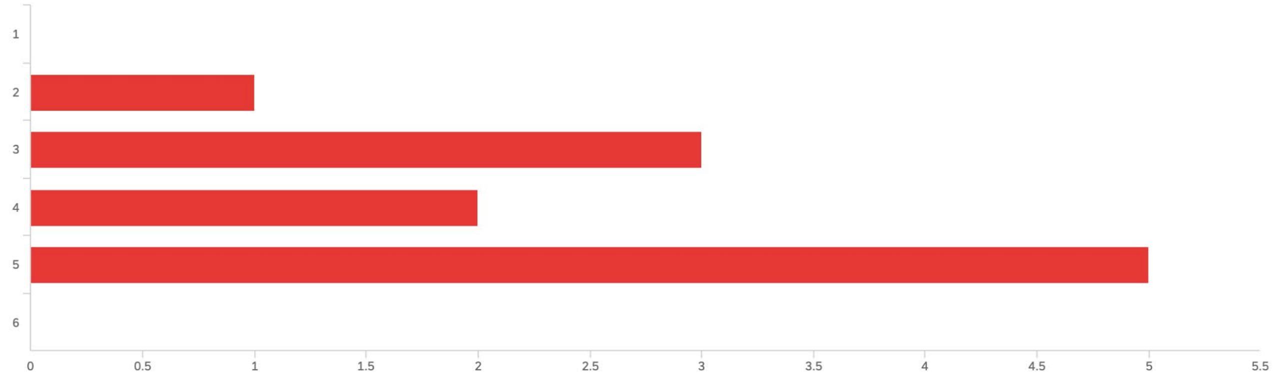


Figure 34. How effectively did the game's mechanics invoke emotions you selected above? Lickert scale, 1=Ineffective, 6 = Highly Effective. Source: Data screen capture.

The next question is a follow-up question for the previous one. We were assessing the effectiveness of game mechanics in invoking the emotions the player felt above. As seen in Figure 6.6, the majority of the players would agree that our game mechanics can invoke the emotions they felt, which means the design of file classification work and looking back to find phantom does help effectively make them feel anxious and excited.

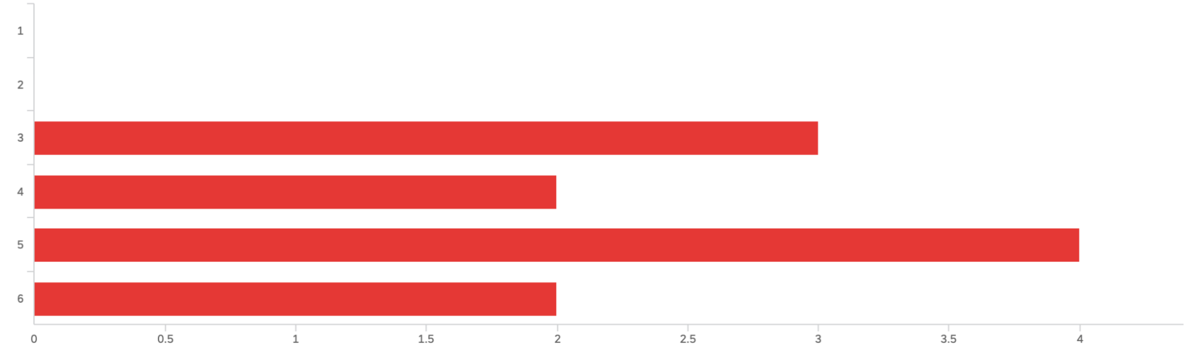


Figure 35. How would you rate the effectiveness of the game's visuals (modeling, layout, color, lighting, etc.) at creating an immersive experience? Lickert scale, 1=Ineffective, 6 = Highly Effective. Source: Data screen capture.

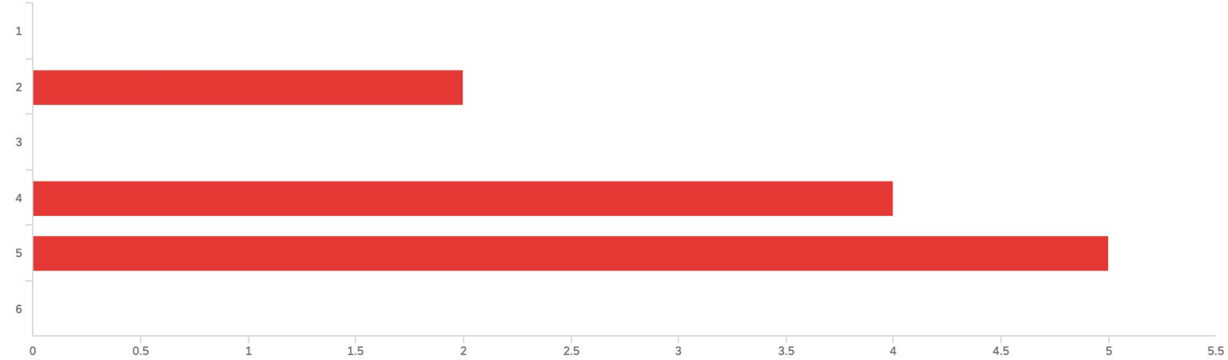


Figure 36. How would you rate the effectiveness of the game's use of sound effects and music at creating an immersive experience? Lickert scale, 1=Ineffective, 6 = Highly Effective. Source: Data screen capture.

The next two questions asked players how they think of the effectiveness of visual and sound effects in creating immersive feelings. Figure 6.7 and 6.8 show responses we have collected for those two questions. Most players would agree that the sound and visual effects can help create an immersive experience. Thus we will keep our current design and not make large changes in future. It's worth noticing that 2 respondents rated 2 for effectiveness of sound effects. This might be due to environmental noises on AlphaFest.

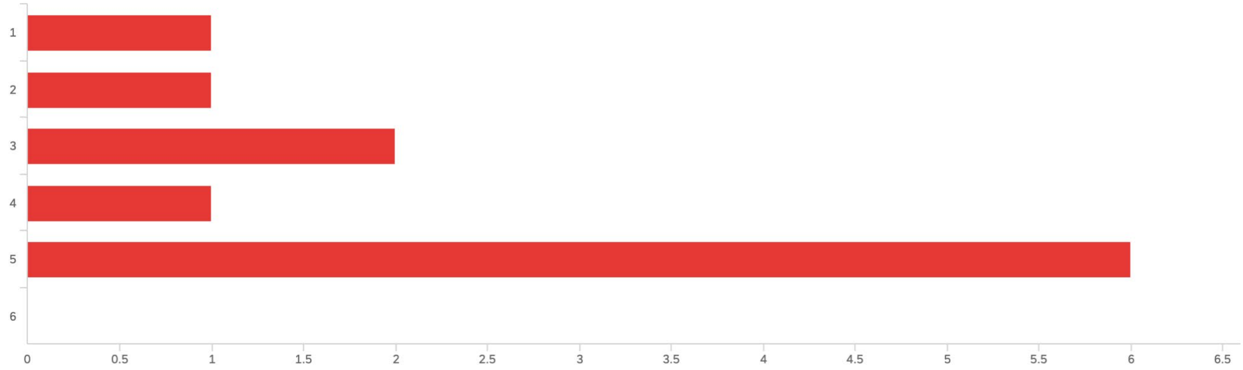


Figure 37. How likely would you be to replay the game to explore different outcomes? Lickert scale, 1 = Not Likely, 6 = Very Likely. Source: Data screen capture.

Next question asks how likely they will replay the game to explore different outcomes. According to responses shown in Figure 6.9, six out of eleven responses rated 5 while only two people rated for 2 or below. At the current stage, we only hard-coded several moving routes for phantoms, and there's few environmental distractions in the first level. In our initial design, we are planning to diversify the game by making phantoms move randomly and adding more environmental distractions such as thunder or ringing. Even if in the current stage our game does not have as many random elements as we planned, we are satisfied with players' responses.

As seen in Figure 6.10, we also collected responses for how likely players would be to recommend our game to friends. As seen from the graph below, over half of the respondents are more likely to recommend the game to others. Our game is still in the early development stage and we haven't implemented all the designs, but we are satisfied with what we have achieved and how players rated our game.

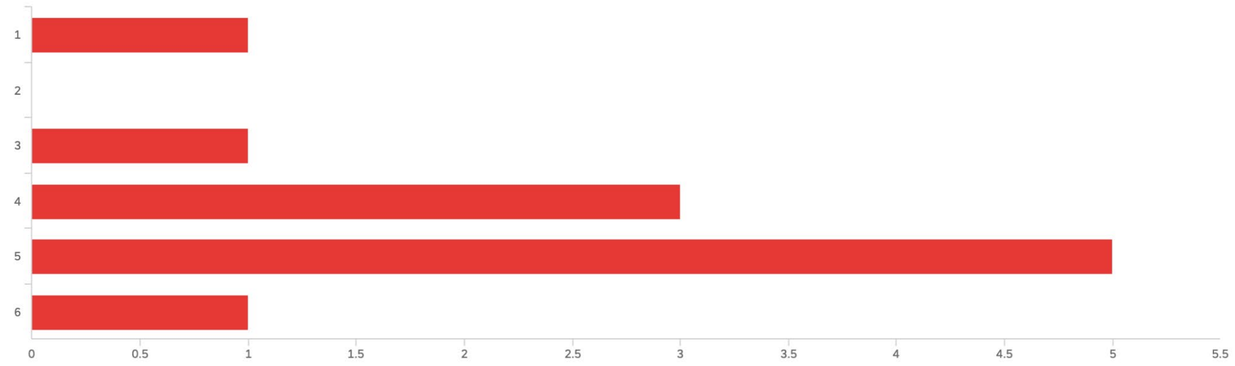


Figure 38. How likely would you be to recommend this game to a friend? Lickert scale, 1 = Not Likely, 6 = Very Likely. Source: Data screen capture.

We left the next question open-ended for players to explain strategies or any satisfying ways to survive the level. We were trying to learn what strategies players have figured out to ace the game mechanics. We have extracted some key patterns that were commonly mentioned in their responses. Most of the responses mentioned speed-clearing files. The game rule is to sort out a certain number of files in order to complete the game. The more time you spend on it, the more sanity you will lose. Thus it's indeed a reasonable strategy to speed clean all the files. We also have several responses that contradict each other. Some players mentioned it's better to look at the back as little as possible while others suggested looking back frequently to kill phantoms. Since we only collected a few responses for this question and we haven't implemented all the functions and designs, this might be the reason why players have completely different winning strategies.

The next question asks players for general suggestions. Two of them mentioned improvement on file classification work. One player suggests adding additional ways such as vibration or color to get to know the file type. When designing the seal, each seal has its unique color. All colors might appear the same under the dim light inside the game. We will fix this by increasing the saturation level of colors. Additionally, we love the idea of adding vibration to our game but we are not considering adding vibration as a way to help players identify file type. We have two other respondents suggesting us adding more ways of interaction with objects. For example, objects can be thrown to phantoms and objects on the desk such as the computer. It's definitely a great idea to make the computer into an interactive item and it can help create distractions as well. Unfortunately the computer in our game is just a model which has no interactive function. We might want to develop an interactive interface for it in the future.

7. Conclusion

7.1. Experience goal achievement

The goal of the *Don't Look Back* team was to design the mechanics for a suspenseful horror VR game and to implement one level with Unity and XR plug-in. We started this project from designing two prototypes with different core mechanics, testing them to select the most appropriate one and elaborating it to a detailed Game Design Document. We fully implemented one level based on the Game Design Document and assessed our development by playtesting it on the AlphaFest. Additionally, we found all needed assets and developed one level through the Unity with XR plug-in. We gathered information from players by attending AlphaFest, and made future plans based on these feedbacks.

7.2. Challenges

As our project is running on an external device, we have encountered problems in both technical and logistic aspects. In terms of logistic issues, we decided to use the latest Oculus Quest 2 headset, but the IMGD department was not able to provide us with the required headset and controllers. We were in a situation that the controller they gave us was not compatible with our game, so two team members bought two VR devices for development and playtesting in the AlphaFest. Additionally, we thought it would be good to have one computer in the Global Lab configured to run our game so that we could hold in-person meetings in the Global Lab. However, it turned out that scheduling the Global Lab was very inconvenient and sometimes administrators might miss our applications. Also, we need the administrator to be physically present in the lab every time we need to use the computer since we don’t have permission to install software or configure devices on the computer. We found that meeting in-person and testing out our VR project was not as simple as we expected, but eventually we managed to solve this problem by having one team member bring a gaming laptop which is powerful enough to run our games on it.

In terms of technical problems, the Oculus Quest 2 can connect to the computer as long as they are running under the same wireless network. However, it was very hard to get our VR headset connected to WPI’s wireless network, as it needs to install the certification and we have no idea how to do that. Instead of spending too much time on configuring the network, we decided to connect the computer with an AirLink cable. The ATC didn’t allow us to keep the cable for more than a week, so our team decided to get a third-party cable as it’s less expensive than the original one. However, in later testing, we found that the performance of the third-party cable was not steady, as it sometimes would lose connection with our headsets. It was not a big problem, and we didn’t replace it due to the limited budget.

In conclusion, different from the development process for a normal PC game, we were facing different problems that we didn’t expect at the beginning. We will also provide recommendations in the later section for future MQP groups who are planning to work on VR game development.

7.3. Looking forward

Although our *Don’t Look Back* team has successfully achieved the goal of designing a VR suspense and horror game and developing the medium level of it, some parts of our project can be enhanced. Therefore, we created a list of future works to accomplish and added some helpful recommendations to students who intend to work on future VR game projects.

7.3.1. Future work

For future plans of *Don’t Look Back* development, we prefer to implement all levels into our game first. Since we have already found all demanded art and audio sources, we only need to modify game attributes like the respawning time of the phantom or the decreasing speed of the sanity bar to adjust difficulty for each game level.

Furthermore, we may change the sanity bar performance. In the present, the sanity bar is shown in the canvas UI and it’s countable, somehow negatively affecting the immersion of players. Therefore, we plan to replace it with other items with special meanings like the cross. The sanity bar in this situation will be performed with the brightness of the light source within the item. If the sanity of players increases, the item gets brighter, and the item gets darker if the sanity decreases. By doing so, the players have no countable source to manage their sanity and our games may become more immersive.

7.3.2. Recommendations

Based on our development experience, we will provide some suggestions and recommendations for future VR MQPs. First, please check with ATC and IMGD department for devices that are available to use. Be sure to confirm if the borrowed VR device comes with the usable controller as well as cables. Furthermore, we suggest future teams carefully select which types of VR game they want to make--walking or sitting. Walking VR games are undoubtedly more interesting, but it’s also hard to develop and developers have to consider the safety of players. Sitting VR games restrict players in one area, but it’s safer and easier to achieve.

Works cited

[1] Audacity® is free software distributed under the terms of the GNU General Public License. URL.

[2] Brodkin, Jon (June 3, 2013). "How Unity3D Became a Game-Development Beast." Dice Insights. Retrieved October 30, 2018. URL.

[3] Cohen, Peter (October 11, 2007). "Unity 2.0 game engine now available." PCWorld. Retrieved January 14, 2019. URL.

[4] Fine, Richard (August 11, 2017). "UnityScript's long ride off into the sunset." Unity Technologies Blog. Retrieved September 18, 2017. URL.

[5] Fink, A. (2017). *How to conduct surveys: A step-by-step guide*. SAGE.

[6] Girard, Dave (September 27, 2010). "Unity 3 brings very expensive dev tools at a very low price." Ars Technica. Retrieved January 14, 2019. URL.

[7] *Job Simulator* [Video game]. (2016). Owlchemy Labs.

[8] Kumparak, Greg (March 18, 2014). "Unity 5 Announced With Better Lighting, Better Audio, And ‘Early’ Support For Plugin-Free Browser Games." Retrieved January 14, 2019. URL.

[9] MannSusan, S. B. (2019, April 5). *Red Light/Green Light - Game rules and variations*. IcebreakerIdeas. Retrieved December 7, 2021. URL.

[10] McWhertor, Michael (October 22, 2014). "Former EA CEO John Riccitiello is now head of Unity." Polygon. Archived from the original on December 5, 2018. Retrieved October 17, 2018. URL.

[11] Richter, J. (2021, October 8). *How darkest dungeon and other horror games make great use of Sanity Mechanics*. Game Rant. Retrieved December 7, 2021. URL.

[12] Sprigg, Sam (June 8, 2018). "Unity MARS Augmented and Mixed Reality authoring studio now available." auganix.org. URL.

[13] Strutt, John William. “Our Perception of the Direction of a Source of Sound.” Scientific Papers, pp. 277–279. URL.

[14] *SUPERHOT* [Video game]. (2016). SUPERHOT Team.

[15] The data analysis for this report was generated using Qualtrics XM software, Version November 2021. URL.

[16] "Unity - Manual: XR." docs.unity3d.com. Retrieved April 15, 2021. URL.

[17] "Unity – Multiplatform." Unity. Unity Technologies. Retrieved June 27, 2018. URL.

Appendix A. Design document

- **Overview**
  - *Don't Look Back* is a survival horror game for VR. The player takes the role of an archivist who must complete a task of file classification while being menaced by a phantom sneaking up behind them.
- **Mechanics**
  - **Core mechanics:**
    - The player will be given a file classification task to work on. At the same time, there will be phantoms approaching the player from behind. The terrifying presence of the phantom causes the player's sanity to decrease over time. A small amount of sanity can be recovered every time the player correctly finishes their work. If the phantom reaches the player, their sanity will sharply decrease. If the player looks back and stares at the phantom for a few seconds, it disappears. However, the player isn't always able to see where the phantom is. When the phantom is approaching, it will leave some traces (examples: footsteps, blinking lights) to give players a hint of its current location. At the same time, there will be environmental noises or visual distractions to interfere with the player's work. The player needs to survive the day by completing all the work while not losing all of their sanity.
  - **Goals:** Complete the work before the sanity bar becomes empty.
  - **Controls:** Use handler to interact with items and finish work, turn headset to look back in-game.
- **Gameplay**
  - The player needs to survive 3 days to complete the game. The difficulty will gradually get harder.
  - Day 1:
    - The difficulty of this level is EASY

        - Once players look back, they should be able to see the phantom itself when it approaches them.
        - There won't be any environmental distractions for players.
        - The sanity bar is visible.
        - The phantom will appear at a low frequency.
    - Day 2:
        - The difficulty of this level is MEDIUM.
        - The phantom won't always be visible, but other signs of it can still be seen in the environment.
        - There will be some sound or visual distractions that can intervene with players' judgments or their working progress.
        - The sanity bar is visible.
        - The phantom will appear slightly more often.
    - Day 3:
        - The difficulty of this level is HARD.
        - The phantom is mostly invisible, so the player can only see indirect signs of its location.
        - Environmental distractions will appear frequently.
        - The sanity bar isn't visible, so the player must track it themselves.
        - The phantom will appear much more often.
- **Game Elements**
    - **Sanity Bar: An initial value of 100%**
        - The sanity bar will *decrease* at a steady rate of *1% per second* with the progress of the time.
        - The sanity bar will immediately *lose 30%* if the phantom reaches the player.
        - Each time the player correctly sorts a file, the sanity bar will *increase by 10%.*
        - Incorrectly sorting a file causes a loss of 5% sanity.

  - Each time the player looks back (stare at the phantom or not), the sanity bar will *decrease at a steady rate of 5%* per second.

- **Phantom**
  - **Movement:**
    - Phantoms will approach players from behind, either slightly left or right. They won’t come from directly behind.
    - When the phantom is approaching, players should be able to hear its footsteps, breathing, and other noises it makes when colliding with objects in the room. This may vary depending on difficulty level.
    - Doesn’t always approach the player, and sometimes even goes backward
    - If players look back and stare at the phantom for a few seconds, it vanishes. There will be a “grace period” of a few seconds before it reappears.
  - **Buff & Debuff (future plan):** Looking back frequently causes it to cost more sanity.
  - **Interaction with objects in the room**
    - Causing the lamp to blink when passing by.
    - Breaking cups/ fragile objects.
- **Work**
  - **File classification:**
    - Players are given a pile of file folders to classify based on a seal on each folder.
    - Each time the player should grab a folder and drop it into a box of its own categorization.
    - Each of the 4 types of seals can appear anywhere in the folder.
    - The player picks up a folder, finds the seal in it, and puts the folder in the correct place.

        - **Sanity recovery (future plan):**
            - Successively sorting a file into the correct box earns a small bonus on the sanity recovery speed until it achieves the max recovery speed.
            - If a file is put into the wrong box, the sanity recovery speed resumes its original rate.
    - **Distractions (only available for Day 2 and 3)**
        - Visual distraction:
            - A lamp near the player can occasionally turn off, making it harder for the player to see what they're doing until they turn it back on.
        - Sound distractions:
            - Heavy rain and thunder outside
            - Phone ringing

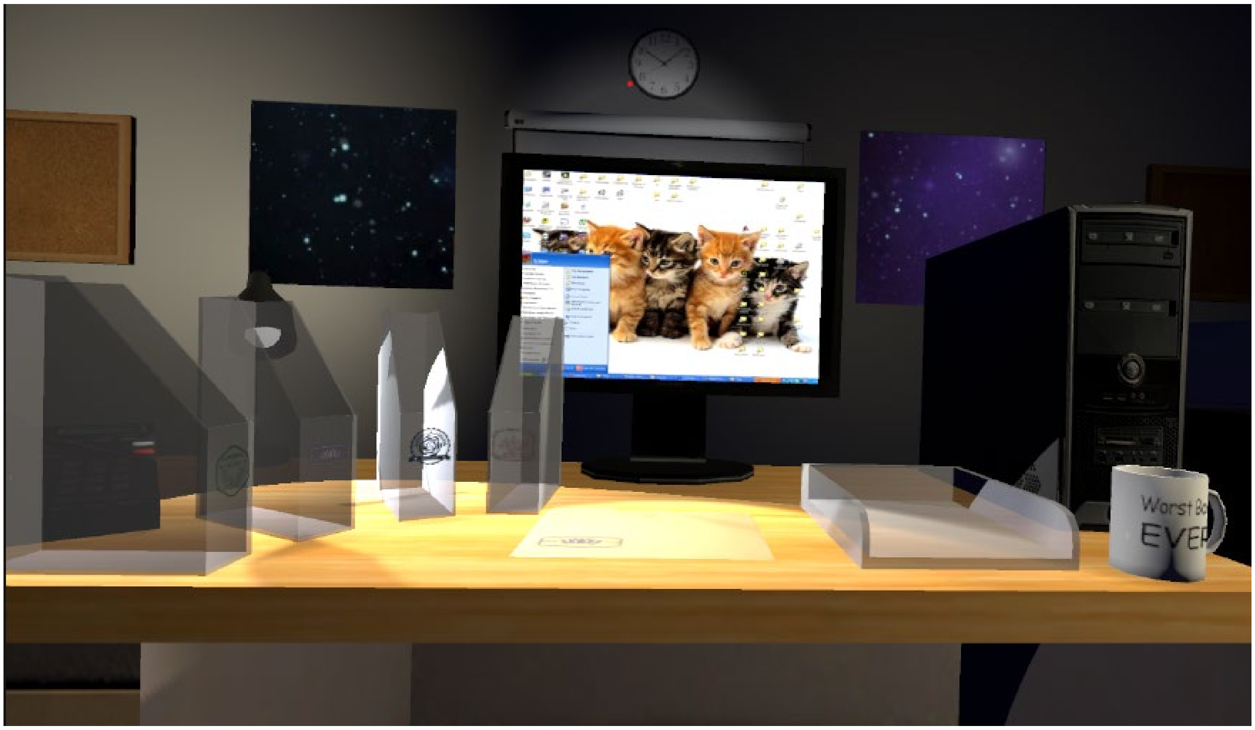
Appendix B. IRB Purpose of Study/Protocol

Title of Research Study: Suspenseful VR

Purpose of study

Obtain user feedback to determine if experience goals are being achieved, locate operational bugs, and identify opportunities for design improvement.

Project description



Preliminary screenshot of the player's virtual workstation.

This project immerses players in a virtual reality simulation of an office environment at night, tasked with simple, repetitive administrative busywork. As the game progresses, eerie sounds become increasingly audible in the space behind the player, who must resist the urge to look backward or risk witnessing the approach of an uncanny supernatural entity, the mere sight of which may threaten their virtual sanity.

Study protocol

Participants are greeted by researchers, presented with an opening briefing (see below), and asked to read and sign the Informed Consent Agreement and COVID Risk Mitigation Protocol forms (attached). PCs equipped with the required VR headset, controllers and ready-to-play software are provided. Researchers will assist participants in the simple procedures for wearing and adjusting the headset, and operating the associated hand controllers.

Instrumentation in the software anonymously records player activity (actions taken, intervals between actions, total playing time, etc.) during the test.

After completing the games, participants are asked to complete a short anonymous survey (sample attached) to characterize aspects of their subjective experience and solicit suggestions for improving the experience.

Opening briefing for testers

“Hello, and thank you for volunteering to test our project. Before we begin, could you please read and sign these Informed Consent Agreement and COVID Risk Mitigation Protocol forms? [Tester signs forms.] Thank you. When your session is complete, we will ask you to complete a brief survey about your play experience. At no point during your test session, or in the survey after, will any sort of personal and/or identifying information about you be recorded. Please begin playing when you feel ready.”

Appendix C. IRB Informed Consent Agreement

Informed Consent Agreement for Participation in a Research Study

Investigator: Brian Moriarty

Contact Information: bmoriarty@wpi.edu, 508 831 5638

Title of Research Study: Suspenseful VR

Sponsor: WPI

Introduction: You are being asked to participate in a research study. Before you agree, however, you must be fully informed about the purpose of the study, the procedures to be followed, and any benefits, risks or discomfort that you may experience as a result of your participation. This form presents information about the study so that you may make a fully informed decision regarding your participation.

Purpose of the study: This study will obtain user feedback to facilitate design improvements and find/address operational bugs.

Procedures to be followed: You will be asked to play a brief game lasting less than ten minutes Instrumentation in the game software will anonymously record your activity during play. After completing the game, you will be asked to complete a brief, anonymous survey describing aspects of your subjective experience.

Risks to study participants: Some players may experience mild disorientation, dizziness or nausea while immersed in a virtual reality experience. **If you begin to experience any physical discomfort during the test session, please ask a test administrator to remove your VR headset immediately.** There are no other foreseeable risks associated with this research study.

Benefits to research participants and others: You will have an opportunity to enjoy and comment on a new game under active development. Your feedback will help improve the game experience for future players.

Record keeping and confidentiality: Records of your participation in this study will be held confidential so far as permitted by law. However, the study investigators and, under certain circumstances, the Worcester Polytechnic Institute Institutional Review Board (WPI IRB) will be able to inspect and have access to confidential data that identify you by name. Any publication or presentation of the data will not identify you.

Compensation or treatment in the event of injury: There is no foreseeable risk of injury associated with this research study. Nevertheless, you do not give up any of your legal rights by signing this statement.

For more information about this research or about the rights of research participants, or in case of research-related injury, contact the Investigator listed at the top of this form. You may also contact the IRB Manager (Ruth McKeogh, phone 508 831-6699, email irb@wpi.edu) and/or the Human Protection Administrator (Gabriel Johnson, phone 508-831-4989, email gjohnson@wpi.edu).

Your participation in this research is voluntary. Your refusal to participate will not result in any penalty to you or any loss of benefits to which you may otherwise be entitled. You may decide to stop participating in the research at any time without penalty or loss of other benefits. The project investigators retain the right to cancel or postpone the experimental procedures at any time they see fit.

By signing below, you acknowledge that you have been informed about and consent to be a participant in the study described above. Make sure that your questions are answered to your satisfaction before signing. You are entitled to retain a copy of this consent agreement.

____________________________________   Date: ____________________

Study Participant Signature

____

Study Participant Name (Please print)

_____________________________________   Date: ____________________

Signature of Person who explained this study

Appendix D. COVID-19 Risk Mitigation Protocol

COVID-19 Risk Mitigation Protocol

At WPI, our primary responsibility related to research is to protect the safety of our research participants.

COVID-19 refers to the coronavirus that is being spread across people in our communities. We need to provide you with important information about COVID-19, and to tell you about ways your study participation might change because of COVID-19 related risk.

If you are considering joining a study at this time or are currently enrolled in a study, **it is important that you consider the following information to determine if study participation is right for you at this time.**

How is COVID-19 spread?

COVID-19 is a respiratory virus spread by respiratory droplets, mainly from person-to-person. This can happen between people who are in close contact with one another (less than 6 feet). It is also possible that a person can get COVID-19 by touching a surface or object (such as a doorknob or counter surface) that has the virus on it, then touching their mouth, nose or eyes.

Can COVID-19 be prevented?

Current ways to minimize the risk of exposure to COVID-19 include “social distancing” which is a practice to decrease the potential for direct exposure to others who may have been exposed to COVID-19, for example by avoiding large gatherings or refraining from shaking hands with others. It is important to understand that since study participation may include increased travel outside of your home and increased exposure to others within a research site it may increase your exposure to COVID-19. At this time, there is no vaccination to prevent COVID-19 infection.

What are the risks of COVID-19?

For most people, the new coronavirus causes only mild or moderate symptoms, such as fever and cough. For some, especially older adults and people with existing health problems, it can cause more severe illness, including pneumonia. While we are still learning about this virus, the information we have right now suggests that about 3 of 100 people who are infected might die from the virus.

Who is most at risk?

Individuals over 60 and with chronic conditions such as cancer, diabetes and lung disease have the highest rates of severe disease from the infection.

What do we do to minimize risk for research participants?

a. All in-person research will take place on the WPI campus.

b. Participation in the study will be strictly limited to WPI students and faculty authorized to attend campus in-person.

c. **Research visits will strictly abide by all official WPI COVID-19 risk mitigation protocols in effect at the time of the test session.** These protocols specify campus-wide standards for minimizing the potential spread of COVID-19, including (but not limited to):

- Visitors allowed on campus
- Required vaccination status of visitors
- Masking requirements
- Social distancing requirements
- Maximum room occupancy requirements

A summary of current WPI protocols is maintained at this URL:

https://www.wpi.edu/we-are-wpi

d. **Regardless of current WPI protocols,** all test administrators and subjects will be required to wear a face mask at all times during the test session.

e. Test subjects will visit the research site only once, and only long enough to review the Informed Consent Agreement, participate in the test and respond to the research survey.

f. The location where study subject visits take place will have hospital-approved hand sanitizer readily available for use before and/or after the test session.

g. All physical equipment handled by subjects during the test (keyboards, mice, game controllers, headsets, etc.) will be thoroughly sanitized with alcohol wipes before each test session.

If you have further questions about COVID-19 and your participation in research, please talk to your study team.

_____________________________________ Date: ___________________

Study Participant Signature

____

Study Participant Name (Please print)

_____________________________________ Date: ___________________

Signature of person who explained this protocol

Appendix E. IRB Survey Instrument

Q1. How would you rate the difficulty of understanding the game's rules and instructions?

	1	2	3	4	5	6	
Difficult	O	O	O	O	O	O	Easy

Q2. How would you rate the difficulty of operating the game?

	1	2	3	4	5	6	
Difficult	O	O	O	O	O	O	Easy

Q3. How would you rate the total duration of the game experience?

	1	2	3	4	5	6	
Too short	O	O	O	O	O	O	Too long

Q4. Which of the following terms would you use to describe the game? Please check all that apply.

□ Tactical

□ Strategic

□ Puzzle-solving

□ Horror

□ Suspense

□ Simulation

□ Role-playing

□ Other (Please Specify) ____

Q5. Which of the following emotions did you experience while playing the game? Please check all that apply.

□ Anger

□ Anxiety

□ Boredom

□ Excitement

□ Disgust

□ Embarrassment

□ Fear

□ Grief

□ Joy

□ Relaxation

□ Sadness

□ Surprise

□ Other (Please specify) ____

Q6. How effectively did the game's mechanics invoke emotions you selected above?

	1	2	3	4	5	6	
Ineffective	○	○	○	○	○	○	Highly effective

Q7. How would you rate the effectiveness of the game’s visuals (modeling, layout, color, lighting, etc.) at creating an immersive experience?

	1	2	3	4	5	6	
Ineffective	○	○	○	○	○	○	Highly effective

Q8. How would you rate the effectiveness of the game's use of sound effects and music at creating an immersive experience?

	1	2	3	4	5	6	
Ineffective	O	O	O	O	O	O	Highly effective

Q9. How likely would you be to replay the game to explore different outcomes?

	1	2	3	4	5	6	
Not Likely	O	O	O	O	O	O	Very Likely

Q10. How likely would you be to recommend this game to a friend?

	1	2	3	4	5	6	
Not Likely	○	○	○	○	○	○	Very Likely

Q11 (optional). How would you describe the most satisfying way to play the game (if any)?

____

Q12 (optional). Please add any additional comments or suggestions here.

____

Q13 (optional; for IMGD students only). Please leave your email if you need credits.

____

Appendix F. Survey results

Q1. How would you rate the difficulty of understanding the game's rules and instructions?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Difficult: Easy	2.00	6.00	3.45	1.16	1.34	11

#	Answer	%	Count
6	6	9.09%	1
5	5	9.09%	1
4	4	18.18%	2
3	3	45.45%	5
2	2	18.18%	2
1	1	0.00%	0
	Total	100%	11

Q2. How would you rate the difficulty of operating the game?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Difficult: Easy	2.00	6.00	4.36	1.23	1.50	11

#	Answer	%	Count
1	1	0.00%	0
2	2	9.09%	1
3	3	9.09%	1
4	4	45.45%	5
5	5	9.09%	1
6	6	27.27%	3
	Total	100%	11

Q3. How would you rate the total duration of the game experience?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Too short: Too long	2.00	5.00	3.27	0.86	0.74	11

#	Answer	%	Count
1	1	0.00%	0
2	2	18.18%	2
3	3	45.45%	5
4	4	27.27%	3
5	5	9.09%	1
6	6	0.00%	0
	Total	100%	11

Q4. Which of the following terms would you use to describe the game? Please check all that apply.

#	Answer	%	Count
1	Tactical	0.00%	0
2	Strategic	11.11%	3
3	Puzzle-solving	11.11%	3
4	Horror	25.93%	7
5	Simulation	22.22%	6
6	Role-playing	0.00%	0
7	Other (Please Specify)	3.70%	1
8	Suspense	25.93%	7
	Total	100%	27

Other (Please Specify)

____

Q5. Which of the following emotions did you experience while playing the game? Please check all that apply.

#	Answer	%	Count
1	Anger	3.57%	1
2	Anxiety	21.43%	6
3	Boredom	10.71%	3
4	Excitement	14.29%	4
5	Disgust	7.14%	2
6	Embarrassment	3.57%	1
7	Fear	10.71%	3
8	Grief	0.00%	0
9	Joy	7.14%	2
10	Relaxation	0.00%	0
11	Sadness	0.00%	0
12	Surprise	10.71%	3
13	Other (Please Specify)	10.71%	3

	Total	100%	28

Other (please specify)

____

Q6. How effectively did the game's mechanics invoke emotions you selected above?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Ineffective: Highly effective	2.00	5.00	4.00	1.04	1.09	11

#	Answer	%	Count
1	1	0.00%	0
2	2	9.09%	1
3	3	27.27%	3
4	4	18.18%	2
5	5	45.45%	5

6	6	0.00%	0
	Total	100%	11

Q7. How would you rate the effectiveness of the game's visuals (modeling, layout, color, lighting, etc.) at creating an immersive experience?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Ineffective: Highly effective	3.00	6.00	4.45	1.08	1.16	11

#	Answer	%	Count
1	1	0.00%	0
2	2	0.00%	0
3	3	27.27%	3
4	4	18.18%	2
5	5	36.36%	4
6	6	18.18%	2

	Total	100%	11

Q8. How would you rate the effectiveness of the game’s use of sound effects and music at creating an immersive experience?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Ineffective: Highly effective	2.00	5.00	4.09	1.08	1.17	11

#	Answer	%	Count
1	1	0.00%	0
2	2	18.18%	2
3	3	0.00%	0
4	4	36.36%	4
5	5	45.45%	5
6	6	0.00%	0
	Total	100%	11

Q9. How likely would you be to replay the game to explore different outcomes?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Not Likely: Very Likely	1.00	5.00	3.91	1.38	1.90	11

#	Answer	%	Count
1	1	9.09%	1
2	2	9.09%	1
3	3	18.18%	2
4	4	9.09%	1
5	5	54.55%	6
6	6	0.00%	0
	Total	100%	11

Q10. How likely would you be to recommend this game to a friend?

#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Not Likely: Very Likely	1.00	6.00	4.27	1.29	1.65	11

#	Answer	%	Count
1	1	9.09%	1
2	2	0.00%	0
3	3	9.09%	1
4	4	27.27%	3
5	5	45.45%	5
6	6	9.09%	1
	Total	100%	11

Q11 (Optional). How would you describe the most satisfying way to play the game (if any)?

Sorting folders really quickly and glancing back as little as possible.

The most satisfying was to try and speed clear the objectives.

To be a paranoid person and always look behind you.

Check symbol - categorize - and then look back to kill the phantom.

The most satisfying way to play the game was when you start hearing stuff that don't really mean anything (false cues). It makes it more suspenseful.

Q12 (optional). Please add any additional comments or suggestions here.

The title is contradictory to the mechanics. There does not appear to be any time pressure for sorting the files and thus no reason to not always look back. Also, I want stuff to throw at the ghost!

Files could spawn in different place.

Louder sounds, I got the rose folder like 50 times, maybe give me other stuff that i need to manage, like checking emails? doing something with the computer? I might just be dead inside but scarier enemies.

If additional ways to get to know which file type it is - like through vibration or through color - it might help more.

When the room goes dark when sanity drops low, it becomes very hard to see the symbols for filing the folders.

Appendix G. Audio assets and sources

Status	Sound	File Name	Source	Notes
Ambient Sounds				
🟩	Raining 6	Ambient_Raining (6).mp3	https://www.ear0.com/sound/show/soundid-37330	
🟩	StartMenu-Sunset	Ambient_StartMenu.mp3	https://freesound.org/people/psubhashish/sounds/514496/	
🟥	Raining 0		https://www.ear0.com/sound/show/soundid-37423	
🟥	Raining 1		https://freesound.org/people/FlatHill/sounds/237729/	
🟥	Raining 2 (start raining)		https://freesound.org/people/roofusj/sounds/217235/	From small rain to medium rain
🟥	Raining 3 (small)		https://freesound.org/people/psuess/sounds/393483/	More like a peaceful mood, maybe not needed
🟥	Raining 4 (medium)		https://freesound.org/people/alexkandrell/sounds/316894/	
🟥	Raining 5 (medium2)		https://freesound.org/people/ivolipa/sounds/276928/	
Hint Sounds				
Collision:				
🟩	Door creaks 1	Door Open Close.mp3	https://freesound.org/people/InspectorJ/sounds/346212/	
🟩	Door close 1	Door Open Close.mp3	https://freesound.org/people/InspectorJ/sounds/411791/	
🟩	Collides with shelf 1	Collides with Bookshelf 1.mp3	https://freesound.org/people/JustInvoke/sounds/446125/	need to accompany with book drop down sound
🟩	Collides with cardboard box	Box1_Left/Right.mp3	https://freesound.org/people/kyles/sounds/450810/	

🟩	4			
🟩	Light flickers 1	Light flickers 1.mp3	https://freesound.org/people/mmaruska/sounds/232447/	need to be edited to form a feeling of "flickering"
🟩	Files collapse 1	File Collapse 1.mp3	https://www.ear0.com/sound/show/soundid-37388	can be used as files collapse, just need to be edited
🟩	Painting drops 1	Painting Drops 1.mp3	https://freesound.org/people/LightDJ/sounds/267481/	
🟩	Paper Fluttering	Paper_Fluttering_Left/Right.mp3	https://freesound.org/people/mickdow/sounds/320913/	
🟩	Books Falling	Book_Fall1_Left/Right.mp3	https://freesound.org/people/MindlessTrails/sounds/509531/	
🟥	Door creaks 2		https://freesound.org/people/JarredGibb/sounds/219499/	
🟥	Door creaks 3		https://freesound.org/people/JarredGibb/sounds/219492/	
🟥	Door close 2		https://freesound.org/people/InspectorJ/sounds/339677/	
🟥	Collides with shelf 2		https://www.ear0.com/sound/show/soundid-15047	need some edition
🟥	Collides with shelf 3			
🟥	Collides with cardboard box 1		https://www.ear0.com/sound/show/soundid-16325	need to be edited
🟥	Collides with cardboard box 2			
🟥	Collides with cardboard box 3			
🟥	Trash bin tips over 1		https://freesound.org/people/suoitnop/sounds/46873/	need to be edited
🟥	Trash bin tips over 2		https://freesound.org/people/Nestra/sounds/498434/	
🟥	Light flickers 2		https://freesound.org/peopl	kind of like the sound of current

🟥			e/scotchio/sounds/143915/	
🟥	Files collapse 2			
🟥	Files collapse 3			
🟥	Painting drops 2		https://freesound.org/people/LightDJ/sounds/267482/	
Illusion:				
🟩	Footstep 1	Footstep.mp3	https://www.ear0.com/sound/show/soundid-16532	need to be clipped
🟩	Breathing 3	Breath 3.mp3	https://www.ear0.com/sound/show/soundid-14824	
🟩	Vanishing 1	Vanishing 1.mp3	https://www.ear0.com/sound/show/soundid-37328	delete the second half, then it will be good
🟩	Vanishing 2	Vanishing 2.mp3	https://www.ear0.com/sound/show/soundid-37286	
🟩	Vanishing 3	Vanishing 3.mp3	https://freesound.org/people/szpury/sounds/236320/	
🟥	Footstep 2 (water)		https://www.ear0.com/sound/show/soundid-21431	
🟥	Footstep 3			
🟥	Footstep 4			
🟥	Footstep 5			
🟥	Breathing 1		https://www.ear0.com/sound/show/soundid-21638	
🟥	Breathing 2		https://www.ear0.com/sound/show/soundid-12403	
Other Sounds				
Work:				
🟩	Picking Up File 1	Picking Up File 1.mp3	https://freesound.org/people/IsakIzzy/sounds/572018/	need to be shorten
🟩	Picking Up File 2	Picking Up File 2.mp3	https://freesound.org/people/IsakIzzy/sounds/572019/	need to be shorten
🟩	Putting Down File 1	Putting Down File 1.mp3	https://freesound.org/people/jomse/sounds/428666/	
🟥	Putting Down File 2			

🟩	Sorting Successfully 1	Putting Down File 2.mp3	https://freesound.org/people/jomse/sounds/428663/	
🟥	Sorting Successfully 2			
🟥	Sorting Successfully 3			
🟩	Sorting Failed 1	Sort_Failed1.mp3	https://freesound.org/people/HipsterTypist/sounds/527491/	
🟥	Sorting Failed 2			
🟥	Sorting Failed 3			
UI:				
🟩	Alerting of Low Sanity 1 (heartbeats)	Heartbeats 1.mp3	https://www.ear0.com/sound/show/soundid-37268	volume need to be improved
🟩	Alerting of Low Sanity 2 (heartbeats)	Heartbeats 2.mp3	https://freesound.org/people/InspectorJ/sounds/485076/	
🟩	Win	Win Sound.mp3	https://freesound.org/people/Fupicat/sounds/521642/	
🟩	Lose	Lose Sound.mp3	https://freesound.org/people/Indian_gamer2005/sounds/548430/	
Environmental Distraction:				
🟥	Dripping 1			
🟥	Dripping 2			
🟥	Dripping 3			
🟥	Thunder 1		https://freesound.org/people/InspectorJ/sounds/360328/	
🟩	Thunder 2	Thunder 2.mp3	https://freesound.org/people/Josh74000MC/sounds/477840/	
🟩	Thunder 3	Thunder 3.mp3	https://freesound.org/people/bajko/sounds/399656/	
🟥	Floor creaks		https://www.ear0.com/soun	

🟥			d/show/soundid-10575	
🟥	Floor creaks 2		https://www.ear0.com/sound/show/soundid-21558	
🟩	Clock	Clock.mp3	https://www.ear0.com/sound/show/soundid-21627	it may be better to slow the sound
🟩	Phone Ringing	Phone Ringing.mp3	https://www.ear0.com/sound/show/soundid-37321	delete the first half, then it will be good

Appendix H. Art assets and sources

Status	Asset Name	Package Name	Source	Notes
Decoration:				
🟩	Skybox		https://assetstore.unity.com/packages/2d/textures-materials/diverse-space-skybox-11044	
🟩	Wall		https://assetstore.unity.com/packages/3d/simple-home-stuff-69129	
🟩	Door		https://sketchfab.com/3d-models/horror-game-floor-generator-free-1d9310352e9942528f4199b5c8e20a88	
🟩	Window		https://assetstore.unity.com/packages/3d/environments/urban/devden-archviz-vol-1-scotland-158539	
🟩	Table		https://assetstore.unity.com/packages/3d/environments/urban/devden-archviz-vol-1-scotland-158539	
🟩	Chair		https://assetstore.unity.com/packages/3d/environments/urban/devden-archviz-vol-1-scotland-158539	
🟩	Monitor Slim	Modern Office Props.unitypackage	https://aigei.com	
🟩	Desktop Tower	Modern Office Props.unitypackage	https://aigei.com	
🟩	Mouse Pad	Modern Office Props.unitypackage	https://aigei.com	
🟩	Mug	Modern Office Props.unitypackage	https://aigei.com	
🟩	Clock	Modern Office Props.unitypackage	https://aigei.com	
🟩	Trash bin		https://sketchfab.com/3d-models/trash-bin-96c6c18cff6b4e1abcdf9f85cb8c690d	
🟩	Brief Case	Modern Office Props.unitypackage	https://aigei.com	
🟩	Water Cooler	Modern Office Props.unitypackage	https://aigei.com	
🟩	Cabinet	Modern Office Props.unitypackage	https://aigei.com	
Collidable:				
🟩	Lamp		https://assetstore.unity.com/packages/3d/simple-home-stuff-69129	

🟩	Bookshelf1	Modern Office Props.unitypackage	https://aigei.com	
🟩	Bookshelf2	Modern Office Props.unitypackage	https://aigei.com	
🟩	Cardboard Box1		https://assetstore.unity.com/packages/3d/props/pbr-cardboard-box-110635	
🟩	Books	Modern Office Props.unitypackage	https://aigei.com	
Interactable:				
🟩	Headset	Modern Office Props.unitypackage	https://aigei.com	
🟩	Keyboard & Mouse	Modern Office Props.unitypackage	https://aigei.com	
🟩	Sorting Area/Folder		https://sketchfab.com/3d-models/several-folders-1a493b49ef954985ab8057ca66c387d5	
🟩	Files with seal		Made by Mingxi Liu	
File Seals:				
🟩	Flower image 1		https://588ku.com/ycpng/12866794.html	
🟩	Flower image 2		https://588ku.com/ycpng/12031904.html	
🟩	Flower image 3		https://588ku.com/ycpng/12154897.html	
🟩	Flower image 4		https://588ku.com/ycpng/12866795.html	
🟩	Stamp image 1		https://588ku.com/ycpng/12453549.html	
🟩	Stamp image 2		https://588ku.com/ycpng/12453549.html	
🟩	Stamp image 3		https://588ku.com/ycpng/12572077.html	
🟩	Stamp image 4		https://588ku.com/ycpng/12572129.html	
UI:				
🟩	Wall Screen	Modern Office Props.unitypackage	https://aigei.com	
🟩	Projector	Modern Office Props.unitypackage	https://aigei.com	
Other Models:				
🟩	Raining Effect VFX		https://assetstore.unity.com/packages/vfx/particles/environment/rain-35156	
🟩	Ghost		https://www.cgtrader.com/free-3d-models/character/clothing/3-black-hooded-capes	

🟩	Hands		https://professorpoly.gumroad.com/l/VRHandFBX	