UNIVERSITAT JAUME I

High School of Technology and Experimental Sciences

BACHELOR'S DEGREE IN DESIGN AND DEVELOPMENT OF VIDEO GAMES

FINAL DEGREE WORK

Developing An Escape Game & A Chatbot: Immersion At Its Finest

Presented by:
Mr. Mario Remacha Sidro

Supervised by:
Mrs. Mar Marcos López, PhD.

Academic course 2020 / 2021

CONTENTS

1. Introduction

1.1. Work Motivation

This project is focused in two main aspects: creating a horror video game and a chatbot related to it. In order to do that, first it will be needed to explain what an escape room is and how it works, as well as what a chatbot is and the technology it uses. At last, a study will be carried out in order to see if immersion affects human's capacity of concentration.

The project has been chosen mainly to create an escape room-related video game that includes a chatbot as the resource to which the player can go to get information about the game itself.

1.2. Objectives

The first objective of the project is to carry out research on escape rooms, how they work and how they are created to know exactly what is needed when designing the escape game. Also, the research will focus on escape rooms in the video game industry and the role of virtual reality in the future of the industry of escape room-themed video games.

As it has been said, knowing exactly how an escape room works from its first scratch design to its final form is essential to accomplish the next objective, which is creating a horror escape room video game. Immersion will be a mandatory concept to accomplish with the game, so every move made in the design process will take it into account to guarantee the best atmosphere possible inside the game.

Once the game is created, it will be chatbot's time. First, it will be needed to carry out an extensive investigation on this technology, the way it works and its approaches in order to know how to create a personalized chatbot later. Research will also pay attention to possible applications of chatbots in the fields of business, education and health.

When the research is over and learned, it will be time to create the video game-related chatbot. The idea is that it will be able to answer any question made by the user about the game and its puzzles, answering with clues that may help the player find the solution to his/her question.

At last, volunteers will be tested with the objective of collecting data on how tension and an immersive atmosphere can affect the gamers' capacity of concentration solving the enigmas and moving around the game's inner world.

1.3. Environment and Initial State

The project team will be formed by just one person who will start this project from scratch and will have every resource needed available in order to get the job done. There is no previous job done on any of the objectives of this project, so the initial state is completely void.

1.4 Structure of the Document

The first part of the document describes the planning process of the project, where every step of the development process is defined. It also includes a Gantt diagram that shows the time spent on each part of the work. After that, an evaluation of the resources is carried out to specify the hardware used to develop the project.

The next part of the project is focused on explaining the concepts of escape rooms and chatbots. This section reflects the investigation research on both concepts and focuses on the relevant information that is used later to develop the video game and the chatbot.

After that, the system analysis is carried out to have a global idea of how the game and the chatbot work and what is necessary to develop both of them.

Then, the game development process is explained showing the most relevant information about it. The same thing happens later with the chatbot development process, where its code is explained with great detail.

Lastly, the results of the study are shown and the conclusions of the entire project are commented. After that, the bibliography and references used during the writing of the document is included to finish with the project description.

2. Planning and resources evaluation

2.1. Planning

At the beginning of the planning process it was clear ~~is~~ that the research process needed to be the first thing to do in order to know exactly how to develop both the game and the chatbot. But considering that games usually take more time than it is expected, only the escape room-related research was carried out at the beginning of the process. The reason for this is not only because starting developing the game was a priority but also because doing the research on chatbots later would be good for the developer to not get tired and bored of non-stop creating the game itself.

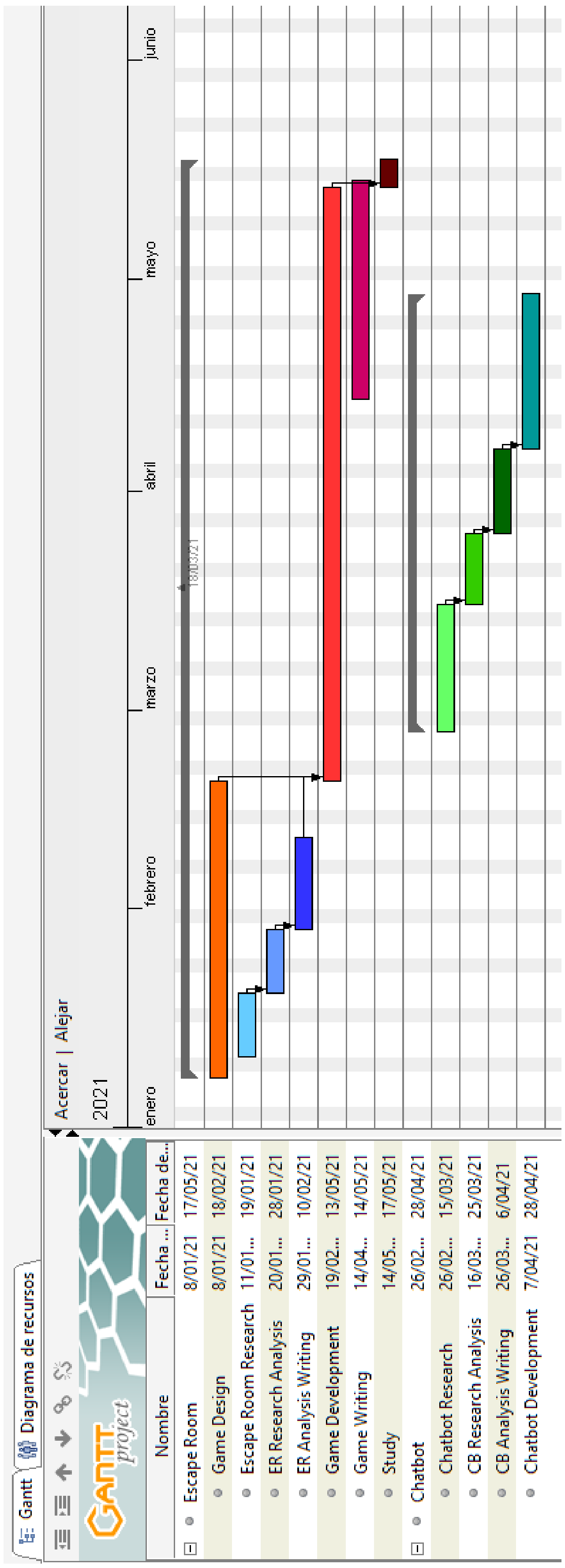
So, once the escape room research was over, synthesized, written and learned the developing process began. Considering that it is a very long task, the development of the project was thought to last almost the entire project creation, but as it has been said, while developing it the research on chatbots and the chatbot creation would also take place.

So some time after starting developing the game, the research on chatbots began. Unlike in the escape room topic (where the developer had a lot of knowledge), this was an extense process in the case of chatbots as much new information was being learnt. Also this topic is more complex than the other one, so it was harder to select the relevant information to be analyzed and synthesized. But once it was done, written and learned, the chatbot creation was ready to begin.

Having the knowledge fresh there was no time to lose in the chatbot development, which took more time than expected. While it was being coded, that same code was being written and explained in paper. Once all the chatbot creation ended, all the attention was focused on the game.

Once the game was finished, a group of volunteers were asked to test it, give feedback and also participate in the final study. Once it was all over, the most relevant aspects of the game were written while more people kept playing the game to give more representative data on the study.

All the planning process can be seen graphically in the following Gantt chart (*see Figure 1*).



Nombre	Fecha ...	Fecha de...
Escape Room	8/01/21	17/05/21
Game Design	8/01/21	18/02/21
Escape Room Research	11/01...	19/01/21
ER Research Analysis	20/01...	28/01/21
ER Analysis Writing	29/01...	10/02/21
Game Development	19/02...	13/05/21
Game Writing	14/04...	14/05/21
Study	14/05...	17/05/21
Chatbot	26/02...	28/04/21
Chatbot Research	26/02...	15/03/21
CB Research Analysis	16/03...	25/03/21
CB Analysis Writing	26/03...	6/04/21
Chatbot Development	7/04/21	28/04/21

Figure 1. Gantt chart of the project planning.

2.2. Resource Evaluation

The production cost of the project can be allowed by almost any developer. The human cost is just one unit of persons (with no salary at all) and the technological resources are Internet connection and a computer (the used one is an average computer), electricity supply and preferably an individual room to work. Also, the programs used are all free or have a free trial version, so anyone can use them. These programs are Unity, Visual Studio, Blender, Photoshop (free access with the trial version or the student version), PyCharm and Discord.

In general, anyone with an investment of around 1.000 euros can develop this project perfectly (the amount of money can vary depending on the components of the computer and the money the developer pays for the room, in this case it is 0 euros).

The specific resources are detailed in the following table.

Processor	Intel(R) Core(™) i7-8700 CPU @ 3.20GHz 3.19GHz
RAM	16.0 GB
Operating System	Windows 10 Pro 64 bits (x64)
Graphic Card	NVIDIA GeForce GTX 1060 3GB
Screen	Philips LED 243V 24 inches

3. Escape Rooms: The Mine Of Gold Of The Video Game Industry

3.1. Introduction

The popularity of Escape rooms has increased in the past few years and people are starting to think of them as a primary option to spend some time with their families and friends in their free time, so this part of the project is going to focus on what constitutes an escape room in a series of different sectors.

In a few words, escape rooms are interactive experiences in which players need to complete a series of challenges to succeed. Originally, escape rooms focused only on difficult logic puzzles, but nowadays the game has evolved into fully immersive environments with high quality settings that accompany the puzzles. The following pages will also explain the history of escape rooms and will provide an analysis of the design of puzzles and clues and the importance of immersion.

Once it is understood that the design of escape rooms is as complex as the way they can be played, a further analysis will be carried out to create a discourse around escape rooms and their use in other areas apart from the entertainment one, like education and corporate training.

At last, video games can be a perfect way to create escape rooms for the enigma lovers who want to try different options than the real-life ones. It will be analysed the role and importance of escape rooms in the video game industry and the increasing market of Virtual Reality escape rooms.

3.2. What is an Escape Room?

An escape room is a game played by a group of people that have to work together as a team to “escape” from a room or a series of rooms filled with challenges. To be successful in the mission, the team has to escape within a given time limit and solve all the puzzles and enigmas contained in the room/rooms. In the gaming context, escape rooms can be understood as Live Action Role Playing (LARP[1]) and Alternate Reality Games (ARG[2]).

In fact, escape rooms share a combination of different game concepts in one, like logic puzzles, board games, adventure games or interactive shows games. Considering this, it would be fair to think about escape rooms as an evolution of gaming itself and one of the most important products of modern game design culture.

Escape rooms always represent a challenge to players. Every room contains different types of challenges that make them think differently and always focus on puzzles with new

[1] A live action role-playing game (LARP) is a form of role-playing game where the participants physically portray their characters.
[2] An alternate reality game (ARG) is a type of multimedia game for multiple players that takes place in real time and evolves according to decisions taken by the players.

perspectives. They tend to be forced to think creatively and engage in critical thinking, because one never knows how the owner designed a certain puzzle, so every approach to which the player can get to must be explored in order to solve it and ultimately win.

Puzzles and enigmas usually have different styles of approach during the same room to avoid repetition and increase expectancy. The most used styles would be logic problems, mathematics problems, investigating and tracking, physical challenges and visual challenges. Despite this, styles and designs can go as far as the owner's imagination, but they must be connected to the main theme of the room. For example, if an escape room is set in a jungle, designing physical and tracking enigmas would be more adequate than mathematical puzzles.

Another important point is that every challenge must be doable by almost every person in the world. This means that, for example, usable objects cannot be too high (because short adult people or kids could never reach it) and logic enigmas must not require previous knowledge in order to be solvable. That is why if, for example, players find a chessboard in a room, solving the enigma should not imply knowing how to play actual chess because not every person in the world knows how to, so they must think of another way to solve it. Also, every enigma must be doable with the information and items the room contains, so the team does not need to bring anything from the outside to be able to get the win.

As it has been commented before, the design and style of an escape room can have a huge variety of forms, but almost every puzzle uses the same game loop (*see Figure 4*).

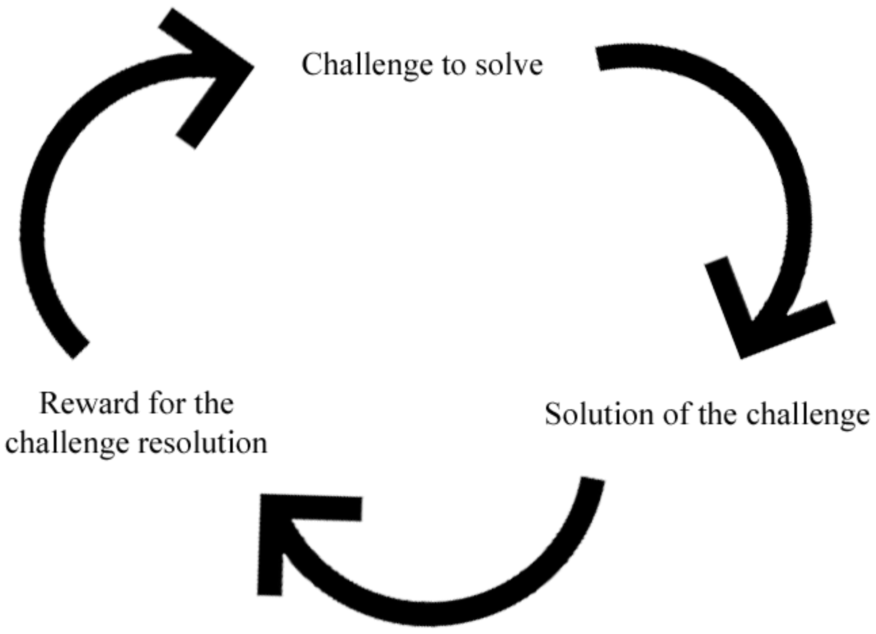

Challenge to solve
Solution of the challenge
Reward for the challenge resolution

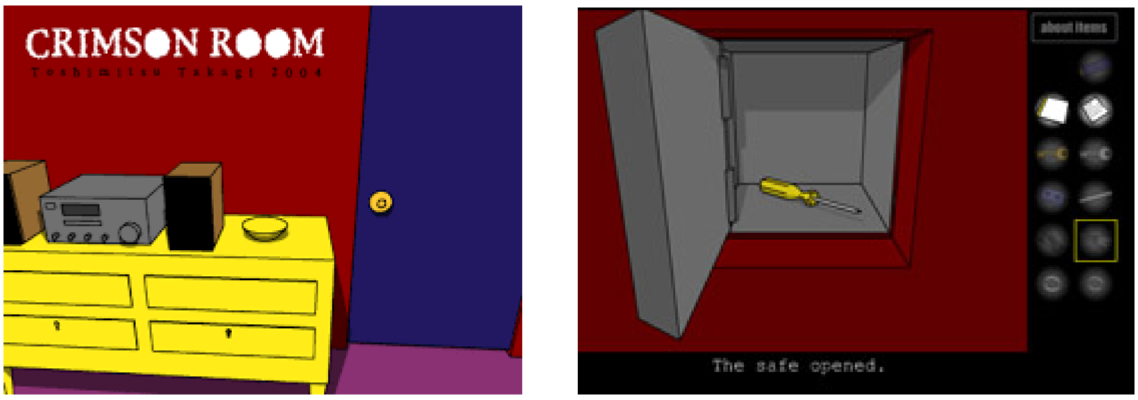
Figure 4. Every escape room puzzle's game loop.

The most common puzzle could be a drawer that is locked with a padlock. In this example, the challenge to solve is the locked drawer, the solution of the challenge is the combination of the padlock and the reward for the resolution is whatever the drawer contains, which in turn could be the challenge to solve for the next puzzle.

The puzzle loop above represents the most usual puzzle solving execution, but enigmas can be more elaborate as the reward could be more information for another puzzle and not necessarily a new challenge to solve, but in the end every single puzzle contains the three parts of the loop.

3.3. History of Escape Rooms and their Origin

Officially, the first time the concept "escape room" appeared was actually in the video games industry, where the players had to solve riddles by interacting with the environment in order to escape from a room and get to the next level. The first Escape Game video game was *Crimson Room* (*see Figure 5*), developed by Toshimitsu Takagi and released in 2004. In fact, the term "Takagism" was created to refer to this kind of games and it is especially used in the escape room fan base, and there is even an escape room club in China called "Beijing Takagism" that was founded in 2012.



The safe opened.

Figure 5. Crimson Room was the video game that defined the term "Escape room".

Once the Escape Game video games became a bit popular in Asia, the japanese company SCRAP decided in 2007 to change the concept by creating Live Room Escapes. SCRAP's founder, Takao Kato, wanted to put the players in a real-life situation of being trapped in a themed room with solvable puzzles as the only way to escape from the place. The history of escape games was then slowly developed in the rest of Asia before arriving in Europe and the rest of the world, where they are becoming more and more popular with the years.

As commented before, officially the term "escape room" appeared in 2004, but that is something arguable. In 1997, the movie industry released the film *Cube* (*see Figure 6*), directed by Vincenzo Natali. This film tells the story of a group of six people who are locked in a maze full of cubic rooms that hold mortal traps, and they have to solve a series of riddles in order to escape and survive.

Figure 6. Cube should arguably be considered the first escape room-related appearance ever.

Since Cube contains every characteristic of the modern concept of "escape rooms", it is fair to say that the term was born with this film long before the release of *Crimson Room*.

Talking about the film industry, even though *Cube* was quite relevant back in the day, there are no other big world-known successful movies related to this concept. The most distinguishable one would be *Escape Room*, whose synopsis is quite similar to the one from *Cube*. Another one is *Fermat's Room* (*see Figure 7*), one of the first escape room-related films. It was written and directed by Luis Piedrahita and Rodrigo Sopeña and released in the year 2007. It tells the story of four mathematicians that have to solve mathematical puzzles in order to escape from a deadly locked room.

Figure 7. Fermat's Room is one of the first escape room-related films that had international recognition.

3.4. Puzzle Path and Hints Design

Puzzle path design is crucial when it comes to designing an escape room. There are three different core approaches to creating a series of enigmas: linear path, open path and multi linear path.

In a *linear path*, the puzzles are done in order and one-by-one as the resolution of one puzzle will lead to the next one, and so on. This type of design makes it easier for the owners to design the room and for players to solve it, as they go all together and can contribute to the solving process. The disadvantage is that, in those puzzles that can be done by only one person (i.e. looking through some binoculars), the rest of the team remains doing nothing, which is not desired if the objective is to make them all live the complete experience. This path is the most common one, and it can be found in escape rooms such as *The curse of Amityville* from the company Ingenious, in Alicante.

The *open path* design includes puzzles that can be done in any order. This means that teams can divide in pairs or small groups to carry out more than one puzzle at the same time, to save time. This path is usually more difficult than the linear one, because teams have a lot of information at the same time and it takes them some time to realize which information is relevant for each puzzle. The main disadvantage of this method is that some team members might not participate in a certain puzzle because they were solving another one, which would mean that they will not live the full experience. This path is not as common as the linear one, but can be found in escape rooms such as *Excalibur* from the company The Lock Room, in Valencia.

Lastly, the *multi linear path* contains a series of linear path puzzles that can be done in parallel. Some escape rooms have more than one path that usually intersect with each other in another point of the experience, which can be presented to the players from the beginning of the escape room experience or at some point in the middle of it. In this kind of situations, teams have to divide to see what lies ahead of each path and continue with their mission. This path is the least common one, but can be found in escape rooms like *Alien: The Origin* from the company Escape Barcelona, in Barcelona.

But independently of the type of path an escape room is based on, there is one thing that is always present: the *game master*. In live escape rooms, the game master is a person that is looking at everything that is going on through hidden cameras placed inside the rooms. This person's mission is to help the players if they are stuck with an enigma, giving them hints related to that enigma. The game master is also responsible for greeting the players when they get to the place and introducing them to the experience, and also receiving them when they get out of the escape room.

Hints can be given in three different ways: the most common one is by voice, using walkie-talkies, telephones, through soundsets, etc. Another way is to give the clues through a screen in-game, which can reproduce videos, pictures or texts. The least common way is to deliver the hints underneath the door written in paper. The system used depends on the quality of the facility and the amount of money the owner has spent on creating the room.

The way the hints are given depends a lot on how the players are doing. If they are playing really well and ask for a hint, they will probably be ignored, so that they can keep on thinking of a solution because they still have a lot of time to end the experience successfully. Also, before the escape room starts, a team can clarify how many hints they want to receive (if any) and when. Succeeding in the mission without given clues is much more satisfactory than with, so most of the teams decide that they are the ones who will ask for clues, instead of the game master giving them whenever he/she wants.

In video games, the hint system is much more simple, as they do not have a game master. Most of the games include three hints for an enigma and the final solution for it. Each hint gives better advices the more the player asks for them, and if he/she does not solve the puzzle with the three of them, he/she can take the solution. Other games give general hints if the player asks for them, but usually he/she has to pay money in order to get many hints.

There is no escape room video game that contains a game master, above all not a single one that includes a game master that the player can talk to. That is why one of the main goals of this project will be to create an escape room video game that contains an intelligent game master to whom the player can talk to, but it will not be a human: it will be an artificial intelligence.

3.5. Immersion

Although everything that has been explained before is crucial for an escape room, the whole experience is much more than a simple puzzle game. Immersion is always a key factor in order to make the players completely engaged in the activity and forget everything outside the experience, and thus it will be one of the major focuses of this project.

Evoking emotions on people is the goal of most of the entertainment fields. A good movie can leave a mark on somebody, a good book can make a person imagine amazing worlds and stories… and escape rooms can transport players to those amazing worlds and situations to live the story in first person.

Mihály Csíkszentmihályi, a Hungarian-American psychologist, defined the term “flow”, saying that it is a mental state where people get so immersed in a task that they forget everything outside of it, even to eat and drink. He gave it this name because «it feels like water taking people down a stream of creativity». This effect manifests itself in many players when they focus on living the experience, being escape rooms one of the best examples of this effect.

Flow also empowers the feelings of awareness and action and combines them during the experience. It also makes people lose track of time and even feelings during the game, as they are challenged and entertained. If done wrong, for example a challenge is too hard or too easy, players could be frustrated or bored and reduce their flow. That is also why an accurate puzzle design is so important, as it was explained before.

An immersive experience can be achieved by combining the theme of the room, the decoration, the narrative of the story and the puzzles inside the room. Every escape room is based on a main theme that needs to be present through the entire experience to guarantee an accurate immersive experience. This means that everything inside the room (decoration, puzzles, etc.) must be related to the main theme, so the theme is the first thing that needs to be clear before designing an escape room. There are thousands of possible themes, as many as anyone can imagine, but some of the most used ones are asylums, pyramids, jungles, investigator offices, laboratories, haunted houses, and prisons (*see Figure 8*).

Figure 8. The theme of the escape room is one of the most important factors to guarantee a memorable experience

Ambience music and sounds are another key components of an immersive escape room. For example if an escape room is located in a jungle, hearing the wind, sounds of animals or a river contributes to creating an immersive atmosphere inside the room.

At last, the ultimate immersive resource is the use of an actor. In some escape rooms, the narrative requires an actor to interpret a character, so the game master dresses up and acts like that character during the entire experience. In some escape rooms, actors also participate inside the experience, most of the time in horror-themed ones, which adds a lot of tension to the experience. This project will evaluate how tension affects a person trying to solve enigmas and how it affects their senses and concentration.

3.6. Escape Rooms in the Video Game Industry

Since their appearance in 2004, escape room video games have grown slowly in popularity. In comparison with other genres, the “escape room” one has not developed as much as fans would like, majorly because most of them lack creativity. The way to play them is very similar to each other, being the puzzles and graphics the thing that changes the most.

One of the factors is that there have not been big companies that developed video games themed in an escape room until recently, so most of these games were created by small companies and indie developers[3]. As said before, the way to play these games is very similar. They show a scene of a room and an inventory in one of the sides of the screen, so every time the player gets a usable object it is stored in the inventory, and to use it he/she must click on it and then click the object where he/she wants to use it on (*see Figure 9*), a method that was first introduced by Crimson Room.

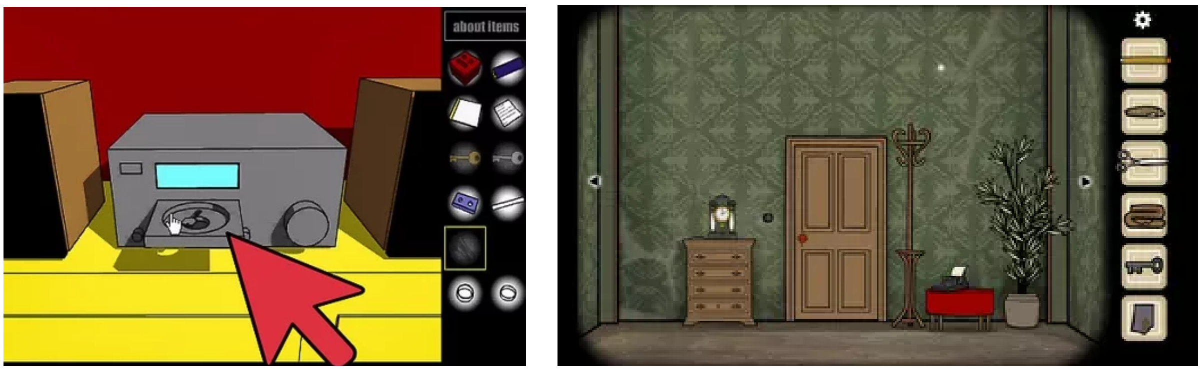


Figure 9. Crimson Room (left) marked the way escape room videogames work, a method taken by the majority of the same theme games, like the famous Cube Escape Paradox (right).

Most of the escape room-themed video games have been developed majorly in formats for Android, iOS and Windows. Internationally known consoles have never been a target for this kind of games, as the production is not affordable for most of the companies that develop these games. Of course, there are some exceptions like "The Room", developed and published by *Fireproof Games* in 2012. This game became very popular and by 2016 the company had already sold around 6,5 million copies around the world. This encouraged them to develop it for Nintendo Switch, being released in 2018 and receiving great reviews from the most important critic journalists in the world. This game also won many awards in 2012, like the "iPad Game of the Year" award, the *BAFTA* "Best British Game" award and the *New York Video Game Critics Circle Awards* "Best Mobile/iOS Game" award (*see Figure 10*).

Figure 10. The Room is the most world-known escape room-themed video game.

[3] An indie developer is a person or a small group of people that create video games without external financial support.

It was not until recent times that big companies got interested in escape room-themed video games, majorly inspired by the increasing popularity of virtual reality (VR). In 2017, Gartner released the “Hype Cycle for Emerging Technologies” from that year and placed the VR market as still an emerging technology, which could establish itself as a competitive technology in the market in a term between 2 to 5 years.

It was from 2017 to 2019 that the VR business start-up rate grew an 86’6%, which means that in just two years the amount of VR companies almost doubled. This event was a true achievement for the industry, and the next year’s “Hype Cycle for Emerging Technologies” deleted “Virtual Reality” from the status of emerging technology, which messed up a bit their previous prediction of two to five years instead of one.

Thanks to this advancement in the industry, most video games companies decided to create more video games playable with the VR technology, and some of them saw the opportunity to base them in the escape room theme. The idea is to create amazing immersive worlds and environments where players can live the experience as if they were actually there, and thus become a real competitor of the classic escape room video games.

Today, there are two different types of VR escape rooms that differ in the way the players have to move around the rooms. The most simple ones are those that implement the movement control in the actual controller. In these video games, the player can sit down in a chair or a sofa and move inside the game pressing a button and pointing to the position they want to move to. This case can be found in Ubisoft’s “Escape the Lost Pyramid”, an Assassin’s Creed-based game released in 2018 where players have to escape from a pyramid solving enigmas, doing parkour and shooting arrows with a bow. This game must be played by two or four people because players need to work together as a team to solve most of the puzzles (*see Figure 11*). This game became so popular that Ubisoft released in 2019 “Beyond Medusa’s Gate”, a new escape game, but it did not receive too good reviews because it was very similar to “Escape the Lost Pyramid”.

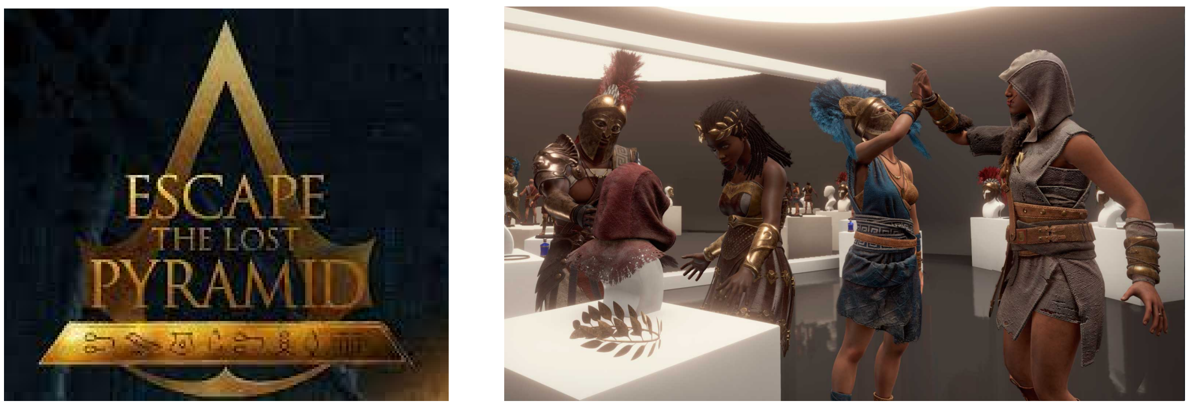


Figure 11. In Escape the Lost Pyramid, players can customize their characters before starting the adventure to make it more realistic and personal.

There are other VR escape rooms where the movement control is not carried out with a controller. These games take immersion to another level, as players actually have to move around the rooms. These escape rooms take place in dedicated places that have rooms prepared for this kind of games. With the recent improvement of VR glasses that do not have to be connected to the console via cable, players can move around the room exploring as if they were in a real escape room, but since this is a very new improvement of the technology, these rooms must allow for enough separation between players, to avoid rolling the cables in the air and ruining the experience and possibly breaking the glasses (*see Figure 12*).

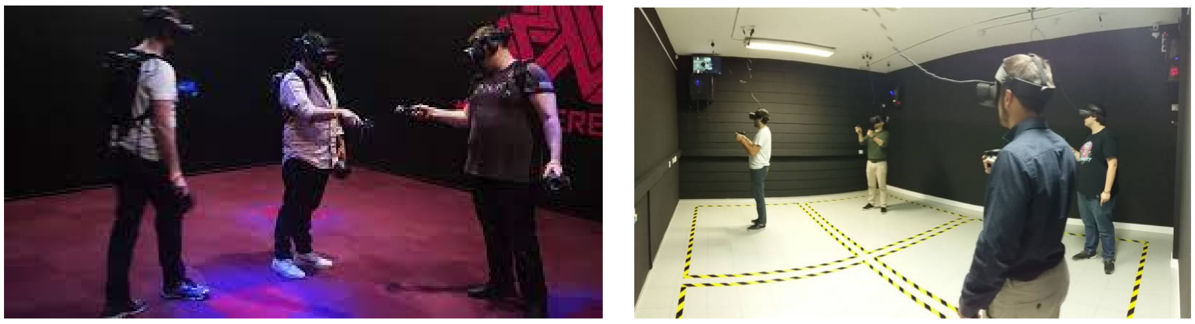


Figure 12. Virtual reality escape rooms are revolutionizing the escape game market.

As a curiosity, in Castellón there is a company called *Utopic Rooms* that has developed an original VR escape room game called “Morgan’s Chamber” that is playable for two people and that takes place in a room similar to the one on the right of the *Figure 12*.

4. Chatbots: The Game Master No One Thought About

4.1. Introduction and Definition

In 1968, Stanley Kubrick shocked the world with his movie “2001: A Space Odyssey”, a science fiction film that left a huge mark in the film industry. In this movie, there was a last generation computer that had the ability to see and listen, which allowed it to maintain conversations with humans. This computer was called HAL 9000, and at some point in the film it feels in danger and acts in consequence. This means that HAL was an artificial intelligence that could think, talk and feel. But it was not real, it was just fiction (*see Figure 13*).

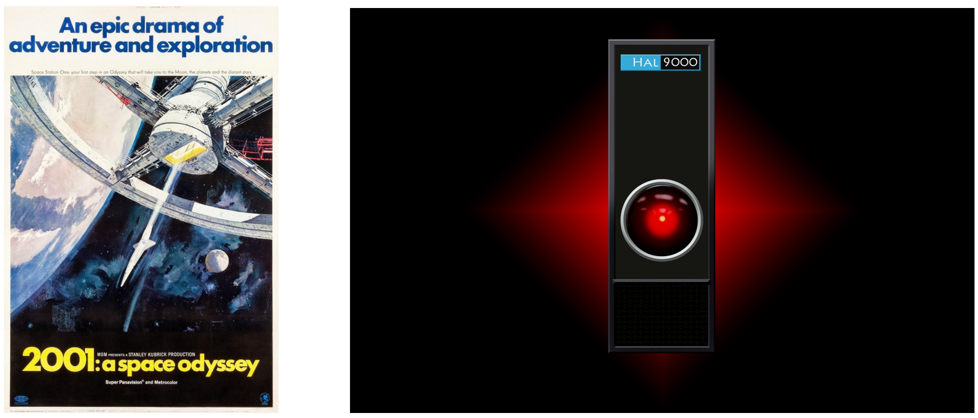


Figure 13. In “2001: A Space Odyssey”, HAL (right) could maintain conversations with humans.

Nowadays, robots still cannot feel emotions, but talking is a reality. A chatbot is an artificial intelligence software that can maintain conversations with human people in natural language. Chatbots exist in certain applications, websites and even in physical robots, and their existence is the first step in most engineer's dream of creating a real HAL.

Chatbots are usually described as one of the most advanced and promising expressions of interaction between humans and artificial machines. To communicate, they use Natural Language Processing (NLP), a technologic field that shares computer science, artificial intelligence and linguistics and that studies the interactions between computers and human language.

As a chatbot can be programmed to talk about any topic, their use can have an important role in fields like education, health and business apart from entertainment. Interactive agents, smart bots or digital assistants are considered chatbots. In fact, chatbots are becoming a very used resource in business online activities, because they reduce the cost of hiring a person to do an informative job and also might provide a more efficient assistance doing the communication as they provide concrete answers to the customer’s problem or question.

Also, most customers prefer talking to a chatbot than searching a company's frequently asked questions (FAQs) to save them time looking for a concrete answer.

On the other hand, there are also many customers that prefer to talk with a person rather than a bot with informational purposes. Some of them might think that bots are less informed or that they are rudimentary technology and get upset if they discover that they are talking to one, which could make them not buy the chatbot's company products. In fact, in an essay written in 1970, the japanese robotics professor Masahiro Mori developed the concept "uncanny valley", which is the uncomfortable feeling that a person can experience when he/she does not know if he/she is talking to a robot or a human.

But chatbots are not only used as a tool to achieve an informational purpose. In may 2017 in the *CHI Conference on Human Factors in Computing Systems* in Denver (Colorado, USA), Anbang Xu affirmed that 40% of users use chatbots for emotional purposes. That is because, in fact, the behaviour of chatbots can evolve thanks to their Machine Learning[4] features and their ability to analyse feelings, so they can respond emotionally to users that need emotional support.

Another reason why people appeal to chatbots with emotional purposes is that they normally play the passive role in the conversation. Chatbots are good listeners and unselfish companions, and as conversations can last as much as the user wants, they are a nice resource for this kind of assistance.

There are many categories of chatbots based on different factors that can be complementary, so every chatbot can belong to more than one category, but the two most relevant ones are categorized based on their range of knowledge and the purpose with which users use them.

In terms of the range of knowledge it is possible to find three categories: generic, open and closed domain. The generic ones can answer any user question from any topic at all and discuss about everything, the open domain ones can talk about more than one topic but not every topic and the close domain ones can answer questions about one topic in particular.

In what comes to giving answers, chatbots can be informative (users use them to learn something about a topic in particular with learning purposes), conversational (users use them to maintain conversations just like they would do with a human) and task based (users use them with commercial purposes, like booking a service).

So, in the case of a chatbot game master of an escape room video game it should be a closed domain informative chatbot.

[4] Machine Learning is a scientific discipline from the artificial intelligence field that gives to artificial systems the ability to learn.

And just because chatbots are perfect listeners, they could be great escape room game masters. In the following pages it will be explained with further detail how chatbots have evolved since their first appearance, how they are made, what kind of chatbots exist and the applications they might have.

4.2. History of Chatbots

The first research ever that tested a machine to measure its intelligence was Alan Turing[5] in 1950, when he put in practice an experiment known as “Turing Test” in which an interrogator had to guess between two participants which one was a human and which one was a machine by asking them questions. His intention was to try to define a standard by which a machine could be considered “intelligent”.

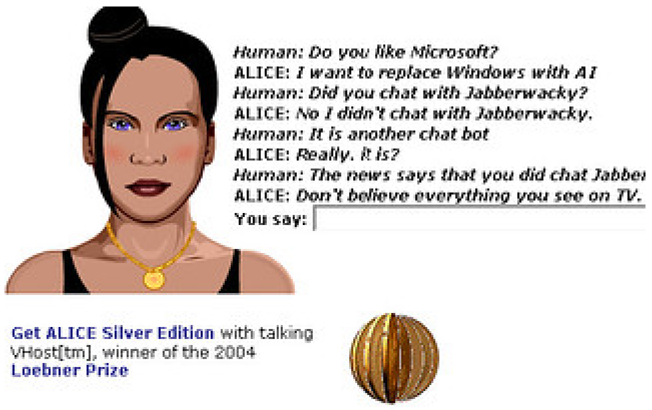
Even though Touring’s experiment was very criticized, some governments and companies started developing “intelligent” systems. ELIZA was born in 1964 in the Massachusetts Institute of Technology (MIT), and it was the first machine ever that was capable of talking. This machine had the role of a psychotherapist, and its function was to listen to the patient’s words and return them in the interrogative form. Of course, its ability to communicate was very limited as it couldn’t answer intelligent responses, but it was considered a success and served as an inspiration for next experiments.

Later, in 1972, American psychiatrist Kenneth Mark Colby developed a chatbot called PARRY at Stanford University, in California. Taking in consideration that ELIZA was a psychiatrist, PARRY was developed to take the role of a person with paranoid schizophrenia, and it was described as «ELIZA with attitude». These two chatbots met several times, and the most famous one took place at the *International Conference on Computer Communications* in Washington DC in 1972.

Several years later, in 1988, british programmer Rollo Carpenter developed Jabberwacky, a chatbot that simulated natural human chat in an humorous and ironic way (*Figure 2*). Its code was written in CleverScript, which is a language based on spreadsheets and that used contextual pattern matching to respond to the users based on previous talks and discussions.

In 1995, american programmer Richard Wallace created ALICE (Artificial Linguistic Internet Computer Entity), the first online chatbot. Inspired by ELIZA, Wallace created a new language for this one purpose and called it Artificial Intelligence Markup Language (AIML) which gave ALICE the ability to discuss with users for long periods of time about any topic. Despite this, this chatbot did not have intelligent features and could not express emotions in its answers. Being improved every year since its release, ALICE has won the Loebner Prize in 2000, 2001 and 2004 (*see Figure 14*).

[5] Alan Turing (1912 - 1954) was a british mathematician and one of the fathers of computer science. He deciphered the secret Nazi codes contained in the Enigma machine, which determined the course of World War II in favour of the Allies. The british film *The Imitation Game*, directed by Morten Tyldum, narrates this heroic event.



Human: Do you like Microsoft?
ALICE: I want to replace Windows with AI
Human: Did you chat with Jabberwacky?
ALICE: No I didn't chat with Jabberwacky.
Human: It is another chat bot
ALICE: Really. It is?
Human: The news says that you did chat Jabbe
ALICE: Don't believe everything you see on TV.
You say:

Get ALICE Silver Edition with talking VHost[tm], winner of the 2004 Loebner Prize

Figure 14. ALICE was the first online chatbot ever.

The first assistant chatbot was SmarterChild, created by the american company ActiveBuddy Inc. It was released for American Online (AOL) and Microsoft Messenger (MSN), being both of them messenger applications. ActiveBuddy was able to give to the users information about recent news, sports events and scores, information about the weather and other routinary information.

Since then, most world-known chatbots have been created by companies with business purposes. Most of them work as personal voice assistants, like Apple's Siri, IBM's Watson, Microsoft's Cortana, Google Assistant or Amazon's Alexa. All of these chatbots perform the same functions and serve as assistants for the users of each company's products.

4.3. Approaches of Chatbots

When it comes to developing a chatbot, there are two approaches to define the code and the used algorithms, which are the pattern matching and the machine learning approaches.

4.3.1. Pattern Matching Approach

Pattern matching algorithms search for any kind of pattern in a given sequence of tokens. These patterns are usually sequences, in the case of chatbots these algorithms try to find a pattern in the user's questions. It is the rule-based chatbots that are the most remarkable for this approach, as they try to match the user's input to a pattern and then selects an answer from a previously defined set of responses. The first chatbots that used pattern matching were ELIZA and ALICE.

These chatbots need as much database capacity as possible to answer the user's questions accurately. The most important and used pattern matching language is Artificial Intelligence Markup Language (AIML), developed in 1995. It is based in eXtensible Markup Language (XML) and its data is divided in topics of categories the chatbot itself can talk about with the users. Each category has a pattern that represents the user's input and a template to describe the response from the chatbot. All the data the chatbot possesses is stored in something called

Graphmater, which is usually represented as a tree in which the categories are drawn as nodes and the chatbot responses represented as its leaves (*see Figure 15*).

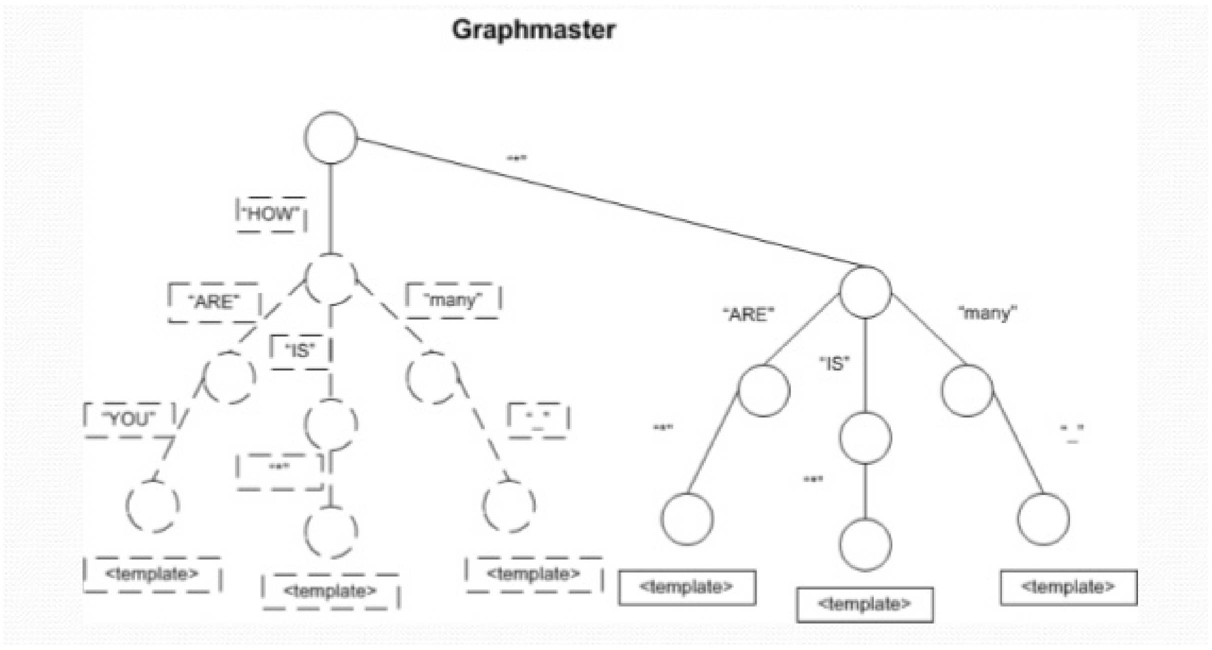


Figure 15. Graphmaster representation.

With this tree in mind, every time the user emits a question or a sentence the chatbot searches for the best pattern inside the Graphmaster.

4.3.2. Machine Learning Approach

Machine learning consists in giving a machine the ability to learn for itself without the need of any human to improve its algorithms. But to be able to learn, machines that use this approach need to be trained and do not require predefined answers to talk to the user. As it was introduced previously, chatbots that use this approach use Natural Language Processing (NLP), an AI technology that studies the interactions between humans and machines.

Also, these kinds of chatbots can use any kind of Artificial Neural Networks. This system works taking the user's input, then computes its vector representation to feed it to the neural network as a feature to produce a response in return. These networks are trained using both the intents vector and the user's input, taking that user's input and giving a probability of every intent to choose the answer.

Machine learning chatbot developers can also use Recurrent Neural Networks (RNN) to take in consideration the context of a conversation. This system uses the information that the chatbot has used and learned from previous conversations with new users as well as the context to communicate knowledge from the previous part of the network to the current one. There is a particular type of RNN called Long Short Term Memory (LSTM) networks that can refer to previous information to create answers, like for example responding something like “Yesterday I was talking about this with a guy and [...]”. RNN chatbots are mostly used for FAQs or informational chatbots, so they can learn from every conversation about a certain topic and also remember the information they have previously spoken about.

Another model used in machine learning is the Sequence-to-Sequence (Seq2Seq), which based on a source sequence (the user’s message) generates a target sequence (the chatbot’s answer). These chatbots improve their humanistic performance the more parameters they have.

4.4. Chatbots Development Process

Nowadays, chatbots can be created in many different languages (like Java or Python) or even inside platforms that are open-source, which gives many facilities to the creators to develop the chatbots as they get most of the skeleton of the code (like RASA or Chatterbot). This project is going to focus majorly on chatbots created in programming languages.

The most important thing to take into account when creating a chatbot is what its utility and functionality is going to be in order to decide the procedure used. The most used way to train a chatbot consists on making them talk to people mostly on web platforms, as it is easier to reach chatbot users there. The thing is that chatbots trained this way, as they speak with many different people about many different things, usually lack of personality.

The other way to train chatbots is making the developer or group of developers and volunteers talk to the chatbot about specific topics and conversations. With this system the chatbot ends up having a more remarkable personality, but the time spent training it is much higher than with the previous method.

Lately, chatbots have gained popularity in messaging applications like Facebook and Twitter, and companies around this sector are starting to add them to their systems to help the users giving them information or attention. This is something that developers have to take into account when creating a chatbot, which is the place where the chatbot will “live” and create the code according to the chosen application’s format.

4.5. Chatbots’ Role in Education and Health Fields

Education and health are very important topics in today’s society, and the use of chatbots in both of them can be very interesting and beneficial. In education, chatbots would have the ability to provide personal assistance to every kid and also educational content and attention. They can also save the information about every class and repeat it for the children that have missed a class and answer every question that the kids would ask being certain that the information they are answering is correct.

Also, chatbots could help students from a foreign country follow a class by translating what the teacher is saying in real time and facilitate the communication between the teacher and the kid by translating their conversation. Another way it could be used would be to make the chatbot interact with the students and take the role of the teacher. In fact, a research performed by Jiyou Jia in the University of Augsburg (Germany) in 2004, a chatbot was introduced to English students to talk to them and produced responses taking into account the context of the conversation and the user's English level and personality. Results showed that learners felt more comfortable talking to a bot because they felt less pressure and felt less judged than talking to a human, and also could talk about any topic and, as their tension level decreased, their English pronunciation was often better than talking to humans.

On the other hand, in the health field, chatbots are designed to customize the patient's health protocol and also provide health information like diagnosis or treatment suggestions based on the patient's symptoms. Some examples are OneRemission (which informs people about the cancer topic), Florence (which works as a reminder for patients and tells them to take the pills they need at the set time of the day) and Youper (which helps the user with his/her emotional health). Also, with the current Covid-19 pandemic, a chatbot called HealthBuddy was created to give the users information about the virus.

On the patients, a chatbot can create two different opinions about them. Some people see them as a more trustworthy companion than doctors and nurses as they usually share more information about their condition or illness and include more details on their symptoms. But unluckily, most patients see chatbots as cold and unknown companions and prefer being treated and diagnosed by a real person. Research has shown that younger people tend to trust in chatbots more than older people for medical reasons.

But in the medical field chatbots do not need to only be part of the medical team, they could also have use in administrative tasks like arranging appointments, sending emails, keeping the calendar structured, etc.

In the end, even though today the use of chatbots in both these fields is still very primitive, a higher use could be very beneficial for both users and professionals.

5. System Analysis and Design

5.1. Requirement Analysis

5.1.1. Functional Requirements

Functional requirements define functions that are going to be developed during the project creation. They usually include a set of the function's inputs, the outputs and its behaviour. The main functional requirements of the project make reference to the developed game and the developed chatbot, as they are the ones that have functionalities. The main functional requirements about the game would be:

Input	Enter the main menu
Output	Show the menu's image and buttons
Behaviour	When the user clicks on the game's executable file, the game will open accurately showing the main menu on the full size screen.

Input	Move the character
Output	Modify character's position
Behaviour	If the player presses the "W", "A", "S" or "D" keys, the character ingame will move.

Input	Rotate camera
Output	Modify character's rotation
Behaviour	When the player moves the mouse around, the game's camera will rotate to the same direction of the mouse's movement.

Input	Select item
Output	Activate event
Behaviour	When the player clicks on an item, the game will show a response to that action (grab the item, show a puzzle, etc.)

Input	Shoot
Output	Instantiate a bullet
Behaviour	When the player uses the shotgun, it will instantiate a bullet that will move to the front until it collides with another object.

On the other hand, the main functional requirement of the Chatbot is:

Input	Ask a question
Output	Give an answer
Behaviour	When the user sends a question to the chatbot, it will respond with a certain answer depending on the topic and the puzzle that the user is talking about.

5.1.2 Non-functional Requirements

These requirements define the limitations and rules of the system. This means that the non-functional requirements of the game are:

1. The game will only be available for Windows, iOS and Linux.

2. Every object of the game will only have one use.

3. The camera allows the player to rotate it as much as a human's neck would.

4. The rate of fire allows the player to shoot one bullet per second, which means that if he/she tries to shoot more than one time the game will only represent the first click in that short time.

The primary non-functional requirements of the chatbot is:

1. The chosen answer will be based on a series of parameters from the question and the inner chatbot's working. If the question has nothing to do with the game, the chatbot will answer it with a message saying so.

5.2. System Design

The behaviour and structure of the game can be seen in the following Use Case Diagram (*see Figure 2*).

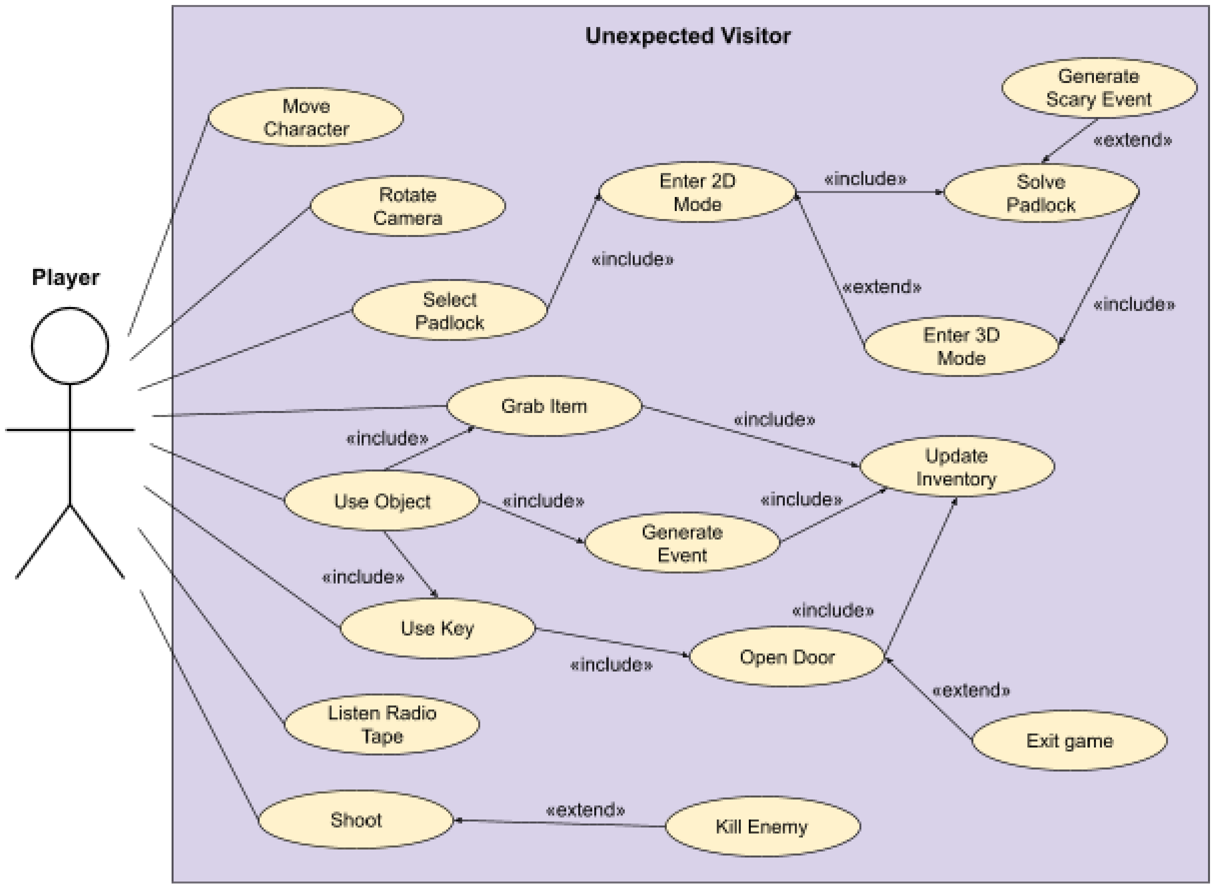


Figure 2. Project's video game's Use Case Diagram.

Every Use Case is performed by the Player actor, and the Use Case represented are:

Use Case Identificator	UC01
Use Case Name	Move Character
Description	It allows the player to move around the map.
Preconditions	The game state is in 3D mode and the player has pressed a movement key.
Basic Sequence Steps	Player's position in the X,Y and Z axis is modified.
Alternative Sequence	If the player is in the 2D mode, he/she will not move.

Use Case Identificator	UC02
Use Case Name	Rotate Camera
Description	It allows the player to rotate the main camera.
Preconditions	The game is in 3D mode and the player has moved its mouse.
Basic Sequence Steps	Player's rotation point is modified, directing it to the mouse's direction.
Alternative Sequence	If the player is in the 2D mode, he/she will not rotate. Also, if he/she tries to rotate more than a human neck would, the camera will not rotate more than its maximum.

Use Case Identificator	UC03
Use Case Name	Select Padlock
Description	It allows the player to click on a padlock to show its 2D form.
Preconditions	The player has clicked in a padlock.
Basic Sequence Steps	The player has pointed the camera in the padlock's direction, and then has clicked on it.

Use Case Identificator	UC04
Use Case Name	Enter 2D Mode
Description	The game shows the 2D canvas containing the padlock in its 2D version.
Preconditions	The player has clicked the padlock.
Basic Sequence Steps	The game manager has placed the 2D canvas on the screen and has frozen the time in the 3D world.

Use Case Identificator	UC05
Use Case Name	Solve Padlock
Description	The player manages to get the combination right.
Preconditions	The player has clicked a padlock and written its combination.
Basic Sequence Steps	Whatever the padlock was holding locked opens.

Use Case Identificator	UC06
Use Case Name	Enter 3D Mode
Description	The player has solved the combination or has closed the 2D menu.
Preconditions	The player was in the 2D mode.
Basic Sequence Steps	The time in the 3D world goes back to normal and the player can move and rotate again.

Use Case Identificator	UC07
Use Case Name	Generate Scary Event
Description	Some objects change their position to scare the player.
Preconditions	The player has solved the padlock.
Basic Sequence Steps	When the player solves the padlock correctly, some objects change their position.
Alternative Sequence	Not every padlock has a scary event addressed to it, so in most cases this Use Case will not happen.

Use Case Identificator	UC08
Use Case Name	Get Object
Description	A clickable object has been selected then taken to the inventory.
Preconditions	The player entered the game.
Basic Sequence Steps	The player has pointed the camera in the object's direction, then has shown the cursor and then has left clicked.

Use Case Identificator	UC09
Use Case Name	Update Inventory
Description	A clickable object has been selected then taken to the inventory.
Preconditions	An object has been clicked.
Basic Sequence Steps	The player has pointed the camera in the object's direction, then has shown the cursor and then has left clicked.

Use Case Identificator	UC10
Use Case Name	Use Object
Description	An object from the inventory can be used in another object.
Preconditions	The player has previously taken an object from somewhere else and now is pointing at this object.
Basic Sequence Steps	After picking up another related object, the player has pointed the camera in this object's direction, then the game has shown the cursor and then he/she has left clicked.

Use Case Identificator	UC11
Use Case Name	Generate Event
Description	When an object is used on another object, an event happens. This event can be many things, like breaking a glass or playing a film.
Preconditions	The player has used one object on another object.
Basic Sequence Steps	Once the player uses the object, an event is generated in the scenario.

Use Case Identificator	UC12
Use Case Name	Exit Game
Description	When the first clicked object is the last key and the second clicked object is the last door, the player can leave the game.
Preconditions	The player has got the last key in the inventory.
Basic Sequence Steps	The player gets the last key, then moves to the last door and clicks on it.

Use Case Identificator	UC13
Use Case Name	Shoot
Description	The player shoots a bullet from his/her shotgun.
Preconditions	The player has taken the shotgun from one of the last puzzles.
Basic Sequence Steps	The player has gotten the gunshot and then has right clicked.

Use Case Identificator	UC14
Use Case Name	Kill Enemy
Description	The player has shot the enemy the amount of times it gets for him to die.
Preconditions	The player used his/her shotgun.
Basic Sequence Steps	The player has gotten to the last room and has succeeded in the battle with his/her enemy.
Alternative Sequence	This will only happen when the enemy's health gets a value of 0.

At last, the Class Diagram helps represent the design system of the game, in which the player can collect objects and store them in an inventory, can solve puzzles to unlock padlocks and can fight an enemy (*see Figure 3*).

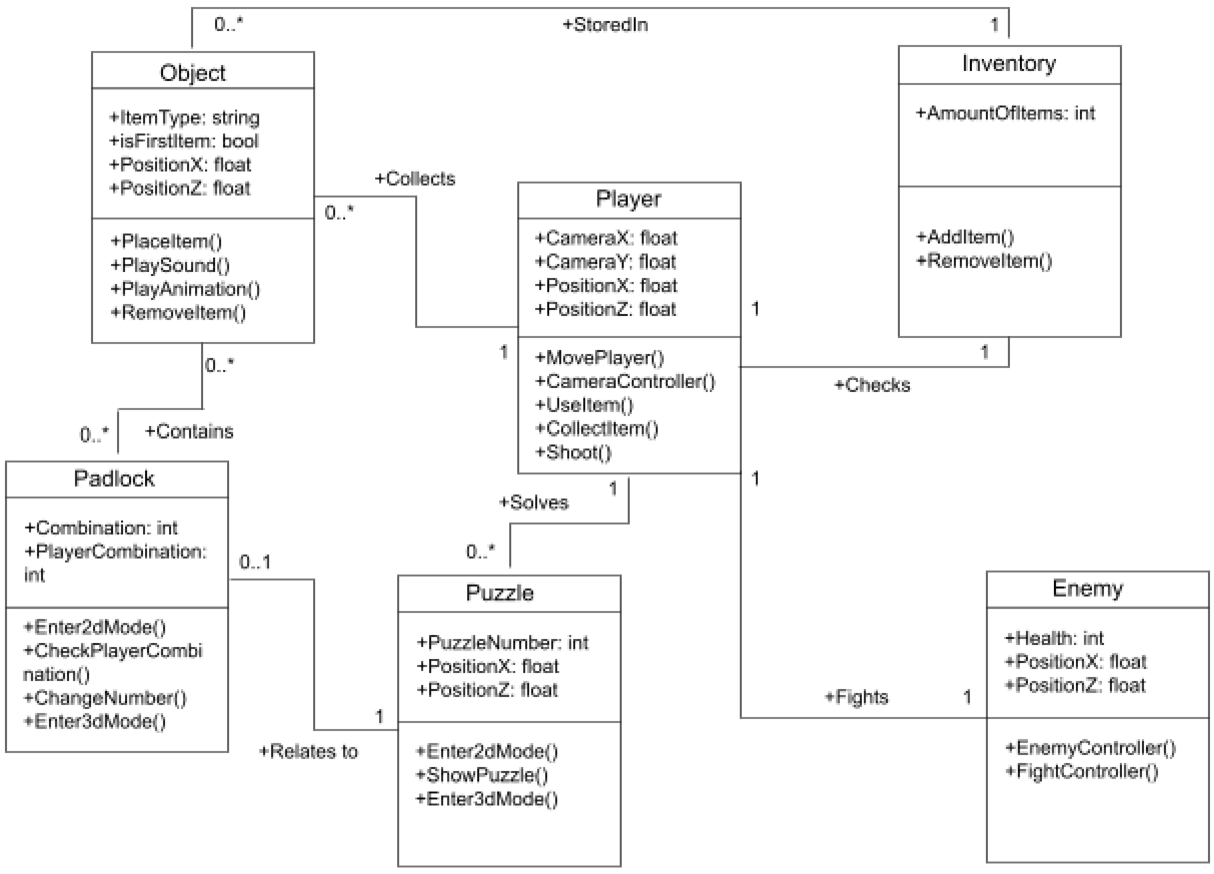


Figure 2. Project's video game's Class Diagram.

6. Unexpected Visitor: Horror Immersive Escape Game

6.1. Concept and Preparation

Unexpected Visitor is the main piece of this project. It has been developed using Unity as game engine and Visual Studio to write the code in the C# programming language. It is a horror escape room game in which the player has to escape from a haunted orphanage solving puzzles and enigmas.

The main goal of the game is to create a fully immersive environment where the player can believe that he/she is actually there living that experience. For this purpose, a realistic ambience had to be achieved. To do that, it was necessary to carry out a deep search of the most appropriate assets that the design required. As every room needed to be different from the others, to avoid repetition and bore the player, the assets needed to be related to different types of rooms, like an office, a kitchen, a classroom, etc.

Although during the level building it was needed to look for more assets, first it was necessary to design and describe the different spaces of the orphanage (*see Figure 16*).

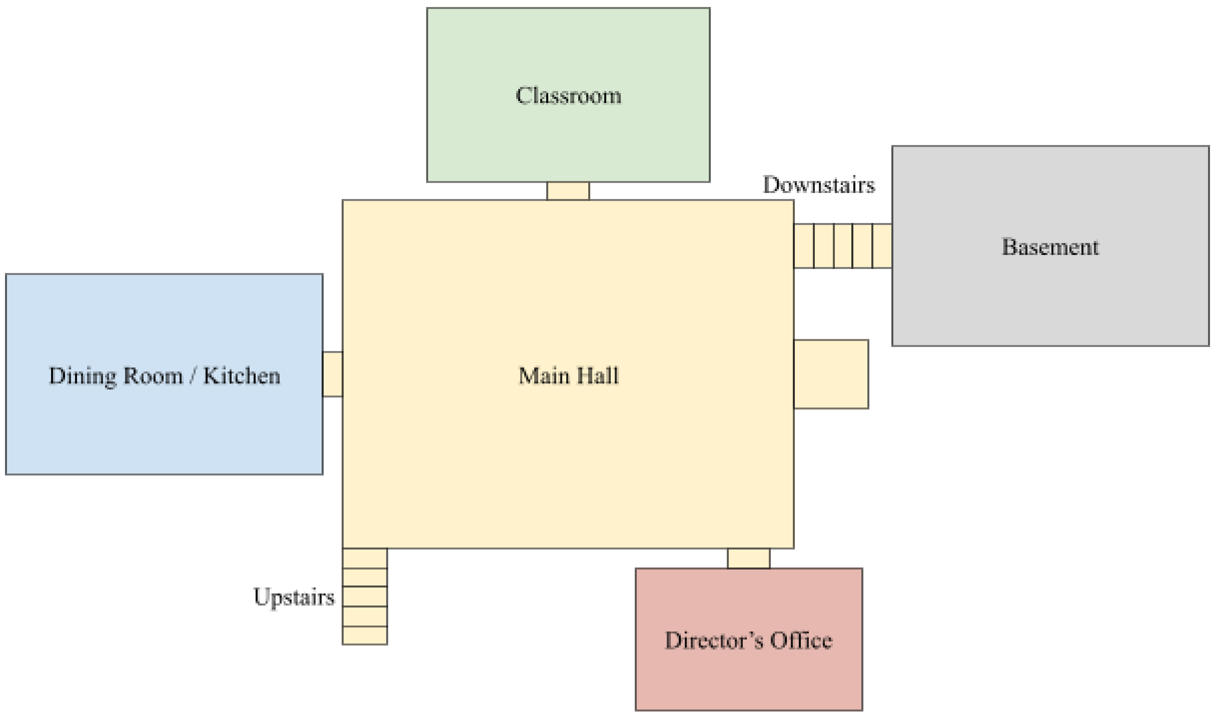


Figure 16.1. First floor and basement of the orphanage.

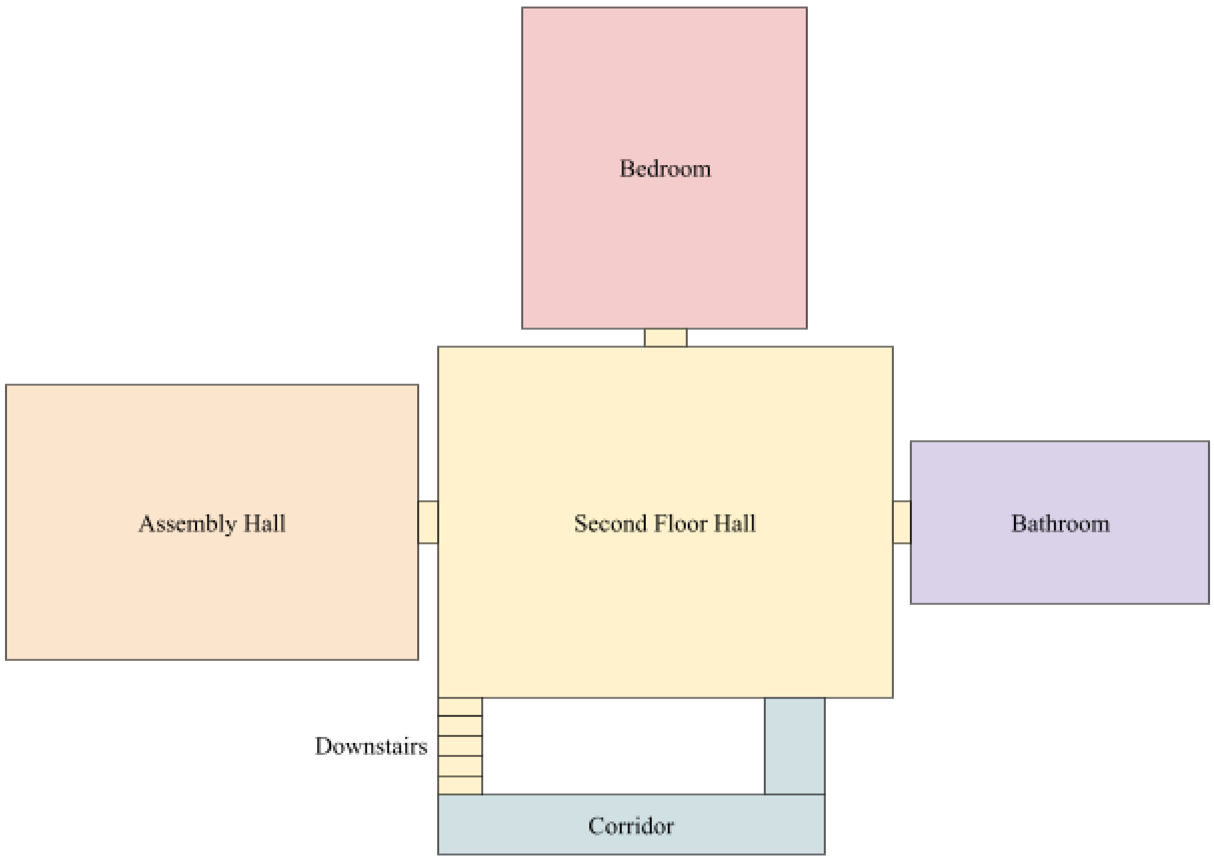


Figure 16.2. Second floor of the orphanage.

Also, to know exactly which assets were needed, it was essential to think about every puzzle the game was going to include. The next step, once every room's concept was clear, was to create some original puzzles. To create an appropriate immersive atmosphere around mystery, the level of tension of the game had to go from low to high, which means that the level of difficulty of the puzzles needed to go from high to low.

Otherwise, if the player finds a very difficult puzzle in a very tense moment, he/she can get too stressed and lose immersion and flow. Instead, if a player finds that difficult puzzle in a low-tension moment, he/she can concentrate better on it and avoid losing flow. Also, easy enigmas located in tense moments avoid losing flow as the player does not get too stressed.

After the idea of the increasing difficulty of puzzles was clear, the next step was to choose which room would hold each type of puzzle. The room path responds to the *linear* one, where the puzzles are done in order and the reward of one puzzle is the beginning of the next one. This type of design makes it easier for the owners to design the room and for players to solve it, as they go all together and can contribute to the solving process.. So in the end, basically the room path was chosen to be: Hall - Director's office - Kitchen - Classroom - Bedroom - Bathroom - Assembly hall - Basement.

6.2. Narrative

6.2.1. Story

When the design started, the idea was to create a game that, despite being relatively short, had a big narrative impact on the player. The idea of having two different endings depending on the player's decisions was very attractive as well, so it was necessary to reflect on both aspects to develop the narrative structure.

Basically, the story takes place in an abandoned orphanage where the player enters. inside each room there is a tape recorder that a paranormal investigator left when he went to investigate the place, and on each recording he says something new about the investigation and gives the player new context. Also, in the director's office the player is able to know a bit more about that director reading his papers, as he is the other important character of the game: Father Elias.

As a resume, in its last years the orphanage took in a pair of twins called Lucy and Ferdinand who were the incarnation of Lucifer. When the teachers found it out based on their late behaviour and several assassination attempts, they decided to carry out a ritual to lock their souls forever. Father Elias, the director, took part in the ritual too and his soul stays in the orphanage to protect the place from unexpected visitors and to avoid them to know the truth.

Many years later, when the investigator goes there he discovers everything and records it, but is killed by Father Elias before he can escape.However, the player gets to defeat Father Elias in the end and a decision comes to his hand. He/she can choose between killing father Elias (which would make Lucy and Fer's souls break free) or leaving him alive (to avoid the children's souls to escape).

The powerful aspect about the story is that the player thinks through the entire game that Father Elias is actually a villain and a bad person, but in the end he/she discovers that he was just trying to protect the place from the player himself/herself.

6.2.2. Characters

The player is an unknown character that enters the place without any context, so each player can imagine a different reason why they are there and make the story even more personalized. The character does not talk and has no visible gender, so that the player can identify even better with him/her.

The guy on the radio is called Isaac, he is a paranormal investigator that is hired to carry out an investigation on the abandoned orphanage. In the end, he manages to discover the truth about what happened in the place, but he is killed by the principal.

Father Elias was the principal of the school in the past, and his soul has remained as a guardian of the place since then. He is the only visible character of the game and can be seen on a few occasions and, of course, during the final fight. His aspect is based on the spanish celebration “Holy Week”, where people dress up with tall hoods during processions. He was created using the 3D modelling software *Blender*. But before that, the first part was to create the concept of the character in 2D using Photoshop, where Father Elias was sketched on his front and side views. Then, the image was put inside *Blender* to start modelling it (*see Figure 17*).

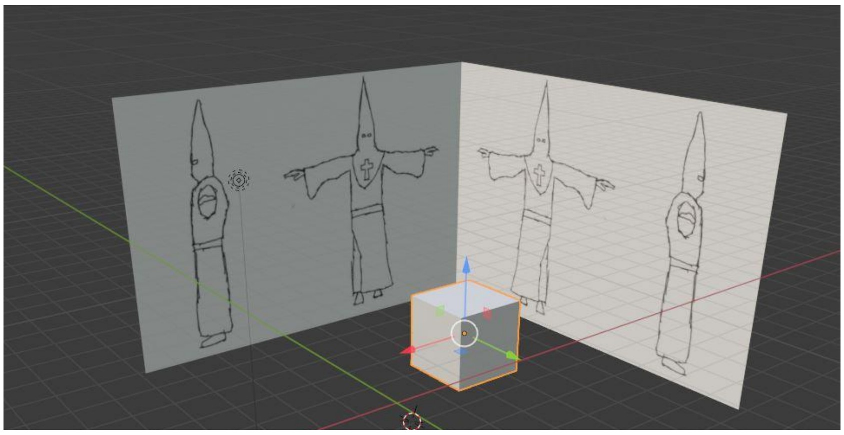


Figure 17. Father Elias' sketch concept.

When it comes to modelling, the designer has to avoid creating a mesh with too many triangles because that could affect the game's frames when the character is on screen. This means that it could be needed to move too many triangles in each second, so the designer has to optimize as much as he/she can the amount of modifications he/she does on the model. Taking this in account and considering that Father Elias would appear on low-lighted scenes, it did not need too much detail to also make it easier to animate (*see Figure 18*).

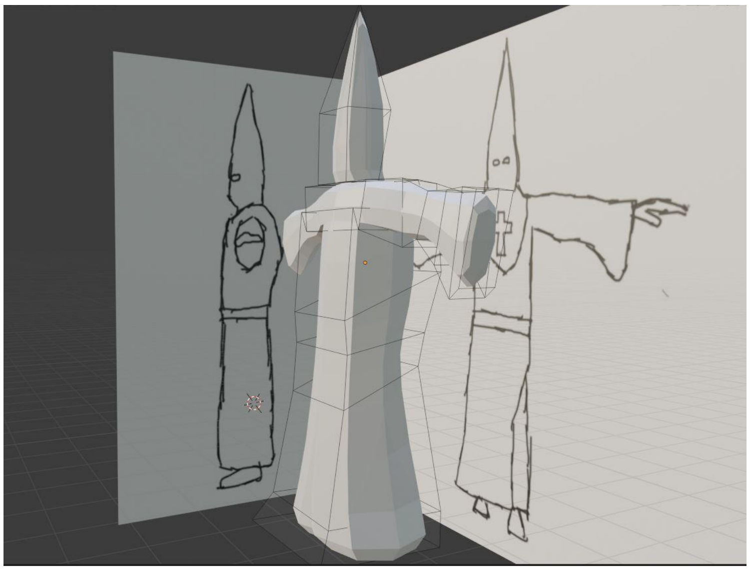


Figure 18. Father Elias' first part of his modelling process.

After having the first part of the skeleton, the next thing was to add him extra elements to characterize him and give him some personality. At last, the only thing left was to apply textures and colour to his clothes (*see Figure 19*).

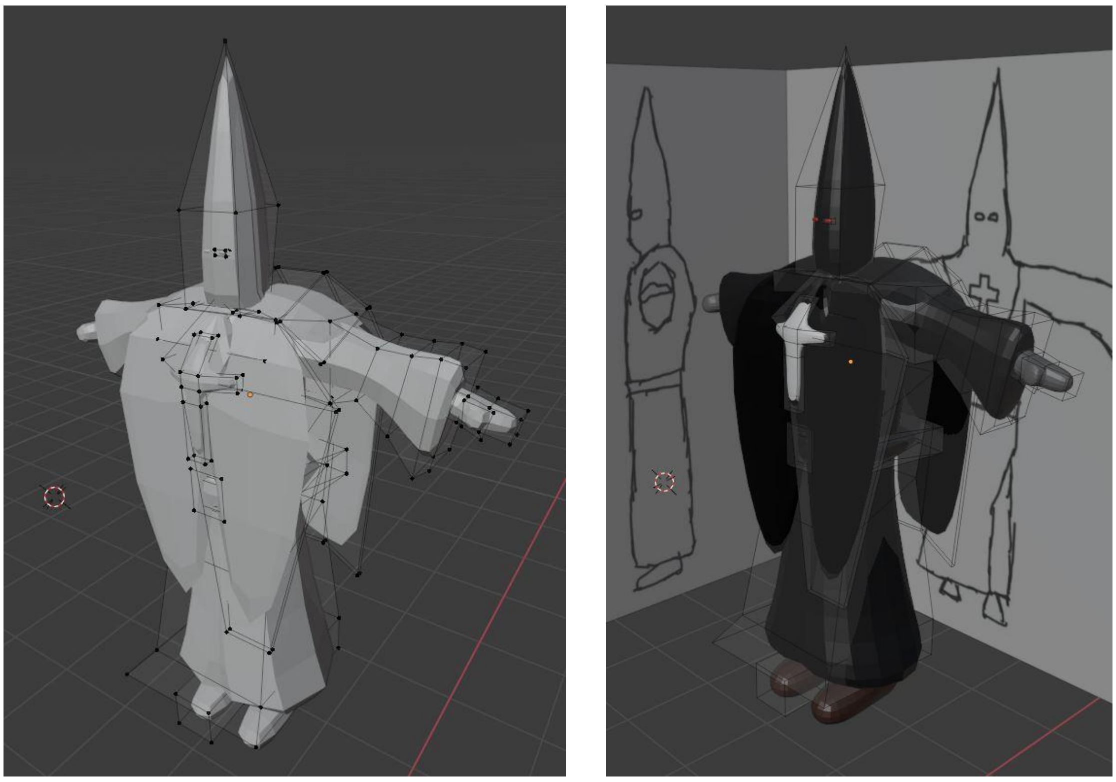


Figure 19. Father Elias without (left) and with colour and textures (right).

Once imported in Unity, for the animation process he was first given a skeleton of head, arms, body and legs and then each animation was done by modifying each part of the skeleton in the indicated moment (*see Figure 20*).

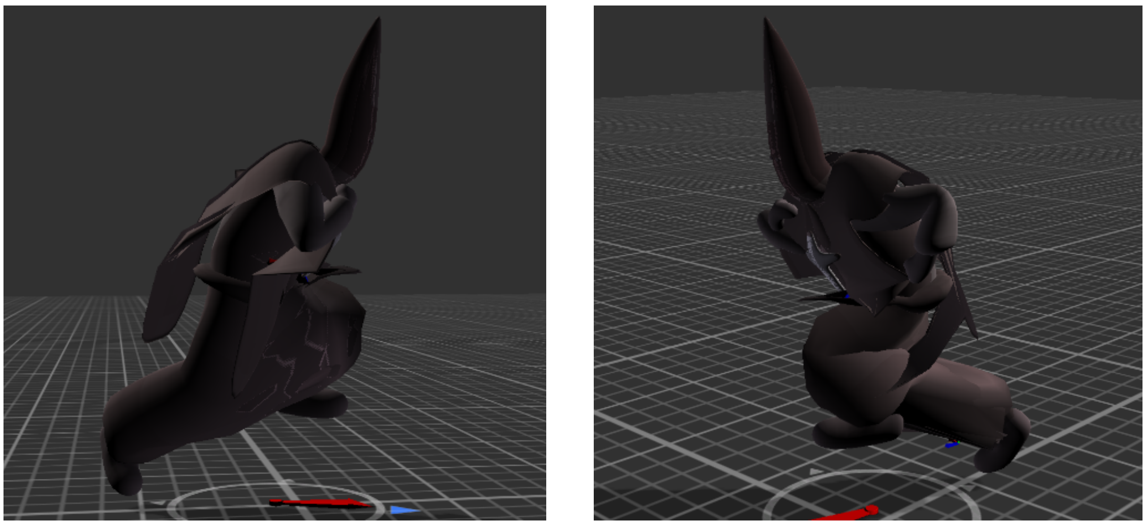
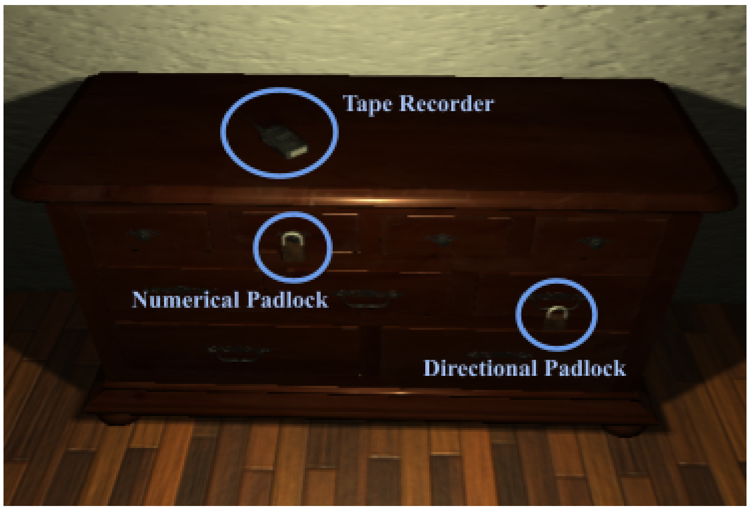


Figure 20. Examples of frames from Father Elias' running animation.

6.3. Puzzle Design

Every room contains at least one puzzle except for the main hall, which only contains the key of the first room. After picking it up, the player will have to solve the following puzzles:

- Puzzle 1: It is located in the director's office, and it relates the room's radio, one sheet of paper and a numeric padlock (*see Figure 21*). As the radio is located above the drawer's padlock, it can give the player the first idea that they can be related. Basically, in the radio, Isaac mentions that the birth date of the director's daughter is important, and in the text from the sheet of paper it can be guessed that information doing simple mathematical calculations. The only "difficult" part of the enigma is to know that february is only 28 days long in non-leap years. Once the day, the month and the year of the kid's birth date is discovered and it's put in the padlock, the drawer opens showing crucial information for puzzle's 2 resolution.



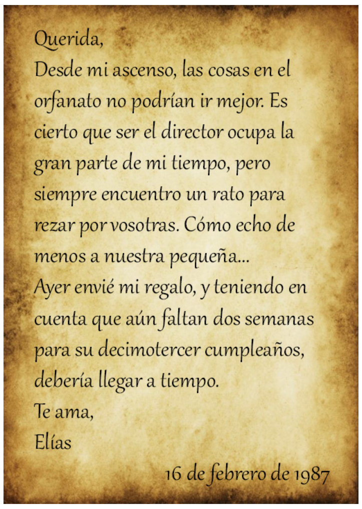
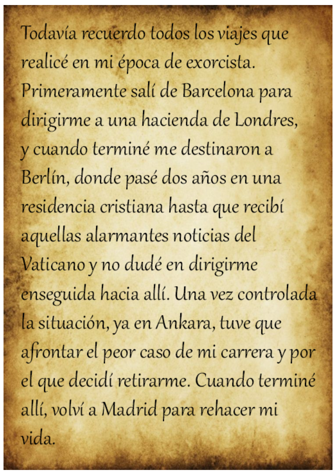

Querida,
Desde mi ascenso, las cosas en el orfanato no podrían ir mejor. Es cierto que ser el director ocupa la gran parte de mi tiempo, pero siempre encuentro un rato para rezar por vosotras. Cómo echo de menos a nuestra pequeña...
Ayer envié mi regalo, y teniendo en cuenta que aún faltan dos semanas para su decimotercer cumpleaños, debería llegar a tiempo.
Te ama,
Elías
16 de febrero de 1987

Figure 21. Puzzle 1's assets to take into account and puzzle 2's padlock.

- Puzzle 2: This puzzle is also located in the director's office. Once puzzle 1 is solved, the drawer reveals a sheet of paper that describes an european trip that the director took when he was young, and describes it saying the name of every city he went to in order. So, if the player looks around the room, he/she will find a map of Europe with all those cities and will have to find a solution for a directional padlock (see *Figure 21*).

If the player follows the director's trip looking at the map, it will give him/her the answer (for example, if the director went first from Barcelona to London, the first direction of the directional padlock would be UP). The "difficult" part of this enigma is that one of the destinations is The Vatican, but the map only shows Rome (so the player must know that The Vatican is in Rome).

In the improbable case that the player does not know where The Vatican is, he/she can know it by guessing the only possible city where the director went from Berlin and then to Ankara using up-down-left-right directions (which only can be Rome in the map). Solving the directional padlock will give the player the kitchen's door's key (*see Figure 22*).


Todavía recuerdo todos los viajes que realicé en mi época de exorcista. Primeramente salí de Barcelona para dirigirme a una hacienda de Londres, y cuando terminé me destinaron a Berlín, donde pasé dos años en una residencia cristiana hasta que recibí aquellas alarmantes noticias del Vaticano y no dudé en dirigirme enseguida hacia allí. Una vez controlada la situación, ya en Ankara, tuve que afrontar el peor caso de mi carrera y por el que decidí retirarme. Cuando terminé allí, volví a Madrid para rehacer mi vida.

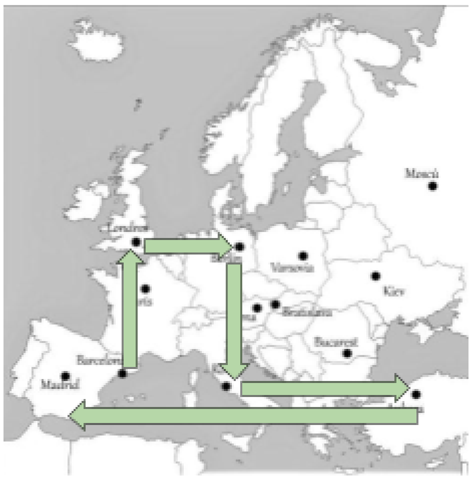
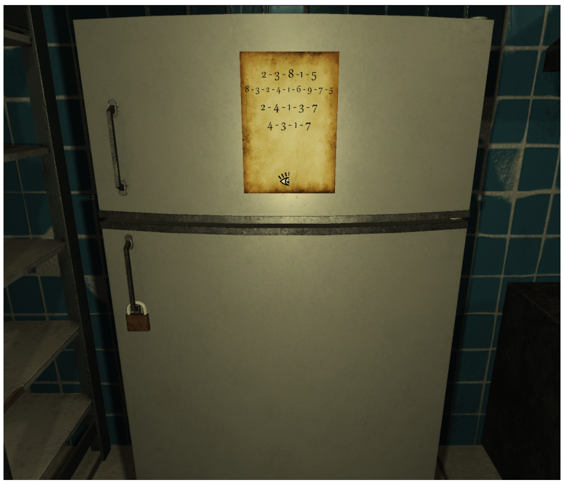


Figure 22. With the note, the player can guess the numerical combination.

- <u>Puzzle 3:</u> This is the only puzzle located in the kitchen, because it is longer than the previous two and involves the entire room. Inside the dining room, there are nine tables forming three rows of three columns each and, of course, each table is numbered with a number between one and nine. In the fridge, the player can find a note with four lines of numbers and a blocked numeric padlock. This note is actually incomplete, as the half eye in the bottom middle shows, and the other note can be found looking around the room behind a radiator, and it's got the other half of the eye (*see Figure 23*).


2-3-8-1-5
8-3-2-4-1-6-9-7-5
2-4-1-3-7
4-3-1-7

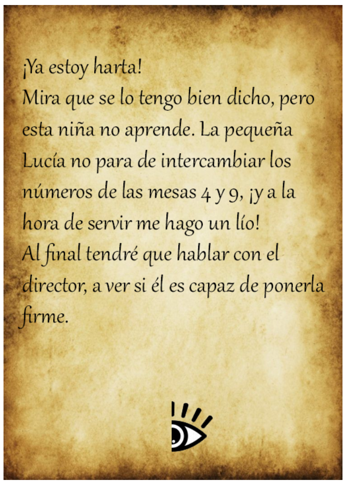

¡Ya estoy harta!
Mira que se lo tengo bien dicho, pero esta niña no aprende. La pequeña Lucía no para de intercambiar los números de las mesas 4 y 9, ¡y a la hora de servir me hago un lío!
Al final tendré que hablar con el director, a ver si él es capaz de ponerla firme.

Figure 23. Kitchen's puzzle information needed.

The puzzle consists in going from one table to another following each line of numbers, which would at last give a number (*see Figure 24*). In this puzzle, if the player does not pay attention to the half eye in the middle when looking at the note in the fridge, he/she can think that he/she can already solve the puzzle, but it would be a waste of time because the second part of the note says that two of the numbers are swapped. Once the puzzle is solved and the padlock is unlocked, the fridge would open showing a plunger inside.

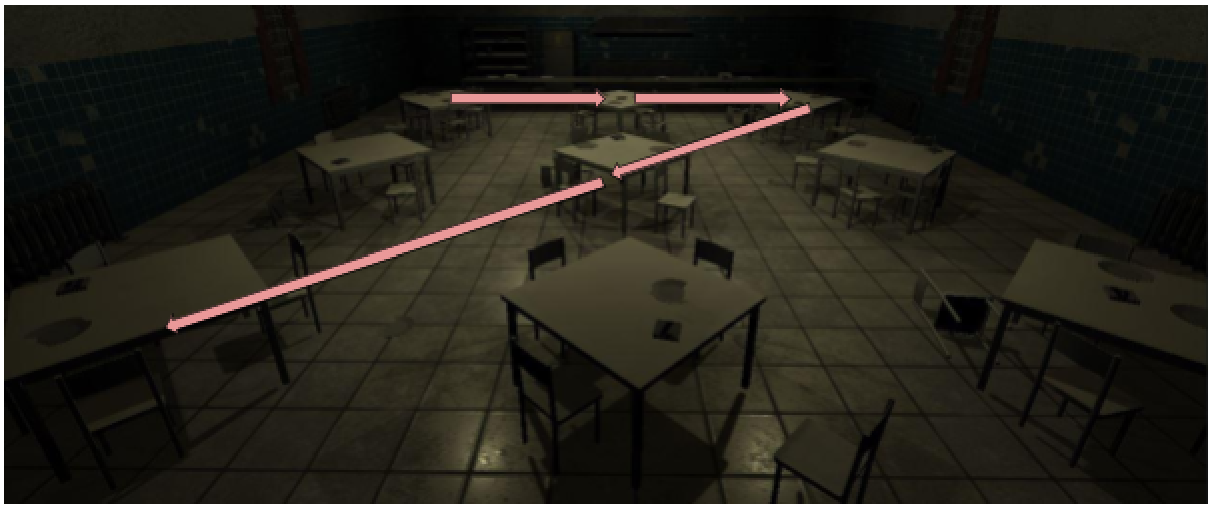


Figure 24. For example, in the first line in the note it says "2 - 3 - 8 - 1 - 5", so if the player follows that path it will give him/her the number 7.

- <u>Puzzle 4:</u> This "puzzle" is more like a simple task, once the player gets the plunger from the fridge the only thing he/she has to do is use it in the sink to clear the water. Then, at the bottom of the sink, he/she can get the classroom's key (*see Figure 25*).

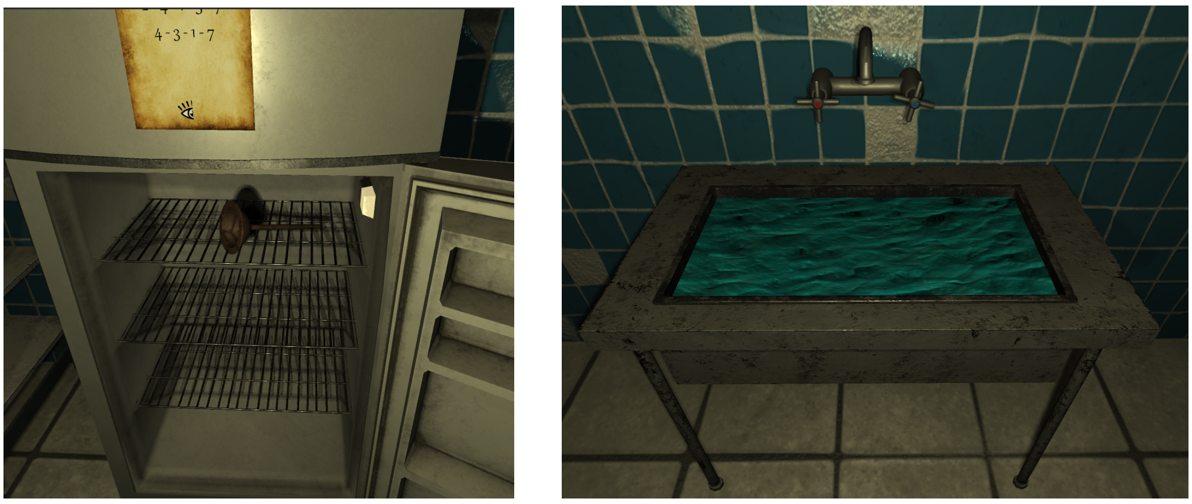


Figure 25. Using the plunger in the sink will give the player the classroom's key.

- <u>Puzzle 5:</u> Inside the classroom, in one of the students' tables, the player can find a "magic triangle". A magic triangle is a triangle that has numbers on each side of it, and the sum of the numbers of each side must be the same as the other sides. Also, used numbers cannot be repeated. Each blank space in the triangle is coloured, and the numerical padlock of one of the lockers is blocked with a numerical coloured padlock, which gives the player the clue that they are connected. If the player manages to discover every blank number, he/she will have the answer to it (*see Figure 26*). When the locker is opened, the player will find a baseball bat inside it.

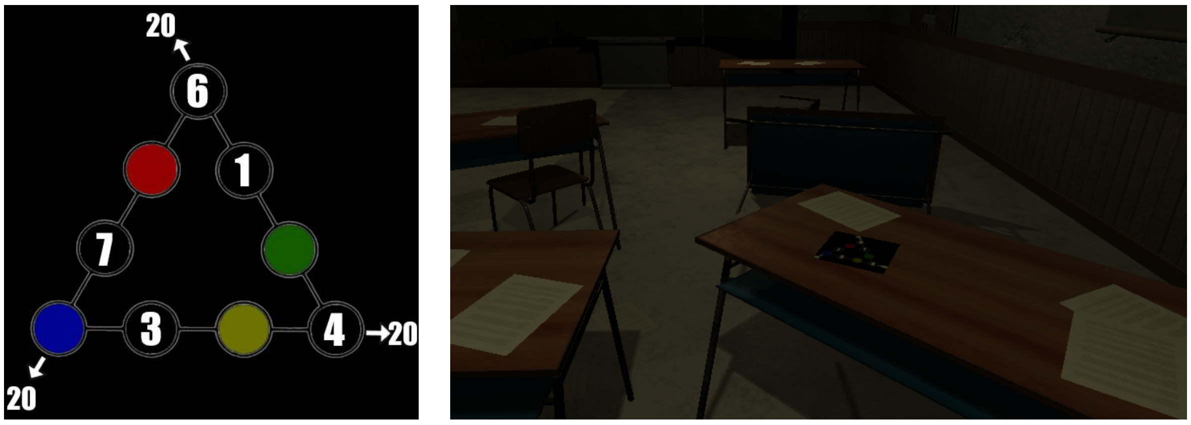


Figure 26. Each side of the magic triangle must sum 20.

- Puzzle 6: This is another action puzzle similar to puzzle 4, in this case the player has to use the bat to break a glass box that contains the bedroom's key (*see Figure 27*).

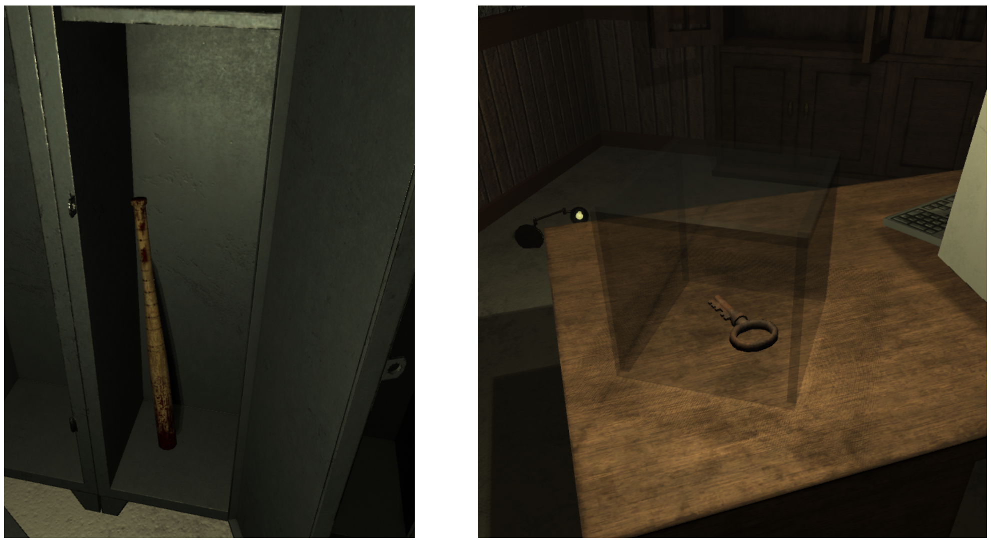


Figure 27. The glass box must be broken using the bat.

- Puzzle 7: With the second floor the game takes one step up in the tense atmosphere, so the puzzles stop requiring a huge capacity of concentration and mathematical puzzles disappear. In this case, once the player enters the bedroom he/she will see many texts in the walls that say what each orphanage's kid used to do with his/her toy. The player needs to know which toy belonged to each kid to solve an animal padlock. This padlock contains a combination of six animals that the player must place correctly under every kid's name's initial letter.

There are three toys inside the bedroom, but the other three are inside the three previous rooms, one in each. If the player remembers which toy was in each room he/she will not need to go downstairs to check, but will have to do it otherwise. Once the padlock is unblocked, inside the drawer the player will find the bathroom's key.

- Puzzle 8: This is the first puzzle of the bathroom. When the player enters the room, the light will start flickering following a pattern of a certain number of blinks. When the player realizes, the only thing he/she will have to do is count the number of blinks and put them in the numerical padlock of one of the lockers. Inside of it, the player will find a pistol with a sign that tells him/her how to shoot (*see Figure 28*).

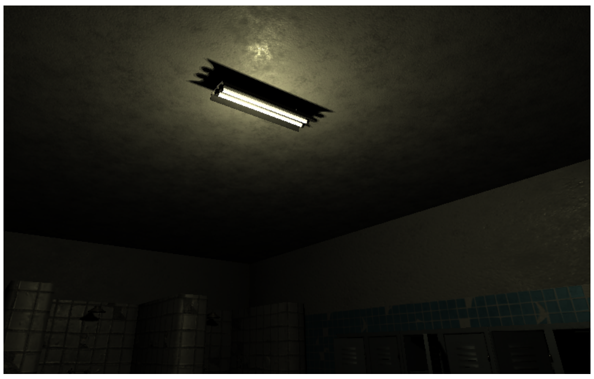
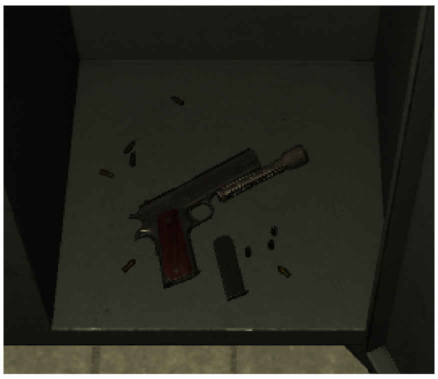



Figure 28. The lighting pattern is the solution for the numeric padlock. It flickers four times, then five, then seven and lastly three, and then repeats the process.

- Puzzle 9: With the pistol in hand, the player will have to shoot at some padlocks that block the toilet's doors. Behind one of the four doors, inside the toilet, the player can find the assembly hall's key (*see* Figure 29).

Figure 29. Shooting at the padlocks will open the toilet's doors.

- Puzzle 10: The last puzzle of the game is located in the assembly hall, and it is divided in two parts but work as the same puzzle. The first part consists in finding two items around the room that the player needs to turn on the projector (a battery and the film). Once he/she finds them, the projector starts playing a film that contains four numbers that are displayed for a short time, so the player has to be focused in watching and remembering the numbers in order. Once the film ends, a magic drawer will come out the wall with a numeric padlock where the player will have to enter the numbers he/she has seen. If the combination is correct, the drawer will give the player the basement's key (*see Figure 30*).

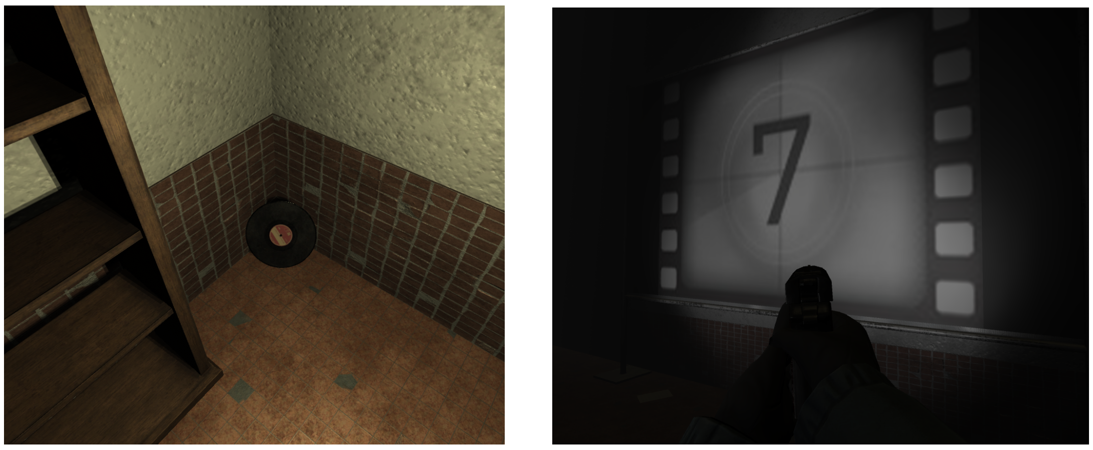


Figure 30. This puzzle puts in practice the player's observational skills.

6.4. Objects and Padlocks

Unexpected Visitor mixes 3D and 2D modes depending on the game state. Some objects from puzzles can be seen closer clicking on them, which shows them in their 2D mode. This happens, for example, with every padlock. When one of them is clicked in their 3D form, the 2D form of them appears on the screen to be manipulated. To do this, the 3D object has to detect the cursor (which is locked in the middle of the screen and is represented by a red hand), show it to let the player know that that object can be manipulated and once it is clicked the cursor stops being locked (*see Figure 31*).

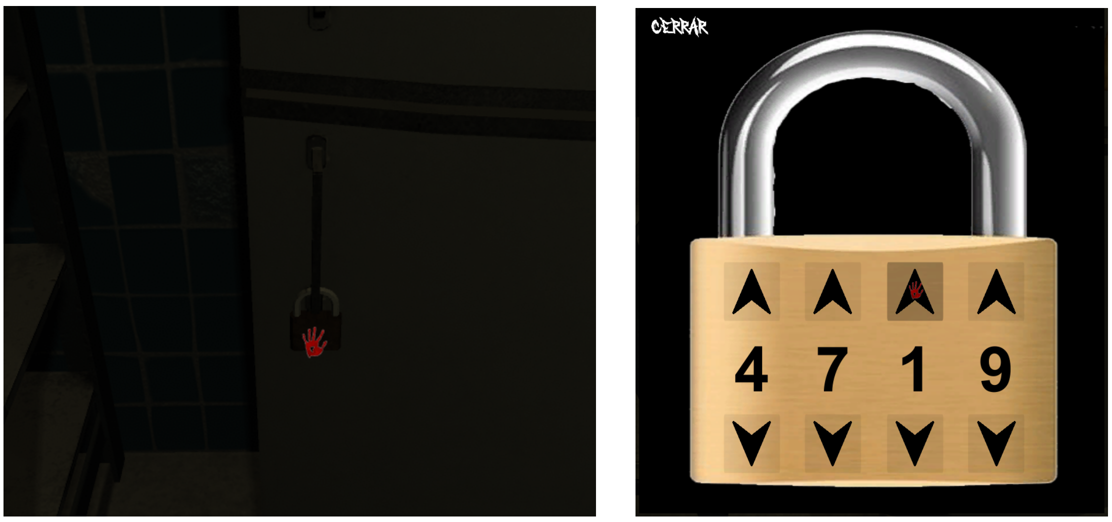


Figure 31. Clicking in a padlock will bring its 2D form to the screen.

This type of clickable objects include buttons in their 2D form. All of them, of course, share the “Close” button to go back to the 3D mode, but padlocks also have buttons to increment or decrement each number in the numeric padlocks and on the numerical and animal padlock contain a rotate button to change the directional arrow or the animal respectively (*see Figure 32*).

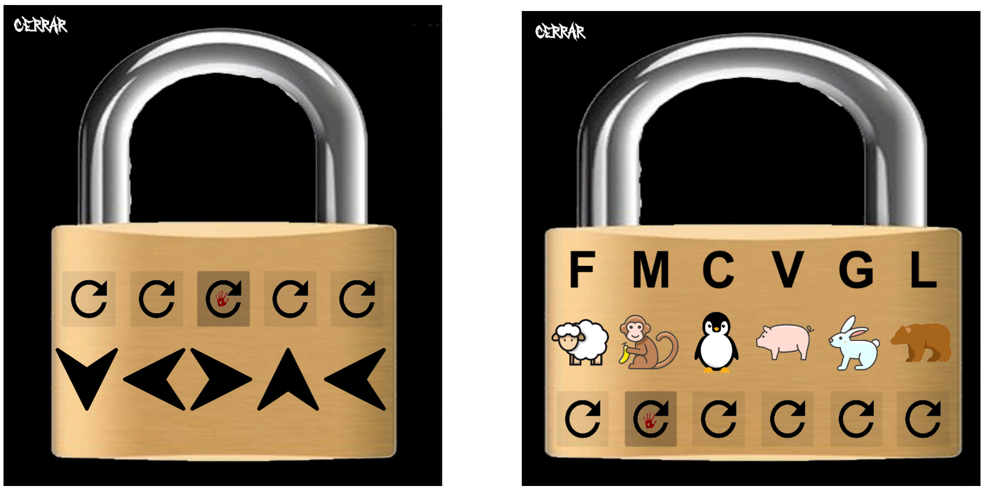


Figure 32. Directional and animal padlocks use rotative buttons.

Every padlock inherits a primary padlock function that constantly gets the value of every number of the padlock (which are strings because they are written in text boxes), then puts them all together on a string and then turns that string into an integer. That integer is compared with the correct padlock’s combination and if they are the same, the game goes back to the 3D mode disabling the UI, locking again the cursor, setting the timeScale back to 1 (which was in 0, which means that the time stops. If its value is 1 it means that time goes back to normal) and disabling the solved padlock from the scene.

Every object inherits a function that activates the cursor to let the player know whether it is clickable. If the player is close enough to the object in both the “x” and the “z” axis, the cursor will show up if the player is also pointing to that object. If the player stops pointing at it, the cursor will disappear.

6.5. Giving Life to the Level Design

So, once the puzzle design was over, the asset searching part began looking in hundreds of different 3D models websites to find the perfect fit for the level design. This process was long and boring, but the chosen ones fitted perfectly in the game concept.

Once all the assets were imported to the unity project, the level design began giving life to scenes concept with the objects needed for each puzzle and some decoration to give life to the place. This process was also very long and tedious, but the combination of all those individual pieces of art turned out to be *Unexpected Visitor*’s orphanage (*see Figure 33*).

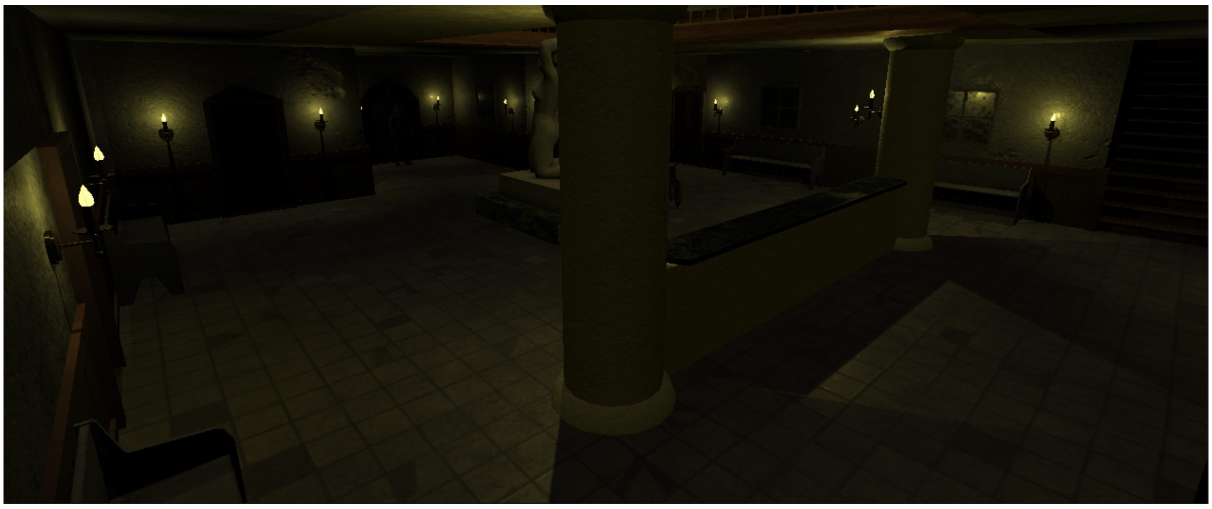


Figure 33.1. First floor's main hall.

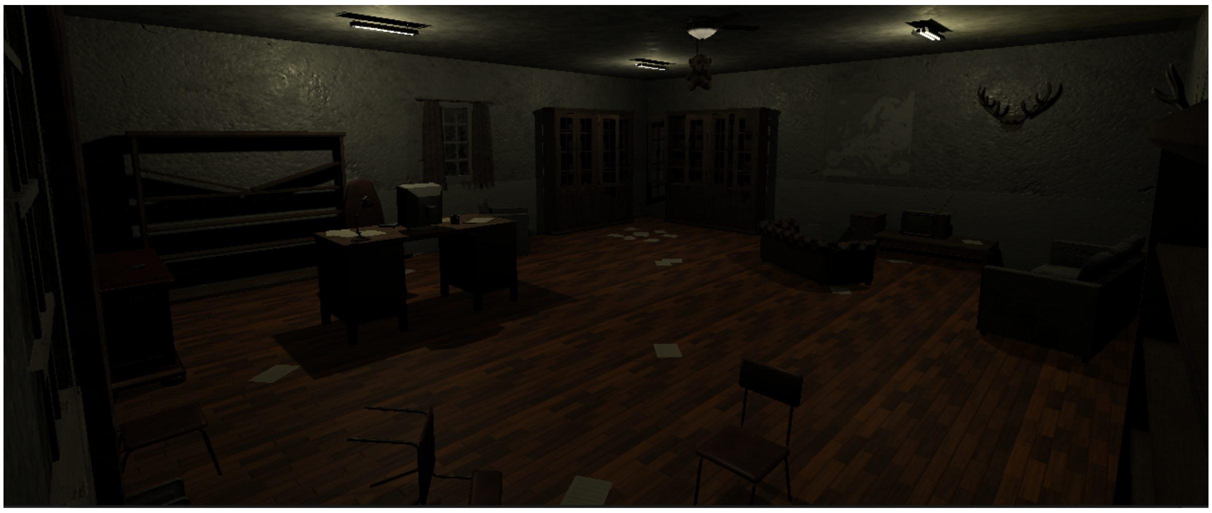


Figure 33.2. Director's office.

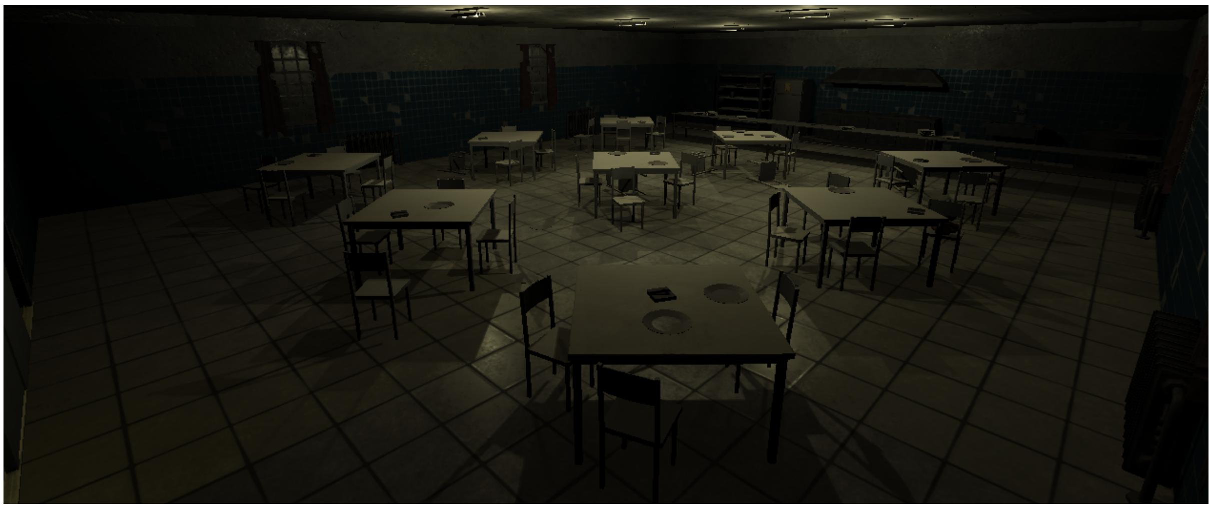


Figure 33.3. Dining room / kitchen.

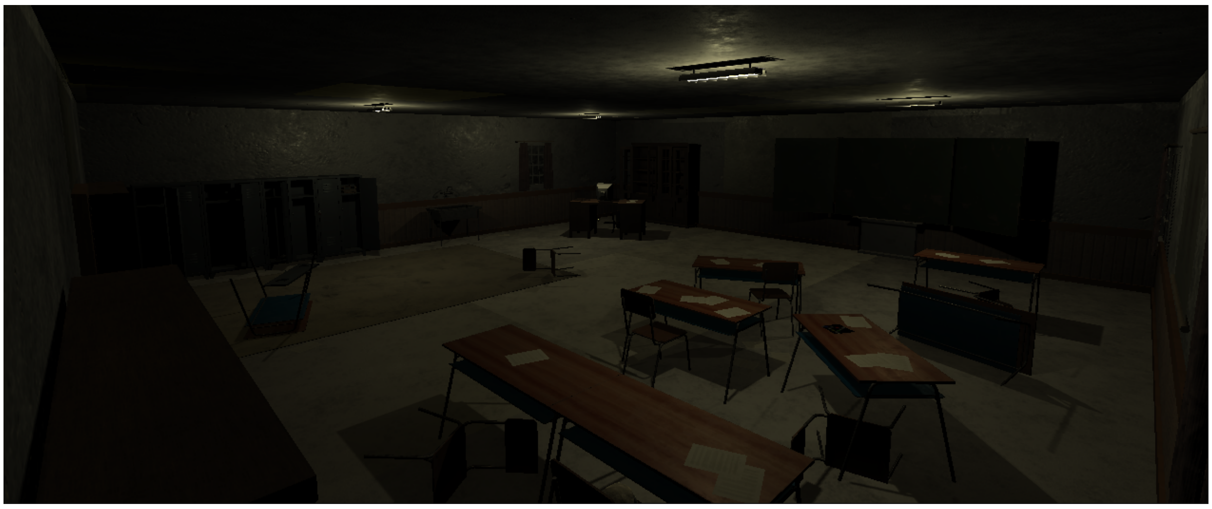


Figure 33.4. Classroom.

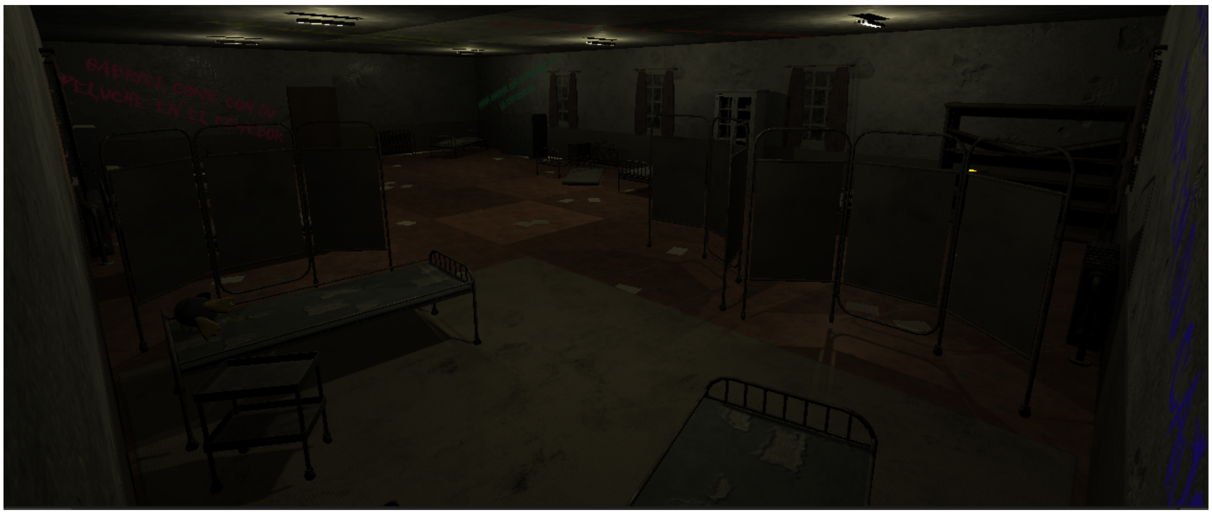


Figure 33.5. Bedroom

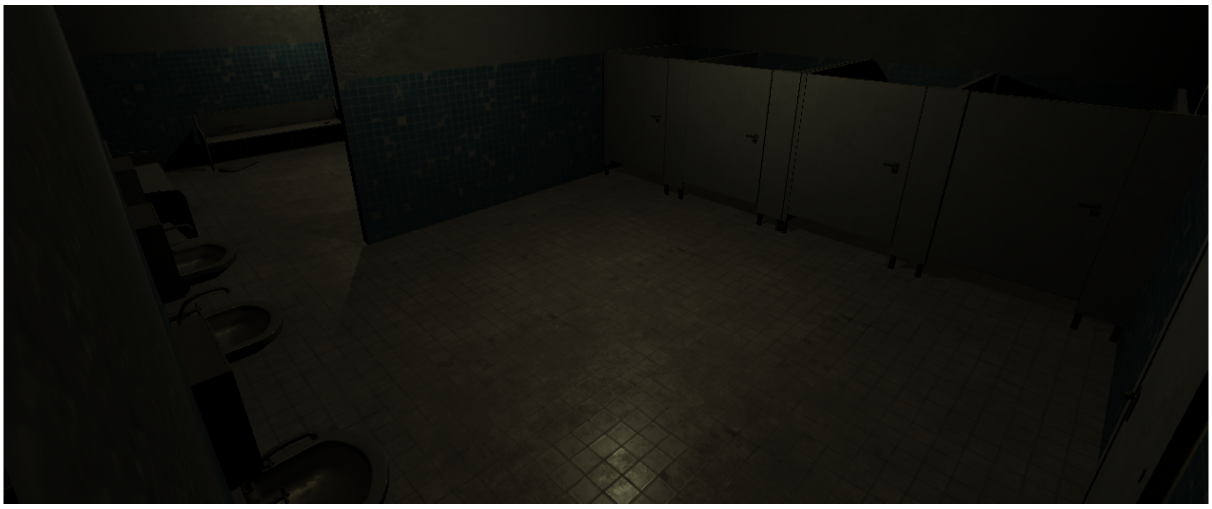


Figure 33.6. Toilets part of the bathroom

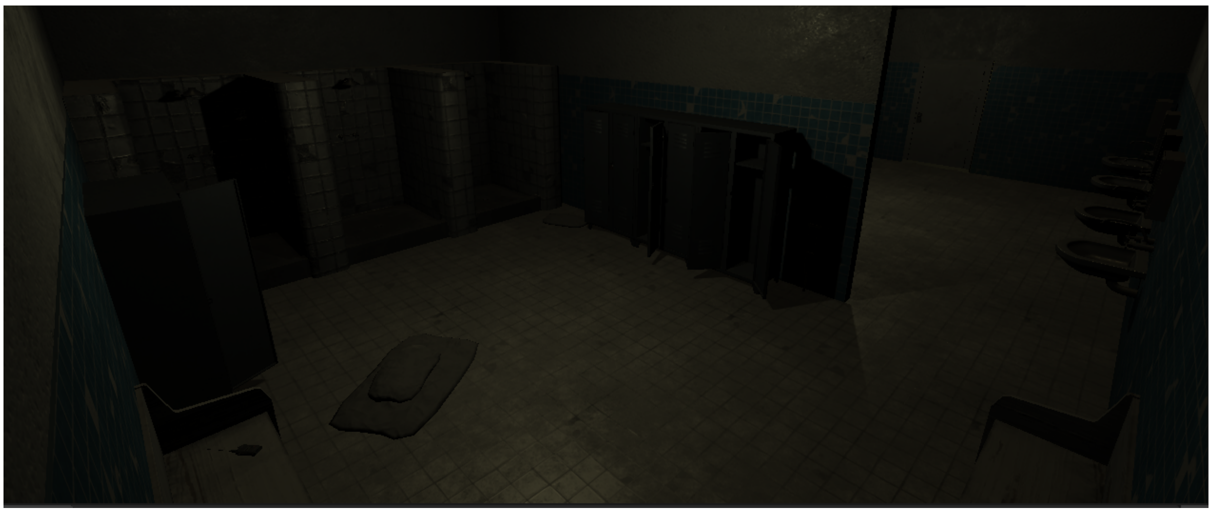


Figure 33.7. Showers part of the bathroom.

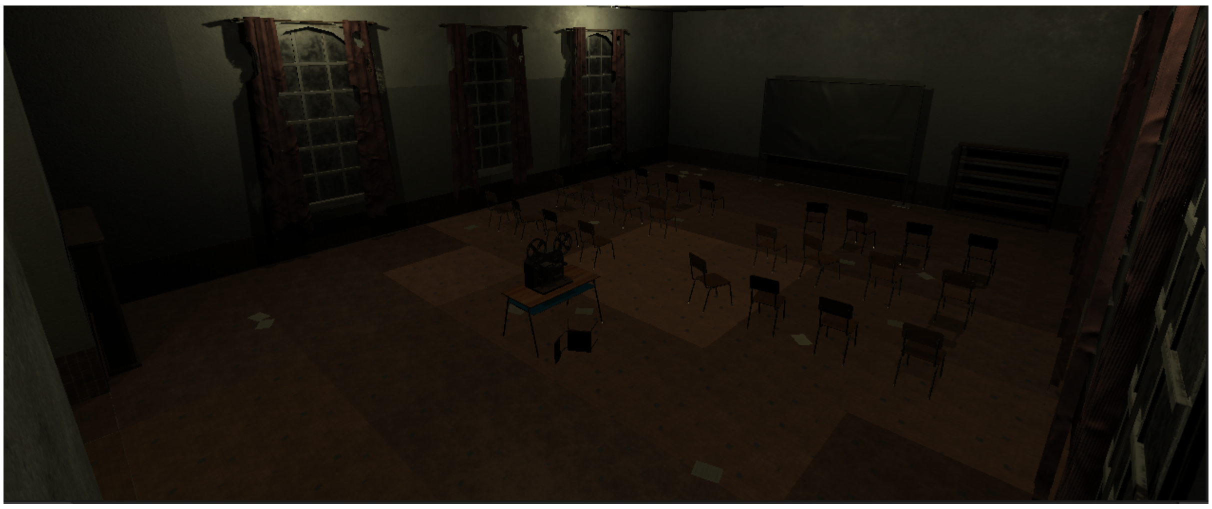


Figure 33.8. Assembly hall.

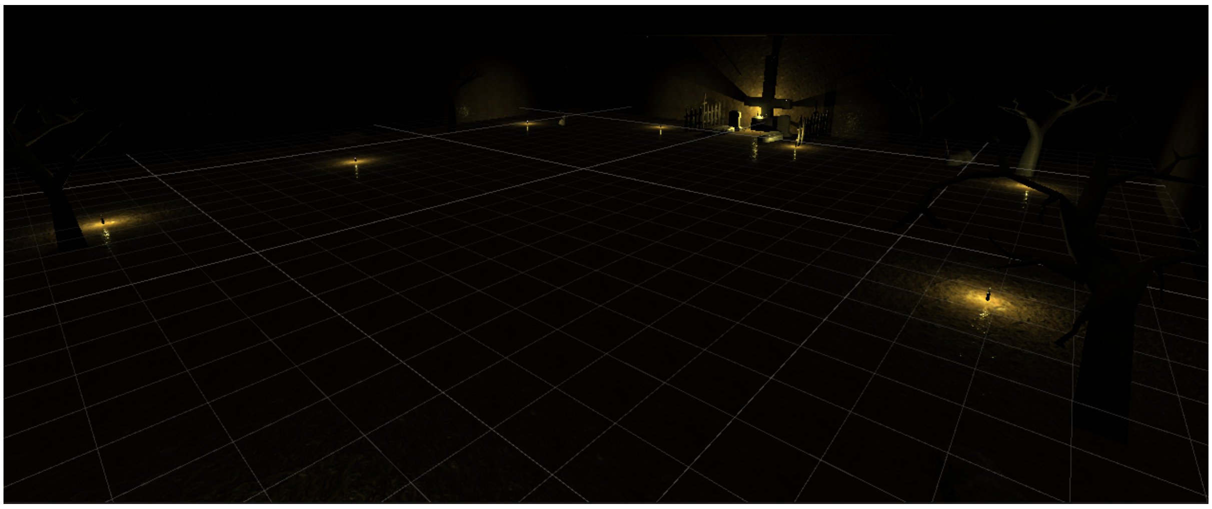


Figure 33.9. Basement / Cemetery.

6.6. Immersion and Tense Atmosphere

As it has been said before, immersion is a crucial part of this game. With the environment created before much of it has been achieved, but both the immersion and the tension are constant and do not vary. So, in order to change that, some things needed to be added to the project.

6.6.1. Lighting

In horror games, lighting is an essential part of the immersion. In this case, the hall lights had to make the player feel like he/she is inside an abandoned creepy place at first sight, so they needed to have an irregular yellow colour. Also, the intensity of each one's light needed to be different from the others to increase that feeling.

In case of the rest of the rooms' lighting, here it needed to be a bit higher. This way, the players can look around and solve the enigmas with enough light to avoid making them feel like they need more light to see a certain puzzle. This light is whiter than the hall's one and lights every room (*see Figure 34*).

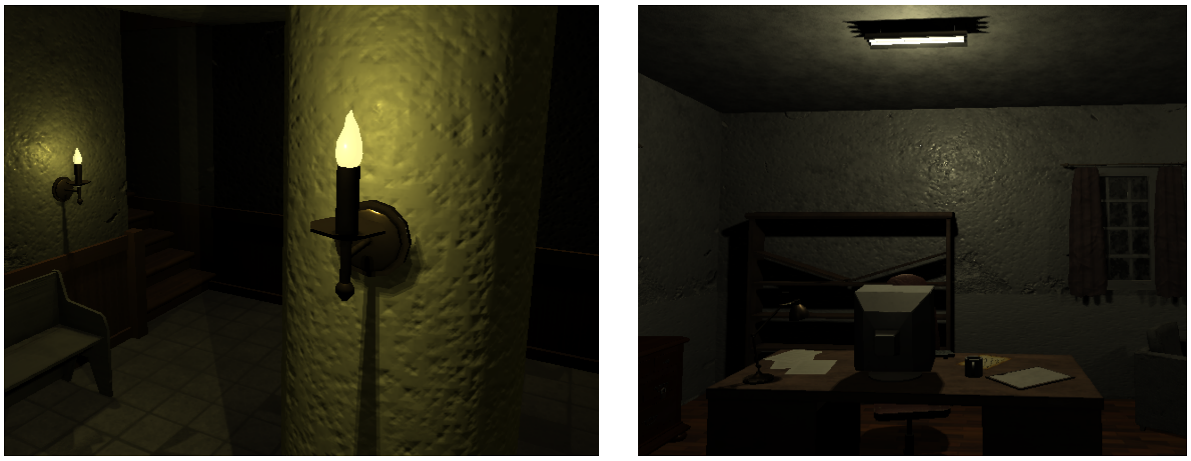


Figure 34. The lights in the hall (left) and the lights in rooms (right) are used for different reasons.

When the player enters the bedroom, the lights from the entire orphanage go off. This increases the tension level a bit and lets the player know that if he/she wants more light out there he/she needs to get it from somewhere else. This happens because in that exact room there is a light that the player must take to see outside the bedroom and look for the children's toys. This handlight emits a low yellow light around it when the player moves which helps him to enlighten the path a bit (*see Figure 35*). Also, the candles from the basement emit similar short light that illuminates the nearest objects to them and casts shadows on the enemy during the fight to create more mystery and increase the tension.

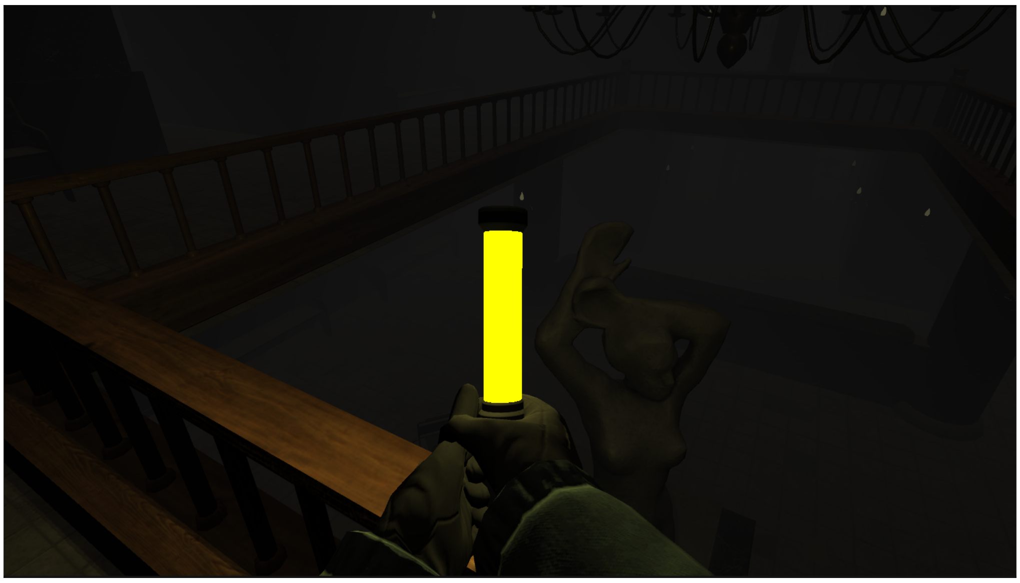


Figure 35. The handlight emits a bit of light around it.

Later, when the player takes the gunshot, it includes a flashlight that emits a strong white light ray that illuminates the path ahead of the player. But his/her joy does not get too far because it stops emitting light shortly after getting it, leaving the player with no light at all and incrementing with it the tension level.

The projector from the assembly hall uses the same illumination type as the flashlight, but with longer range and with some particles at the beginning of the light ray seeming like air dust (*see Figure 36*).

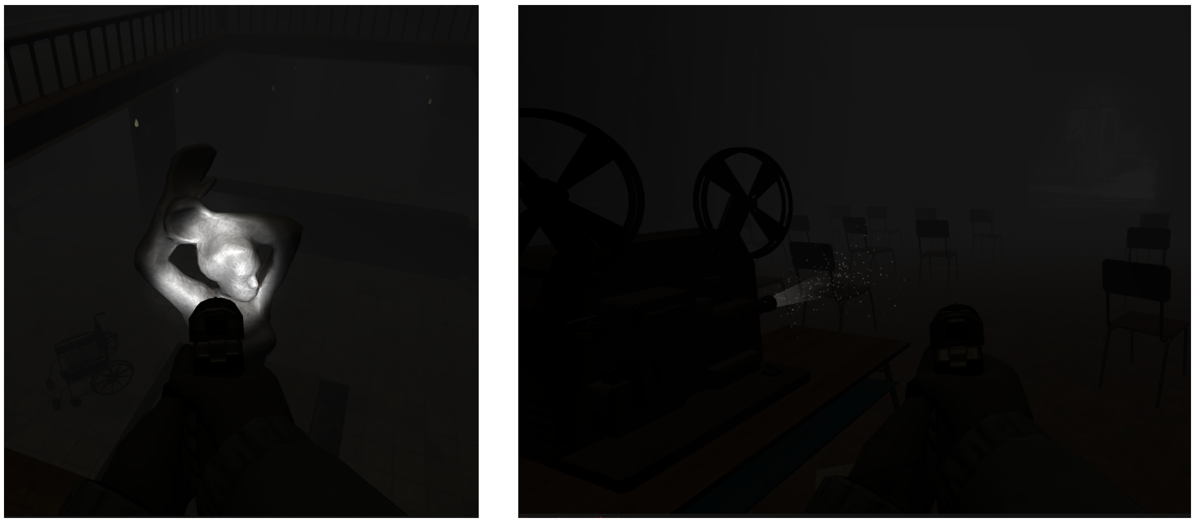


Figure 36. The flashlight and the projector emit a lot of light to the front.

At last, another thing that has a lot to do with the lighting setup is the fog, which gives the feeling of the presence of dust in the air and has a huge role in the immersive atmosphere (*see Figure 37*).

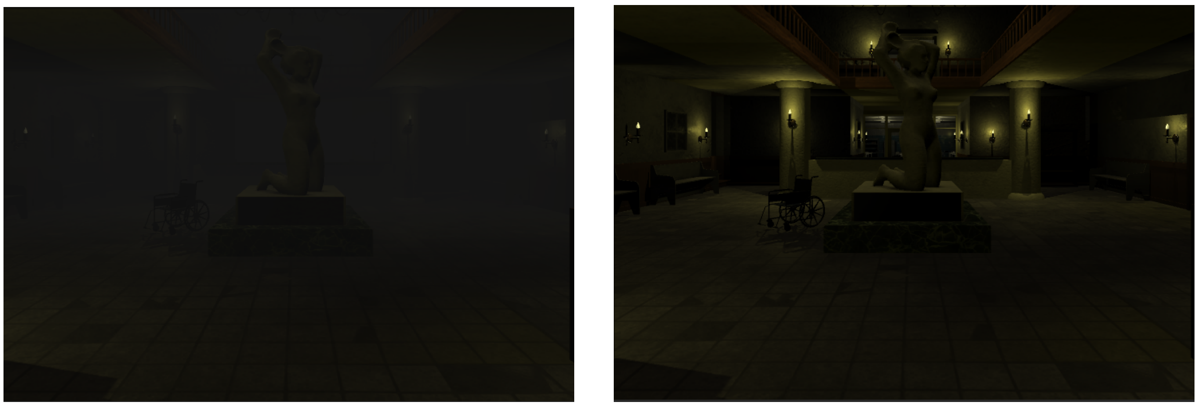


Figure 37. Scene comparison with fog (left) and without fog (right).

6.6.2. Progressive Disturbing Events

As it has been said before, the tension level has to increase from low to high as the player advances through the rooms. Therefore, at the director's office there is no disturbing event to happen at any time. It is in the kitchen where once the player gets the plunger a small event happens, which is that one of the table's chairs that was on the floor moves to the top of the table itself. This is not something that would disturb a player too much, but it sets the beginning of a series of events that will tell the player that he/she is not alone in the orphanage.

Then, when the player solves the magic triangle puzzle and grabs the bat, some students' tables rotate to look to the player's position along with the toy that was in one of the chairs and looks straight at the player. Also, in the blackboard the player can see written in red (seeming like blood) the sentence "You're going to die", which sets a point of no return to the player's tension as he/she realizes that he/she is not alone indeed.

After that, once the player enters the bedroom and the lights from the outside go off, if the player goes downstairs to see which toys were located in each room, the player will meet Father Elias for the first time, as he/she will get scared by him when going downstairs. If by any chance the player remembers all the toys and does not go downstairs, this event will happen any other time he/she goes downstairs instead (*see Figure 38*).

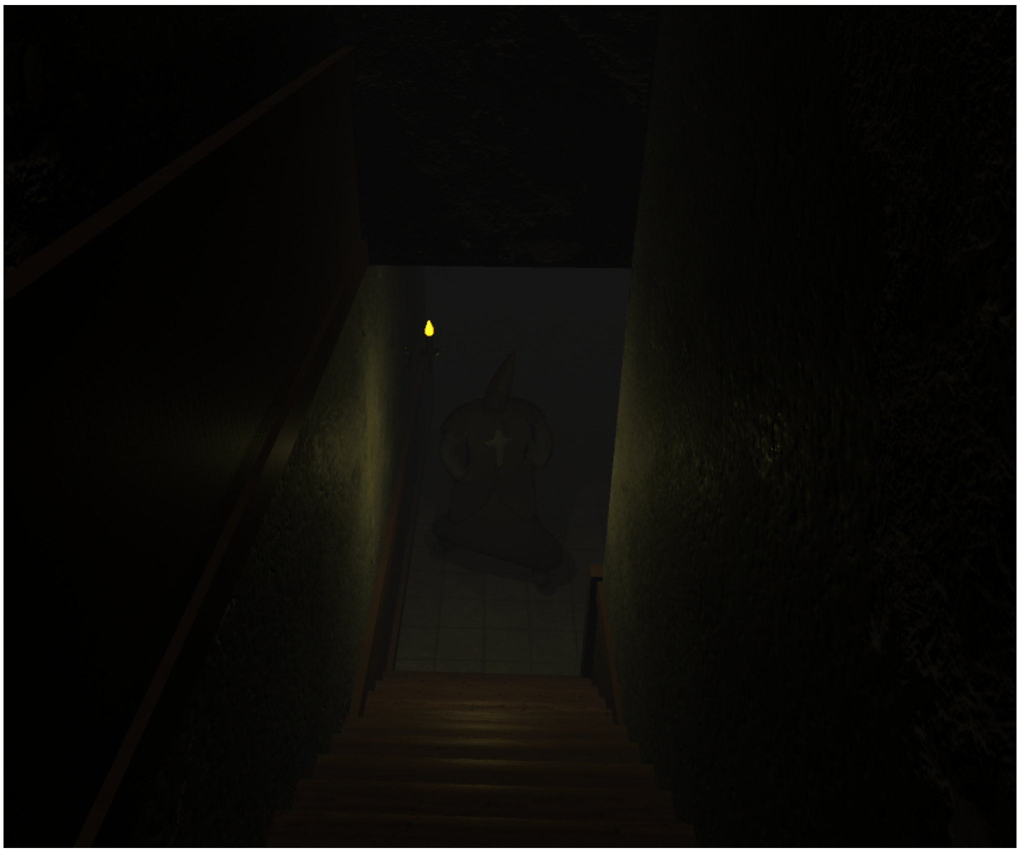


Figure 38. Father Elias' first appearance.

Later, once the player solves the bedroom's padlock, every toy will appear behind the player looking straight at him/her. If the player has already seen Father Elias this will not be too disturbing, but otherwise it keeps increasing the tension level progressively.
The last disturbing event is made by Father Elias again, who appears behind the player when he/she solves the assembly hall's padlock and scares him/her when he/she turns around (*see Figure 39*).

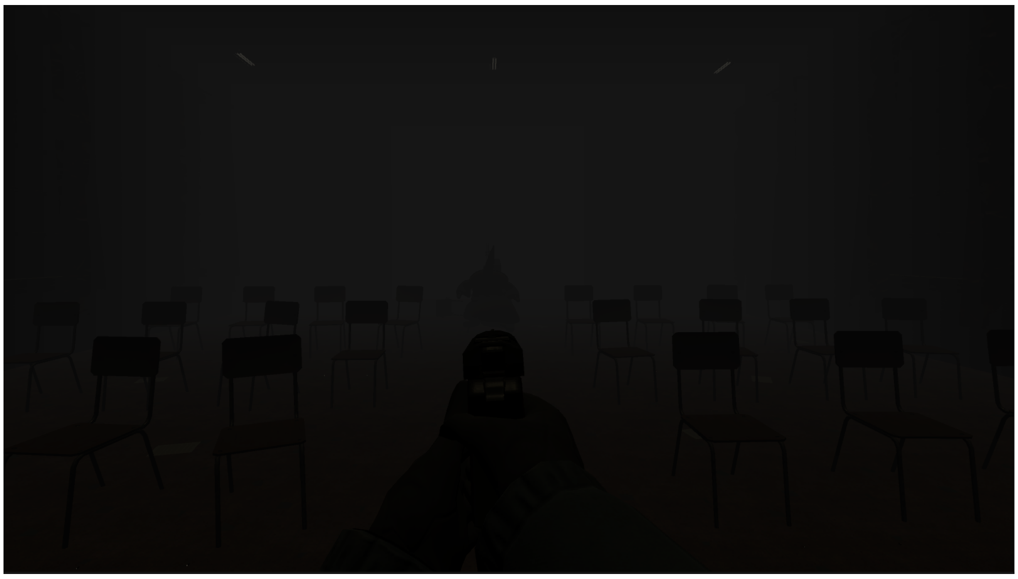


Figure 39. Father Elias' second appearance (or first for some players).

All these previous events help setting the perfect tension level for the player to lastly face the final fight against Father Elias at the peak of his/her immersive experience.

6.6.3. Music and Sound Effects

Ambience music and sound effects are also important when it comes to horror-themed video games. In this case, the background music is soft and calm but also a bit terrifying, which is perfect because it does not disturb or bother the player to concentrate on puzzles but also helps create the perfect tense atmosphere desired. On the contrary, when the final fight begins the music changes for a loud rock but also terrifying music that elevates to the maximum the player's tense feeling. When the fight ends, the music goes back to the previous background music.

There are a total of 41 sound effects inside the game. Literally every event of the game has its own sound effect to maximize the immersive effect. The radio tapes have been recorded and the voice has been modified to give it a real radio effect, so it sounds like a real recording. Father Elias' voice has also been recorded and modified to make it look like he is talking from another dimension.

6.7. Relevant Code Mentions

Unexpected Visitor includes around 45 different scripts. Although it can seem like a big number, some of them (like for example the different types of padlocks) inherit a big part of the code from a primary main code (in this example would be the general padlock) and add some modifications to specialize the script. This part of the project is going to focus on the most relevant information from the most relevant code fragments.

6.7.1. Player and Camera

The player's main code takes place inside the Update() function as it needs to be constantly checked to guarantee a smooth movement. Basically, this part of the code is focused on moving the player. First, it plays the movement animation or not depending on if the player is moving or not in any "x" or "z" axis. Then it allows the player to move if he/she has pressed any of the horizontal or vertical keys (which basically are the "A" and "D" keys for horizontal and "W" and "S" keys for vertical). After that, it just checks if the player is on the floor and moving to play the walking audio or not. At first, the option of jumping was also implemented in the game, but after some considerations it was removed because it did not have a real utility and players could jump from the second to the first floor.

The code of the camera is also developed in the Update() function because it needs to rotate the player constantly. It basically checks if the mouse is being moved in the "x" and "y" positions and rotates the player in those directions. The interesting thing about this function is that it includes the “Clamp(variable, min value, max value)” method, which constrains rotation to the valid values. This way, when the player looks up the camera does not allow him/her to make a 360 degrees rotation and stays locked up there, to give it more realism because obviously humans cannot rotate their neck more than 90 degrees both up and down.

6.7.2. Bathroom Lighting

The first bathroom puzzle, as commented before, consists on a loop of a light turning on and off a certain amount of times. To do this and try new ways of coding, it was developed using invocations. First, once the player enters the bathroom he/she steps into an invisible trigger that sets this code’s variable “isActivated” to its “true” value, so this Update() function activates and plays a function called “StartLighting()”, which includes and Invoke() method. This method invokes a function in a given value of time. In this case, it invokes the function”SetLightActive()” one second after this method is called.

Then, the SetLightActive() function sets the light active and plays its turn on sound and invokes another function one second after that. This function will turn off the light and check if the number of times that the light has been turned on and off (“take” variable) is equal to the target number of times it has to do that (“targetNumber” variable) or not. If those numbers are different, the first number adds one unit to itself and invokes again the SetLightActive() function.

On the contrary, if those numbers are the same, it checks in which state of the loop it is. The combination of this padlock has four numbers, so if the “targetNumber” variable is equal to one of those numbers apart from the last one it will be changed to the next one and the “take” variable will be set to 0 and invoke the “WaitTimeMiddle()” function which basically invokes back the “SetLightActive()” function but this time in 3 seconds instead of 1, to let the player know that one part of the loop has finished. If the “targetNumber” variable was the last number of the combination, the same thing would happen but instead of waiting 3 seconds it will be 5 seconds and it will also play a different sound that will indicate that the loop has finished and it is restarting. This invoking loop repeats forever until the player solves the padlock.

6.7.3. Final Fight

The final fight of the game involves many events. But with Father Elias it was wanted to go a step further and make him slightly intelligent. That is why it was decided to develop the A* Pathfinding method for Elias’ movement. To do it, the first thing that was needed to do was creating a “Node” class to later create a Grid of nodes. The Node class included a boolean variable to let the grid know if it is a wall or not, a 3D vectorial position and two integer variables for its “x” and “y” position.

Then, the grid is created using an array of nodes the size of the basement room. The size of the nodes must not be too small to avoid lagging problems as they require a lot of the game’s computing speed, but they should not be too big either to let Father Elias have many points to move to. Once the grid is created, each node calculates its neighbour nodes and their position to see to which one of them it will have to move depending on the player's position and adds each of them to a list of neighbours. Each node will have its own list of neighbors at the beginning of the fight already calculated to avoid errors.

After that, the grid will calculate constantly the player's position and Elias' position to draw in a red colour the nodes that form the closest path from one point to the other. It will also write in a yellow colour those nodes present on the grid that had their attribute "isWall" from their class code mentioned before in a "true" state. This means that those yellow nodes cannot be written in a red colour therefore the enemy cannot go through them. Obstacles in the middle of the map will also be considered walls and Elias will avoid them. At last, any other node that is not a wall and is not part of the closest path will be coloured in a white colour (*see Figure 40*).

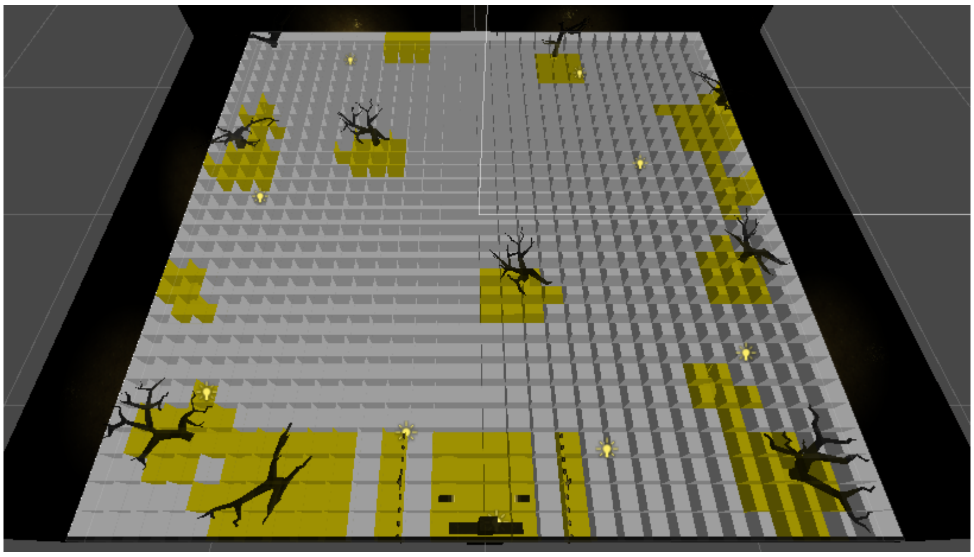


Figure 40. The grid shows in white the free nodes and in yellow the obstacles. Yellow nodes cannot be occupied by the enemy.

The red path mentioned above is calculated in the "Pathfinding" script, where the player gets the player's node and the enemy's node and finds the shortest path between each other. To do it, the script's main function "FindPath()" and starting from the enemy's node, every neighbor of that node is compared to the player's node and the one that is the closest to it is chosen to be the next evaluable node, and each of its neighbor nodes will be analysed then following the same procedure than before, taking into account of course only the white free nodes. Once the last node to analyse is the player's node, the path finding is complete and the enemy can know exactly where to move from each position. This pathfinding is calculated in milliseconds all the time, so every step the player or the enemy makes is reflected in the red path in real time, getting a final intelligent pathfinding as result (*see Figure 41*).

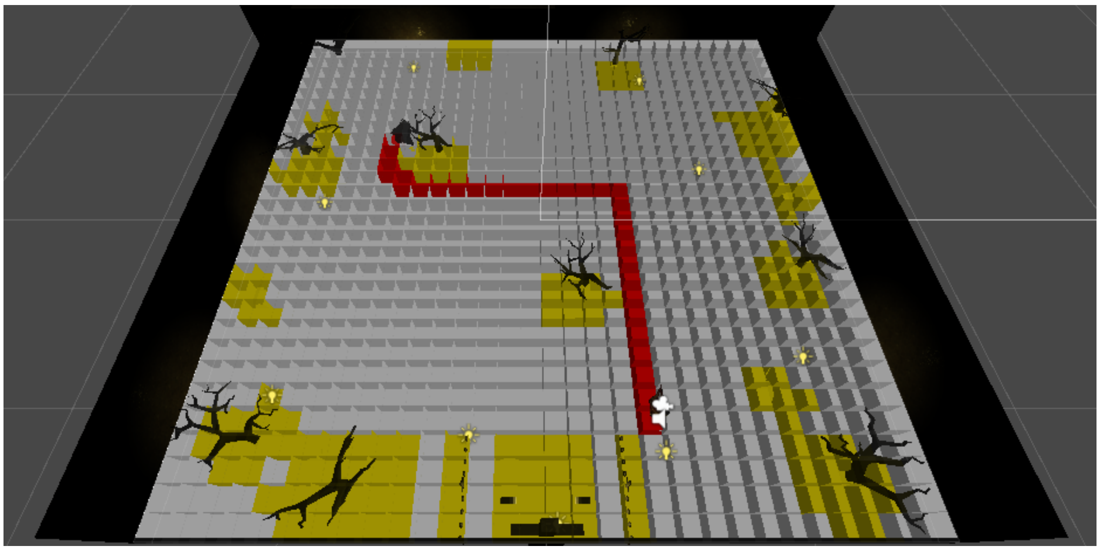


Figure 41. The pathfinding is calculated constantly until the fight ends.

The final fight is monitored by a fight controller script. To enrich the game's code, this part uses coroutines. Coroutines allow the code to stop a procedure during a certain amount of time and continue with it once that time has finished. The algorithm works like this: once the player steps a foot in the children's grave, Father Elias spawns in one of his six possible spawns randomly. The spawns are located on the other side of the room, and Elias will start running directly to the player. If Elias manages to get to the player's position, he will stab him and then disappear. The only negative effect that the player will suffer if this happens is that his/her screen will stain with blood, but it will fade out in a couple of seconds. The player does not have health points or anything because it is not desired that he/she loses the game if he/she gets stabbed many times, but the blood helps keep the atmosphere (*see Figure 42*).

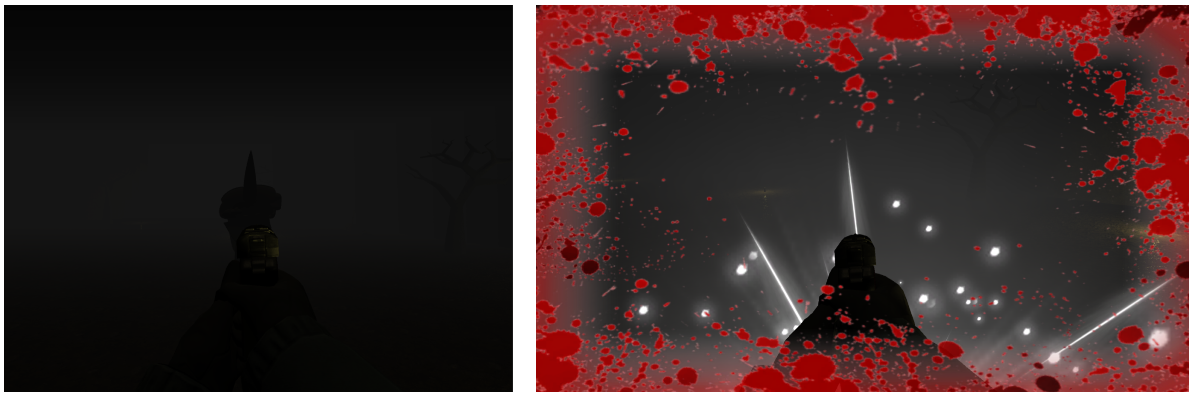


Figure 42. If Elias gets to the player, he will stab him.

On the other hand, if the player shoots Elias, he will vanish and appear in another spawn point one second later. Father Elias has got ten health points, and when this value gets lower than five, the animation of the enemy changes showing a lame movement. The same happens when this value gets lower than three, because then his animation shows him creeping on the floor and making him more difficult to shoot. The moment the player shoots his/her enemy ten times, he will appear in the middle of the basement asking for mercy and saying that killing him would be a mistake.

Once Father Elias is defeated, the player has two options: killing him or not. Next to the children’s graves there is a radio that reproduces the reporter’s last words and says that Father Elias must not die, but if the player does not listen to it or even having listened to it decides to kill the priest, the camera will shake like if an earthquake occurred and the graves will open and free the children’s souls. Based on this event, when the player gets outside the orphanage, the end screen will show the time that has taken the player to escape and the ending of his/her story.

7. Unexpected Visitor’s Chatbot: The Game Master

7.1. Introduction

In escape rooms, the role of the game master is crucial. He/she is in charge of making sure that the players do not get stuck in a puzzle too much time so they are able to finish the experience. When it comes to video games, as it was explained at the beginning of the report, they lack a game master.

Most of the games include clues in their games that players can click on and an emergent message shows on the screen with the hint of a certain puzzle, which is far different from the classic way of working of a game master. This is done this way because obviously players cannot talk to anyone the exact moment they want to because it would be crazy having a person on the other side of the screen to guide every player the 24 hours of a day.

For this project, one of the most important goals was to create a believable game master that could resemble a real game master. That is why a chatbot seemed the best option, also because no other escape room-themed video game includes one.

During the investigation about chatbots and the way they work, the most impressive thing about the definition itself were the concepts of both approaches to face the coding of the chatbot. So, by the time the development of the chatbot began, one of the primary objectives was to make an IA Chatbot slightly intelligent by making it use both of these concepts. This chatbot in particular is based on multiple other chatbots developed using these methods.

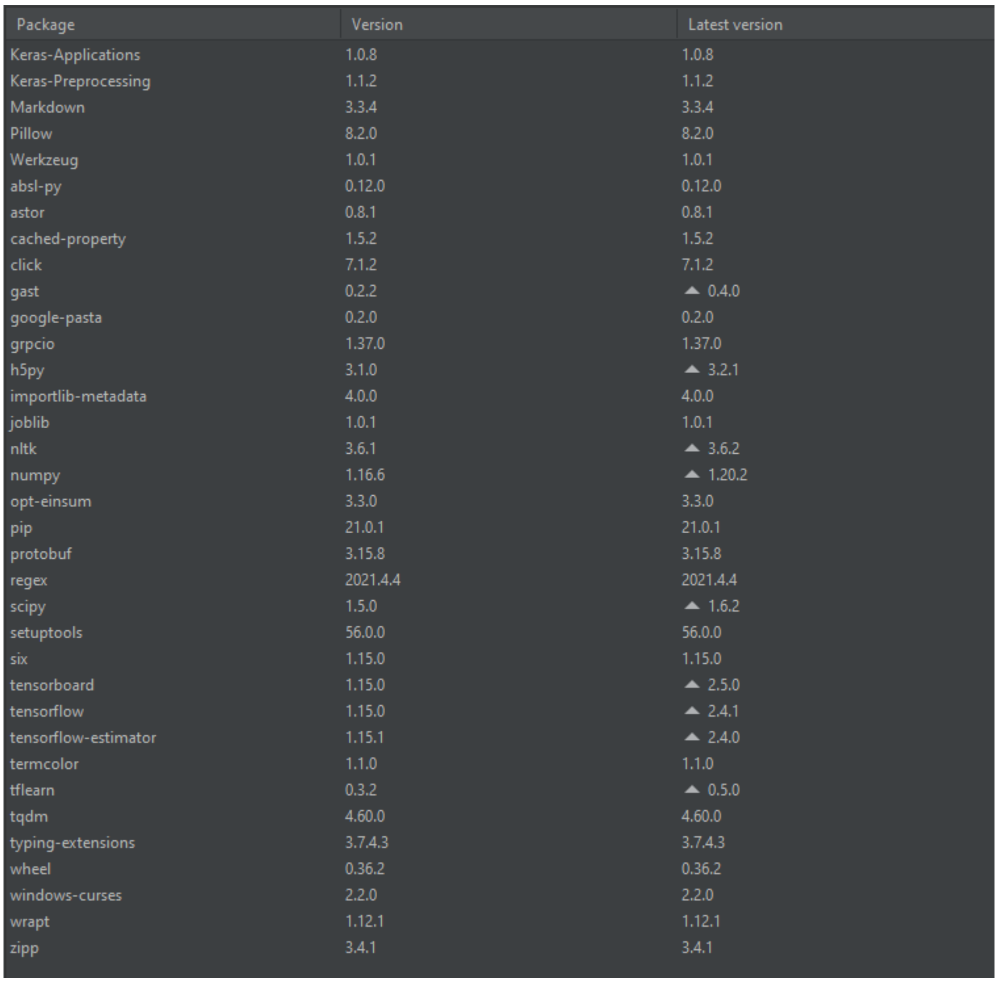
The chatbot itself has been created using the PyCharm software, and the code has been written on the version 3.6 of Python.

7.2. First Steps

To begin with the program it was needed to import a few packages in the Python Interpreter that had to be used. To do that, PyCharm includes a tool that contains every extension that can be used in a specific Python version, so the only thing the programmer needs to do is to search the name and choose the version. This task can also be carried out using the command-line interpreter (cmd) typing "pip install (name of the package)", but it only works if the programmer uses other programming softwares like Anaconda.

Installing these packages correctly is vital to avoid compiling errors, because depending on each package's version the code will not work. The packages installed were "nltk" in its latest version (this one does not affect negatively when compiling), "numpy" in its 1.16.6 version (versions above 1.17 are not valid for the code), "tensorflow" in its 1.15 version (versions above 2.0 do not work for this code) and "tflearn" in its latest version.

Even though those packages are the primary ones, once the code testing was giving compiling errors, it was also needed to install the package "curses" in its latest version and "pickle" in its latest version as well (*see Figure 43*).

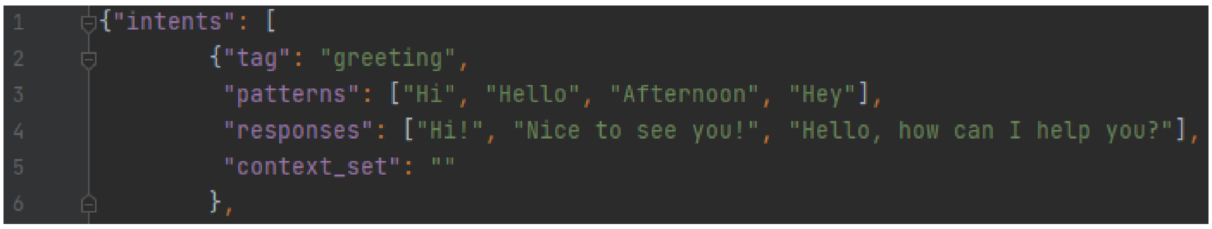


Package	Version	Latest version
Keras-Applications	1.0.8	1.0.8
Keras-Preprocessing	1.1.2	1.1.2
Markdown	3.3.4	3.3.4
Pillow	8.2.0	8.2.0
Werkzeug	1.0.1	1.0.1
absl-py	0.12.0	0.12.0
astor	0.8.1	0.8.1
cached-property	1.5.2	1.5.2
click	7.1.2	7.1.2
gast	0.2.2	▲ 0.4.0
google-pasta	0.2.0	0.2.0
grpcio	1.37.0	1.37.0
h5py	3.1.0	▲ 3.2.1
importlib-metadata	4.0.0	4.0.0
joblib	1.0.1	1.0.1
nltk	3.6.1	▲ 3.6.2
numpy	1.16.6	▲ 1.20.2
opt-einsum	3.3.0	3.3.0
pip	21.0.1	21.0.1
protobuf	3.15.8	3.15.8
regex	2021.4.4	2021.4.4
scipy	1.5.0	▲ 1.6.2
setuptools	56.0.0	56.0.0
six	1.15.0	1.15.0
tensorboard	1.15.0	▲ 2.5.0
tensorflow	1.15.0	▲ 2.4.1
tensorflow-estimator	1.15.1	▲ 2.4.0
termcolor	1.1.0	1.1.0
tflearn	0.3.2	▲ 0.5.0
tqdm	4.60.0	4.60.0
typing-extensions	3.7.4.3	3.7.4.3
wheel	0.36.2	0.36.2
windows-curses	2.2.0	2.2.0
wrapt	1.12.1	1.12.1
zipp	3.4.1	3.4.1

Figure 43. Every package imported to the project with their version

7.3. JSON File

Before starting coding, it was necessary to add a ".json" file to the project folder that contained the information the chatbot had to analyse and the responses it might answer, which is a perfect graphical definition of the graphmaster from the pattern matching approach explained in the previous chapter (*see Figure 44*).

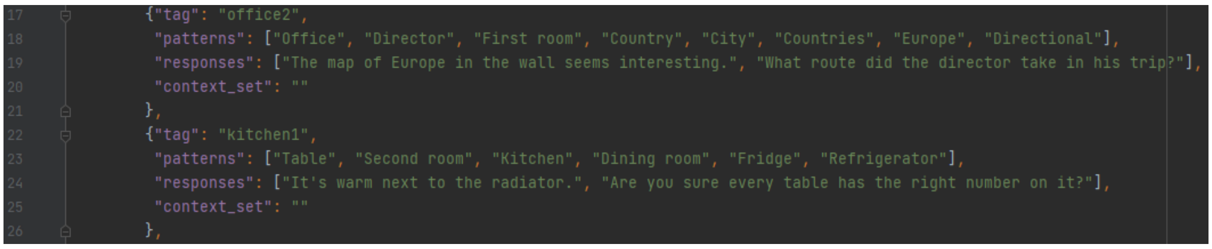


```
{"intents": [
        {"tag": "greeting",
         "patterns": ["Hi", "Hello", "Afternoon", "Hey"],
         "responses": ["Hi!", "Nice to see you!", "Hello, how can I help you?"],
         "context_set": ""
        },
```

Figure 44. Content of the information the chatbot will have to analyse for each "tag".

As it shows, the structure contains a main list called “intents”, which includes every possible conversation topic that the player can discuss with the bot. For example, the first one of it would be the “greeting” tag, which includes possible words that the player can type in his/her message. The patterns to take into account for this tag would be words like “Hello”, “How are you”, “Greetings”, etc. If the bot detects that the player is sending a greeting message, it will answer with one of its possible answers.

In the case of the room tags, the possible answers that the bot can give are clues to help the player guess the correct way to solve the enigma (*see Figure 45)*.



```
        {"tag": "office2",
         "patterns": ["Office", "Director", "First room", "Country", "City", "Countries", "Europe", "Directional"],
         "responses": ["The map of Europe in the wall seems interesting.", "What route did the director take in his trip?"],
         "context_set": ""
        },
        {"tag": "kitchen1",
         "patterns": ["Table", "Second room", "Kitchen", "Dining room", "Fridge", "Refrigerator"],
         "responses": ["It's warm next to the radiator.", "Are you sure every table has the right number on it?"],
         "context_set": ""
        },
```

Figure 45. Chatbot’s responses for game-related questions are divided by rooms and a numbers if a room contains more than one enigma.

7.4. The Code

The code itself starts importing every package that is going to be used and the components of most of those packages, the ones that have been commented before. After that, the “.json” file has to be opened and its information has to be stored in a variable called “data”. “punkt” is an extension of the “nltk” package that has to be downloaded to avoid compiling errors and to make the word tokenizer work.

The stemmer will be used to stem the accessed words, which is the process of reducing the extension of the words to their root forms. After that, it is necessary to create a few lists to save specific data from the ”.json” file in each of them. First, the bot is going to loop through all the dictionaries and stem every word of the “patterns” section.

As mentioned in the previous paragraph, stemming is the first intelligent behaviour that the bot is going to do. With stemming, every word from the “patterns” section is going to be changed to its root form by using the “word_tokenize()” function. In language, most words can be used by adding a prefix, suffix or infix to describe an action or change the meaning of the sentence, but all of those words are part of a root word.

For example, the words “Playing”, “Plays” or “Played” have the common stem form “Play”, and this is what this function does. Also, the words “Am”, “Are” or “Is” belong to the verb “Be”, so if for example the player writes a sentence like “The boy’s cars are different colours?”, the information stored in the stemmed variable will be “The boy car be differ colour”, so that with this behaviour the bot will be able to store the useful information of the sentence and ignore the useless one.

With the ".extend" function, all the data stored in the stemmed variable is stored in the list "words", which is more optimal than to append every word one by one. On the list "docs_x", the stemmed pattern is stored, and on "docs_y" the bot knows what tag the pattern stored belongs to. At last, all the tags are stored in the "labels" list.

Next, the bot uses the ".stem()" function to separate the stemmed words in individual gaps of a list. For example, if the previous content of "words" is ["The boy car be differ colour"], it will now have ["the", "boy", "car", "be", "differ", "colour"]. All the capital letters are lowered with the function ".lower()" to avoid possible understanding errors.

With the next line, the function "set()" removes all the duplicate components the list might have, then turns the components back into a list (as "set()" only works with DataType) and with "sorted()" the bot gets the list ordered in alphabetical order. After that, the bot sorts the "labels" list too.

The next part is creating the outputs and making them work with the Neural Network. So far the bot has got lists of texts (string), but neural networks only understand numbers (int), so the bot needs to create what is known as "Bag of words", which represents all the words in a given pattern, and then use that to train its model. Also, a bag of words is known as "One hot encoded", which means that the bot is going to have a list the length of the amount of words that it has, so if for example it had 7 words then each encoding is going to have 7 entries of numbers.

Each position of the list represents if a word exists or if a word does not exist, so following the previous example the written sentence by the player has its final stem appearance ["be", "boy", "car", "colour", "differ", "the"], the bot is going to check if each word is contained in any of the pattern's options, so maybe it finds the word "boy" and the word "differ" in the "greeting" pattern its bag of words will have the appearance [0, 1, 0, 0, 1, 0].

Let's say that the other tags do not include any of those words (so their bag of words will look like [0, 0, 0, 0, 0, 0]) but the encoding "goodbye" contains the words "car", "be" and "differ", so its bag of words will look like [0, 1, 1, 0, 1, 0]. As it can be seen, the encoding "goodbye" contains more words than any other encoding, so the bot will probably choose an answer from that encoding to respond to the player.

The last part of the code is to build the bot's model using the package "tflearn". First, the "reset_default_graph()" function resets the data graph to avoid possible errors with previously outdated saved settings. The "input_data()" function defines the input shape that is expected for the model, in this case it takes the length of "training[0]" because each "training" input is going to be the same length.

Then, the following lines use the "fully_connected(layer, number of neurons)" function to add a layer to the bot's neural network with a specific number of neurons. The layer of line 80 allows the bot to get probabilities for each output by using "softmax" activation.

Understanding that previous code is essential to know how this neural network works, so let's try to represent it using the example mentioned before. The player types the sentence "The boy's cars are different colours?", so once all the process explained before is carried out the bot has six words. Each of those words is an input, and they are located in the first layer. As it can be seen, the next layer contains eight neurons, and each of the input layer's networks are connected to each second layer's networks (in the following schema there is only one of them connected for the sake of clarity (*see Figure 46*)).

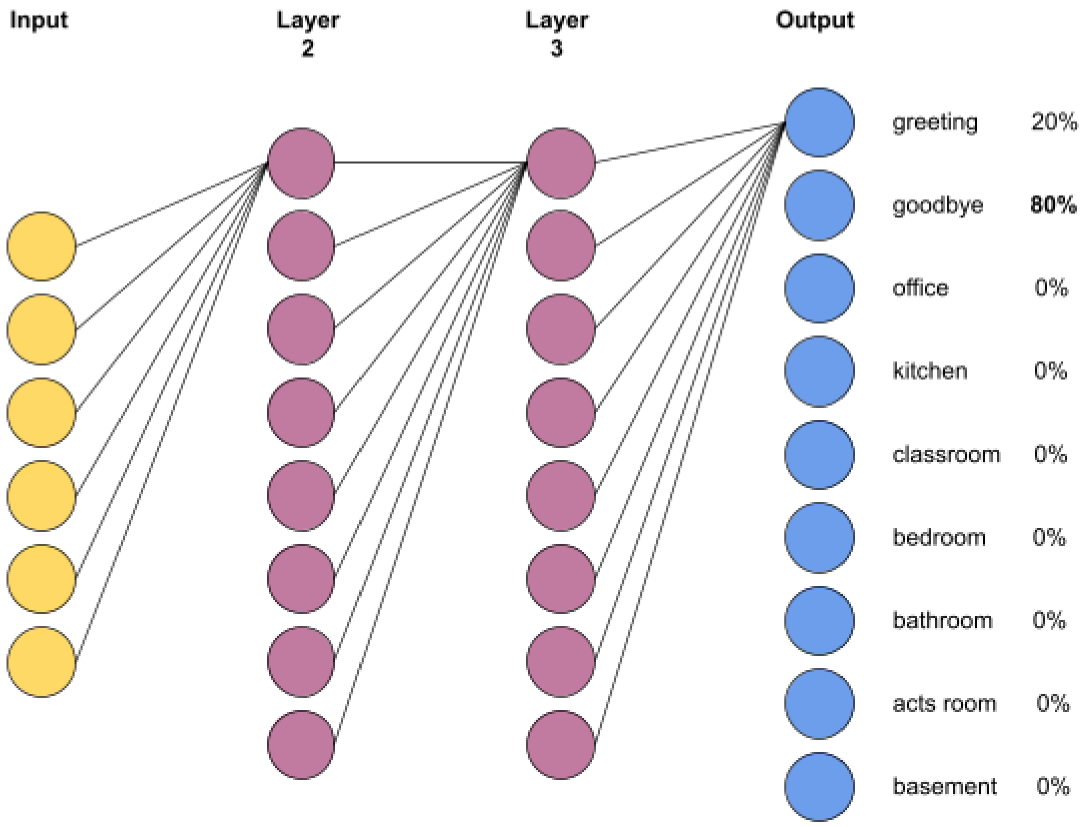


Figure 46. Behaviour of the bot when it comes to decide the most probable "tag" that player is talking about with his/her question.

The third layer also has eight neurons, and following the same behaviour as before all of them are fully connected to the previous layer's neurons. Then, at last there is the final layer which is the output layer, which has a "softmax" activation and has nine neurons (one per each tag), also fully connected to the previous layer's neurons.

Softmax activation gives each neuron a probability of being the one that the player is talking about. So, following the example commented before, the bag of the "greeting" tag was [0, 1, 0, 0, 1, 0], the bag of the "goodbye" tag was [0, 1, 1, 0, 1, 0] and all the rest bags of the rest of the tags were [0, 0, 0, 0, 0, 0]. Taking that into account, the "softmax" behaviour would give a big proability to the "goodbye" tag, a decent probability to the "greeting" tag and almost zero probability to the rest of the tags. The bot will decide which tag it chooses based on the probabilities and select a random answer from that tag.

The interesting thing about the neural network is that once the player starts training and feeding information to the chatbot's model, if a word is typed by the players most of the time like for example the word "Hello", the layers will start changing some biases and weights to get to the "greeting" tag faster than before.

7.5. Saving the Data

The next part of the process is to save the data, so that if all the lists are already full of the content they need to have, all the process of creating those lists and appending the information is not needed to be done.

So, before doing all that code, the package "pickle" allows the bot to try to open the data located in a file saved as bytes ("rb" stands for "read bytes") and if it finds the lists it needs to carry out with the rest of the program, it is not needed for him to get all that data again. If it does not find the correct information on those lists or it finds them empty, all the code explained before is carried out.

Also when the "except" code finishes, all that gathered data has to be stored in the pickle file, which is a code similar to the previous one but changing the "rb" (read bytes) for "wb" (write bytes) and also adding the function ".dump()" to store the information of the lists "words", "labels", "training" and "output" on the pickle file.

7.6. "Bag of Words" Function

The "bag_of_words" function is very similar to the output creation done above, but it will be very functional on the bot's "chat" function. It starts creating bags full of zeros and if the stemmed list of words includes one of any tag's possible patterns, it will change the zero for the one.

7.7. "Chat" Algorithm

At the beginning of the "chat()" function, the bot tells the player that if he/she wants to stop speaking to the bot he/she has to type the word "Exit", which would end with the program execution.

After that, the function "bag_of_words" is called and its content is loaded on a variable called "results" with all the predictions already made thanks to the "predict()" function. But this new variable does not contain an accurate output that can be given to the player, it only contains a list of the probability that each tag has. To get only the greatest number of that list, the function “argmax()” does exactly that, and it stores that value of that probability on a variable called “results_index”.

Then, on a new variable called “tag”, the bot stores the string of the tag that had the greatest probability mentioned before. If that was the final output, when for example the player types “Hello”, the bot would respond “greeting”, which is the tag of words like “Hello”. Then, the bot makes a comparison: if any of the possible input words does not have a probability greater than 70% of belonging to one of the tags, the chatbot will consider that that input written by the player does not have anything to do with the conversation topics it can talk about, so it will just print a sentence like “Sorry, I cannot understand you. Please, try again.”. This improves the intelligence of the bot, because if this comparison did not exist and the player just asked a dumb question, the bot would pick any of the possible answers that had even a 0.01% probability of being the correct one.

Otherwise, if the bot considers that one tag has a chance greater than 70% of being the correct one, it will just pick a random answer to that topic, resulting in an optimal and accurate ending of the program.

Once the function is finished, it will be called on the last line of the program. With all that code, the result is a perfectly functional and intelligent game master for “Unexpected Visitor”.

7.8. Uploading the Chatbot to a Discord Server

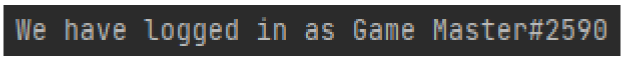
There is a package of Python to implement Unity code, but they work together in a very primitive way and in versions that are now almost obsolete, so the idea of implementing this exact chatbot inside the game is impossible. But despite this, there is a very interesting solution for this: implementing the chatbot in a Discord server.

Discord is an instant messaging application that allows users to talk to each other using text, video and voice among other applications. Since its creation in 2015, is has not stopped growing around the world, having around 25 million registered users in just one year and achieving its maximum value of 300 million registered users in 2020. Nowadays it is one of the most used instant messaging applications that holds around 140 million users per month around the world, which makes it a perfect place to hold the Game Master chatbot of Unexpected Visitor.

So, in order to upload the chatbot to a Discord server, it was necessary to first create it in “Discord Developer Portal”, a webpage where people can create and upload bots. The proceeding is very intuitive, and the most important information that was needed to get from here was the *Token* address. A token is a “word” represented by letters, numbers and symbols that acts as a “key” to control a discord bot. An analogy would be that if a chatbot is a door, a token is the key to open/use that door.

Once the token is copied, it is time to go back to Python to adapt the previous code. The first thing to do is to install the “discord” package as it was done at the beginning of the chatbot creation. Once it is done, the token can be pasted in the code.

Then, the next thing is to create a discord client, which receives an event from Discord and locates the interactions with the server, messages in this case. Then, the bot performs a coroutine to log in the server, which will make it visible for the rest of the users and will print a message saying that it has connected successfully (*see Figure 47*).



```
We have logged in as Game Master#2590
```

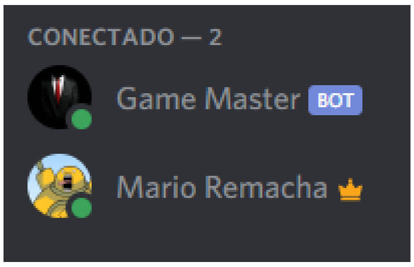
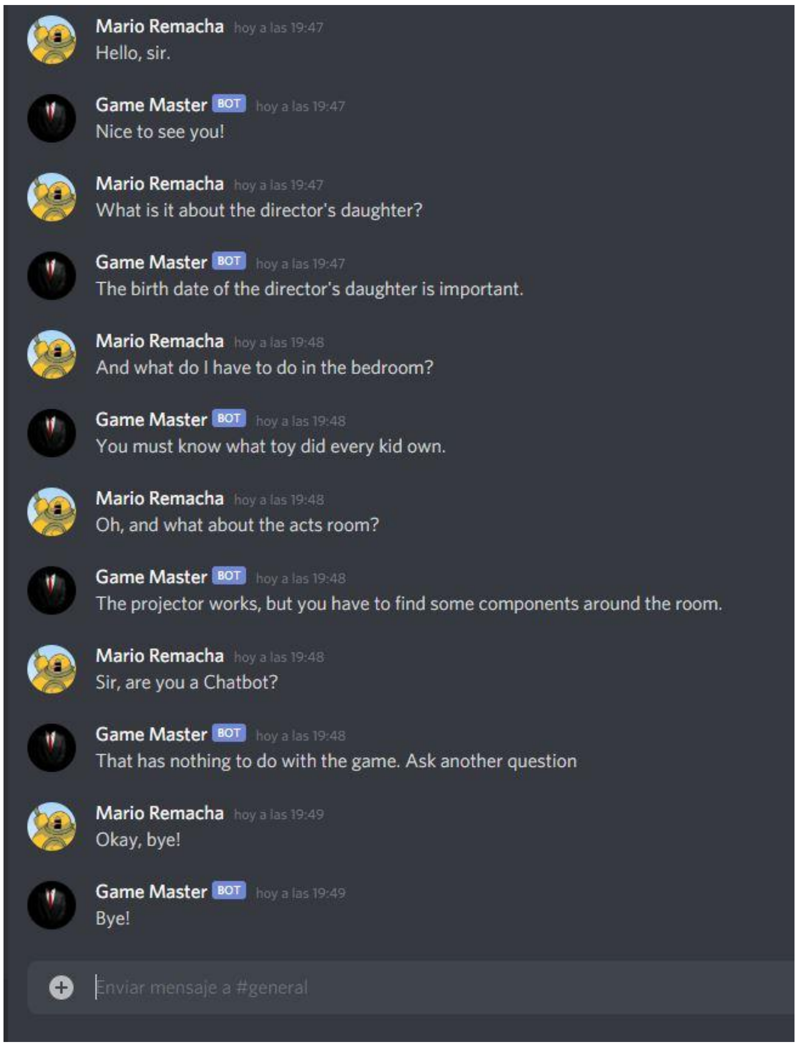


Figure 47. The chatbot will log in the server just like any other user.

Then, another coroutine is created to take control of the messages displayed in the server. This function will substitute the “Chat()” function, which is included with most of its previous code in the new function. First, the bot declares the variable *username* (which will give it the name of the user’s nickname), the variable user_message (which stocks the user’s message) that substitutes “Chat()”’s variable *inp* (input) and the variable *channel* (which gets the name of the channel the bot will talk in). Then, in the Python console, it will print the name of the user followed by his/her message and the channel he/she is talking in. As the chatbot himself is also an user, its message will appear too.

The next two lines are added because if they are not there, the bot would be always answering to its own last message. This way, if the last message was written by himself, it just stops answering. Then, if the user’s message has been written in the “general” chat (one Discord server can include many chat spaces, for example one for playing video games, another one for working, etc.) it reproduces the “Chat()” function and sends a response message (*see Figure 48*).


Mario Remacha hoy a las 19:47
Hello, sir.

Game Master BOT hoy a las 19:47
Nice to see you!

Mario Remacha hoy a las 19:47
What is it about the director's daughter?

Game Master BOT hoy a las 19:47
The birth date of the director's daughter is important.

Mario Remacha hoy a las 19:48
And what do I have to do in the bedroom?

Game Master BOT hoy a las 19:48
You must know what toy did every kid own.

Mario Remacha hoy a las 19:48
Oh, and what about the acts room?

Game Master BOT hoy a las 19:48
The projector works, but you have to find some components around the room.

Mario Remacha hoy a las 19:48
Sir, are you a Chatbot?

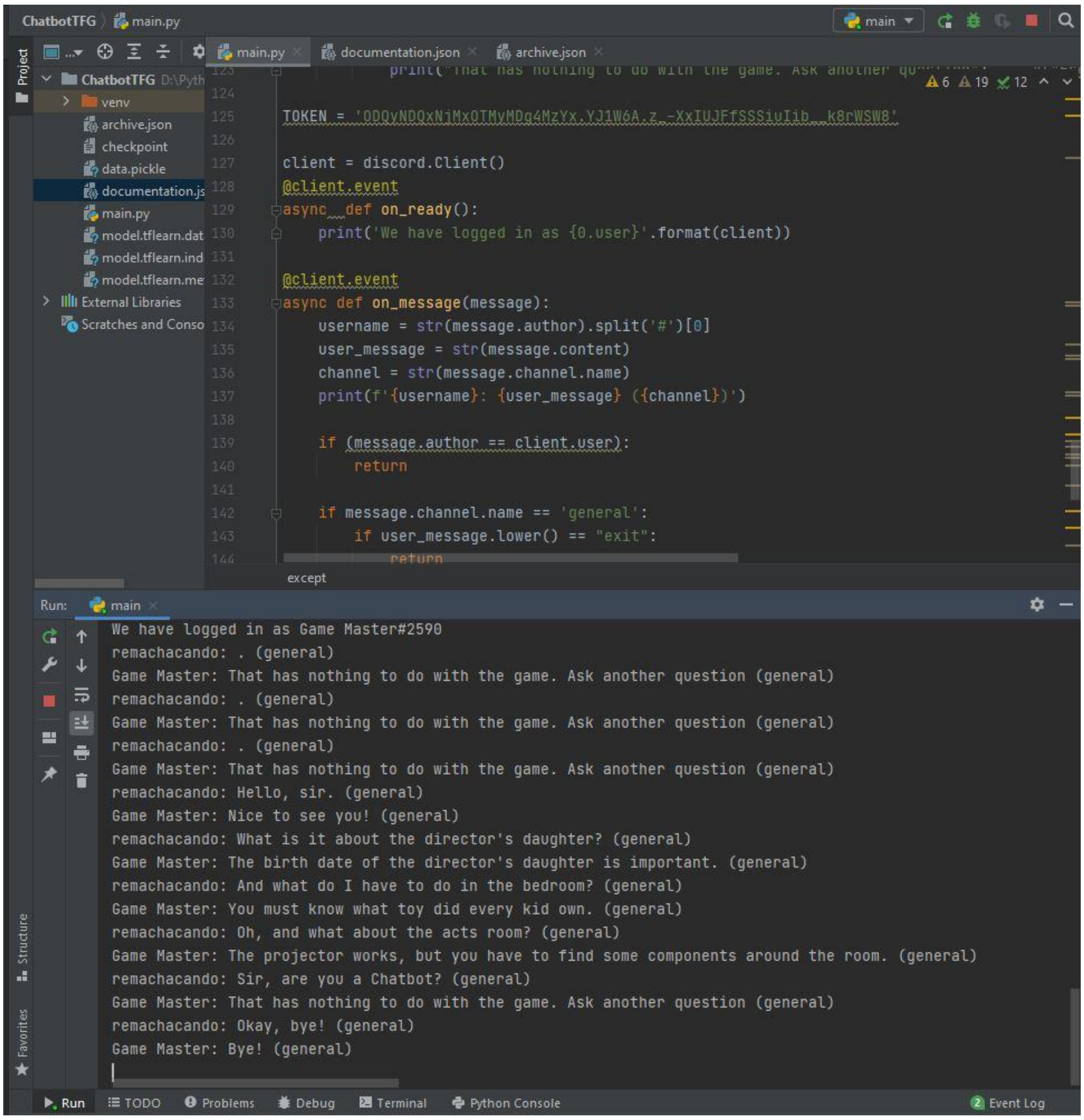
Game Master BOT hoy a las 19:48
That has nothing to do with the game. Ask another question

Mario Remacha hoy a las 19:49
Okay, bye!

Game Master BOT hoy a las 19:49
Bye!

Figure 48. Discord's conversation between an user and the chatbot.

Also, as it was explained before, the Python console also prints every part of the Discord's conversation (*see Figure 49*).



```
            print("That has nothing to do with the game. Ask another qu

TOKEN = 'ODQyNDQxNjMxOTMyMDg4MzYx.YJ1W6A.z_-XxIUJFfSSSiuIib__k8rWSW8'

client = discord.Client()
@client.event
async  def on_ready():
    print('We have logged in as {0.user}'.format(client))

@client.event
async def on_message(message):
    username = str(message.author).split('#')[0]
    user_message = str(message.content)
    channel = str(message.channel.name)
    print(f'{username}: {user_message} ({channel})')

    if (message.author == client.user):
        return

    if message.channel.name == 'general':
        if user_message.lower() == "exit":
            return
```

```
We have logged in as Game Master#2590
remachacando: . (general)
Game Master: That has nothing to do with the game. Ask another question (general)
remachacando: . (general)
Game Master: That has nothing to do with the game. Ask another question (general)
remachacando: . (general)
Game Master: That has nothing to do with the game. Ask another question (general)
remachacando: Hello, sir. (general)
Game Master: Nice to see you! (general)
remachacando: What is it about the director's daughter? (general)
Game Master: The birth date of the director's daughter is important. (general)
remachacando: And what do I have to do in the bedroom? (general)
Game Master: You must know what toy did every kid own. (general)
remachacando: Oh, and what about the acts room? (general)
Game Master: The projector works, but you have to find some components around the room. (general)
remachacando: Sir, are you a Chatbot? (general)
Game Master: That has nothing to do with the game. Ask another question (general)
remachacando: Okay, bye! (general)
Game Master: Bye! (general)
```

Figure 49. Python also writes the Discord's conversation.

8. Study: Does Tension Affect Human's Capacity Of Concentration?

Once people finished playing the game, they were asked a series of questions about it to receive feedback and also carry out a study on the connection between tension and concentration. First, to decide the average time the game lasts for most people that play it, players were asked how much time did it take them to finish their experience (*see Figure 50*). Based on the results, most people finished the game between the 45-75 gap of minutes, so in order to guarantee that most players finish their experience successfully and with a victory it has been determined to place the mark of 75 minutes as the maximum amount of time to finish the game successfully.

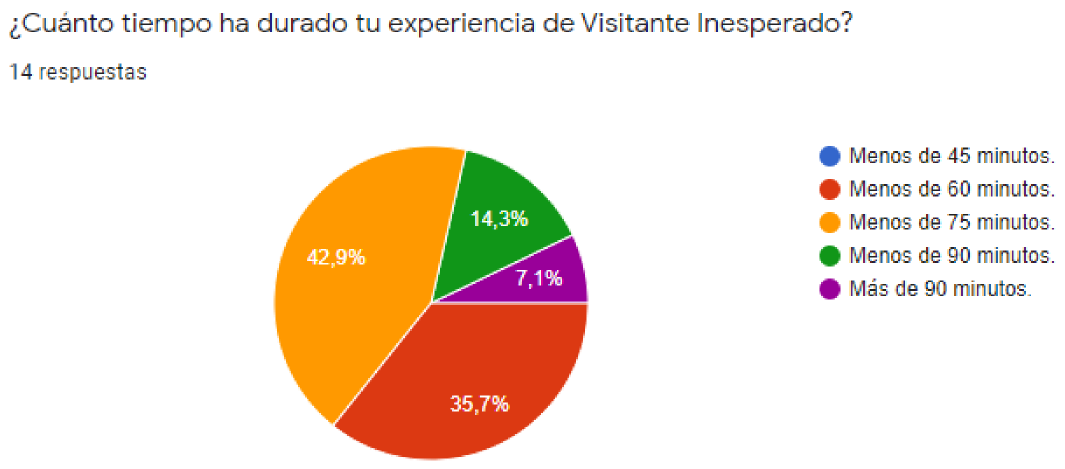


Figure 50. Graph on player's taken time to finish the game.

Also, in order to carry out this study it was needed to have two different groups that played two different versions of the game, one with horror interactions and the other without them (this second one would be the same as the first one but without the interactions with Father Elias, including the final fight, and all the events commented in the point 2.5.2.). Despite this, people were given the option of playing any of them (*see Figure 51*).

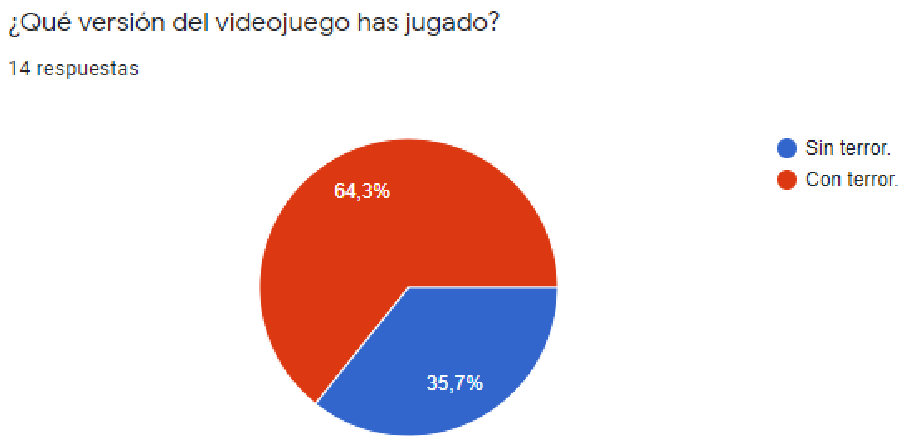
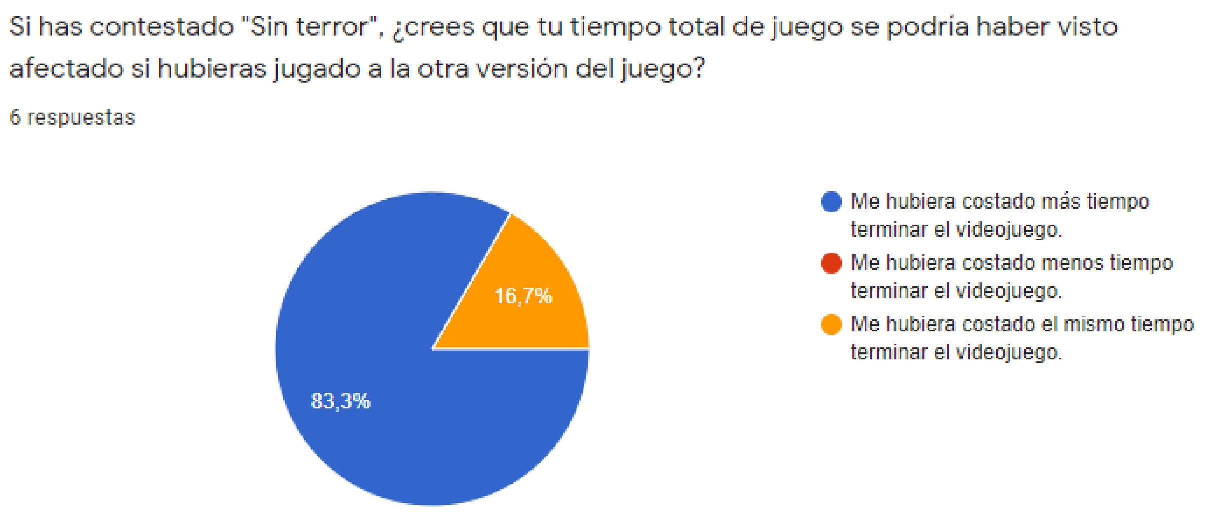


Figure 51. Almost ⅔ of the players prefered to play the horror version of the game.

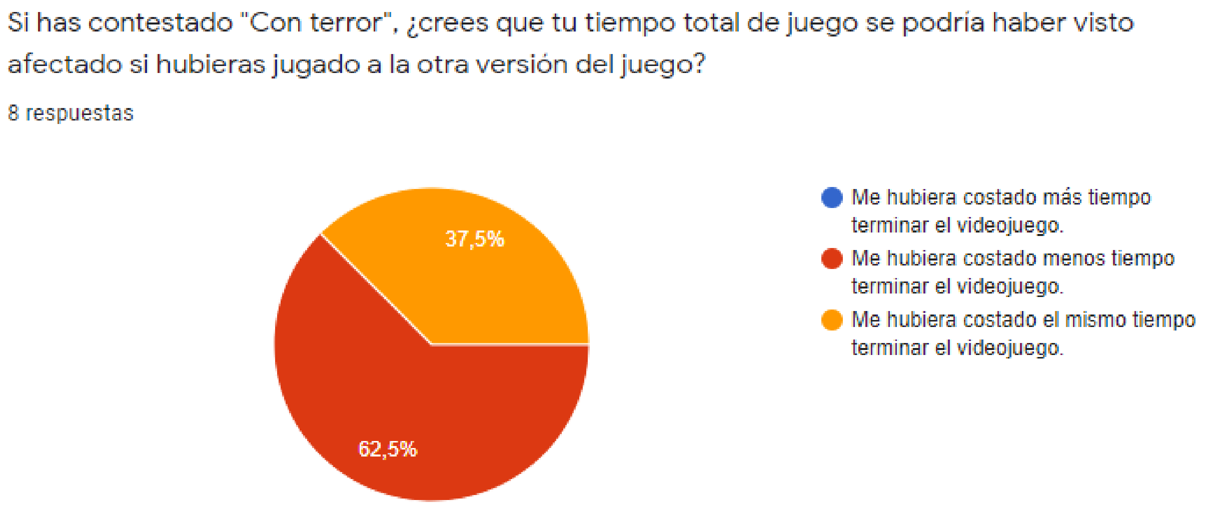
As it can be seen (*see Figure 52*), people who played the non-horror game think that playing the horror version would have taken them more time to finish the game. This is because horror affects people's tension and therefore their capacity to concentrate on their task.


Si has contestado "Sin terror", ¿crees que tu tiempo total de juego se podría haber visto afectado si hubieras jugado a la otra versión del juego?

6 respuestas

Figure 52. Non-horror gamers' opinion on both gamemodes.

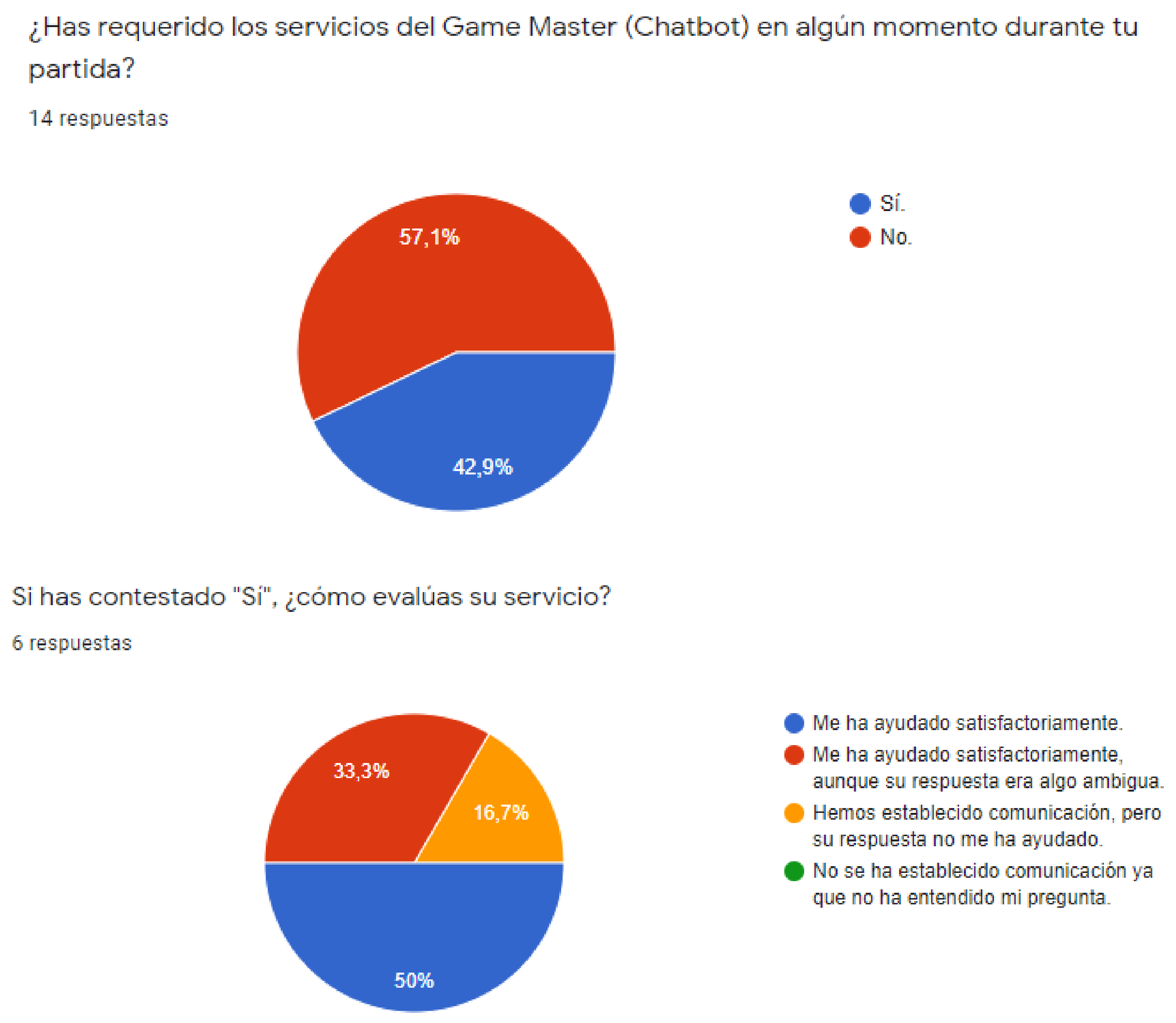
On the other hand, horror version gamers think that it would have taken them less time to finish the game if they played the other version. Others, who enjoy the horror genre in any form, are not too affected by its immersive factor and think that it would have not mattered which version they played the ending time would have been the same (*see Figure 53*).


Si has contestado "Con terror", ¿crees que tu tiempo total de juego se podría haber visto afectado si hubieras jugado a la otra versión del juego?

8 respuestas

Figure 53. Horror gamer's opinion on both gamemodes.

Also, in the last part of the questionnaire, players were asked about the chatbot and its utility. A bit more than half of the players did not need him to solve the enigmas, but those who required assistance value its assistance mostly as satisfactory. Future versions of the chatbot will keep working on the bot’s responses to make sure that everyone is satisfied with its service (*see* Figure 54).


¿Has requerido los servicios del Game Master (Chatbot) en algún momento durante tu partida?

14 respuestas

Si has contestado "Sí", ¿cómo evalúas su servicio?

6 respuestas

Figure 54. Players’ opinions on the chatbot’s services.

9. Conclusions

The conclusions taken at the ending of this project are very optimistic about every commented topic. Live escape rooms are now more popular than ever and many companies and entrepreneurs see the option of an escape room business more profitable than other topics, which is great for the community and the business in general.

In what comes to escape room video games, thanks to the development of virtual reality it is becoming a very requested genre of video game and important companies are joining this amazing business. This fact is not only beneficial for gamers in a playful way, the advances in virtual reality can give the chance of creating escape rooms destined with educational or even medical ways too, which could be a perfect therapy for the users.

Creating *Unexpected Visitor* has been a path full of stones in the process, from the designing process to the coding one. Bugs and compiling errors are always present during the trip and it is sometimes very stressful, but it feels nice when everything is over and people start sharing good opinions with each other.

On the other hand, in what comes to chatbots, it is generally a very underrated technology. If people stopped for a second and started thinking about the concept of a learning machine in a profound way, it would blow their minds and kill them of curiosity. Even if Internet users do not realize, chatbots are present in almost any webpage and companies are starting using them for most of their informational spaces. They can also be a very good companion for people who do not interact too much with the outside world or with other people and make them feel thought about and cared for. They could also have a relevant use in education and health fields as their utility has been proven on many occasions.

Creating the Game Master was really challenging above all at the beginning when it came to choose the approaches it was going to use and the needed packages and their versions needed to develop it. Coding it and understanding its working method was very satisfactory and enjoyable, and seeing its final look inside a Discord server makes it even more beautiful.

At last, the study is not too representative as not many people have played the game and answered the questionnaire, but as more people do it the data will be updated and taken into account in case any conclusion changes. But apart from that, gamers enjoyed taking part in it and giving their thoughts as feedback, which is always appreciated.

10. Bibliography

10.1. “Escape Rooms: The Mine Of Gold Of Video Game Industry” Bibliography

10.1.1. Global Information

Escape Kit - What is an escape room? (https://escape-kit.com/en/what-is-an-escape-room/)

Anim Escape - The history of escape rooms (https://aimescape.com/blog/post/the-history-of-escape-rooms#:~:text=The%20history%20of%20escape%20rooms%20is%20a%20hotly%20debated%20topic.&text=The%20first%20live%20escape%20room,escape%20from%20a%20locked%20room.)

The Escape Game - What is an escape room (https://theescapegame.com/blog/what-is-an-escape-room/)

The Free Dictionary - Alternate reality game (https://www.thefreedictionary.com/alternate+reality+game)

20 Minutes - Evolution of chatbots (https://www.20minutos.es/noticia/4635577/0/desde-alan-turing-hasta-siri-como-han-evolucionado-los-chatbots/)

Lock Academy - History and origin of escape games (https://lockacademy.com/en/history-and-origin-of-escape-games/)

Breakout Game - Escape rooms history (https://breakoutgames.com/escape-rooms/history)

The Escape Game - The history of escape rooms (https://theescapegame.com/blog/the-history-of-escape-rooms/)

New York Times - In escape rooms, video games meet real life (https://www.nytimes.com/2014/06/04/arts/video-games/in-escape-rooms-video-games-meet-real-life.html)

Psychology Of Games - The psychology of immersion in video games (http://www.psychologyofgames.com/2010/07/the-psychology-of-immersion-in-video-games/)

Wikipedia - Mihaly Csikszentmihalyi (https://en.wikipedia.org/wiki/Mihaly_Csikszentmihalyi)

Brantford Games Network Lab - Ontario escape room unconference 2015 (http://bgnlab.ca/blog/2015/10/27/ontario-escape-room-unconference-2015.html)

Scott Nicholson - Peeking behind the locked door: A survey of escape room facilities (http://scottnicholson.com/pubs/erfacwhite.pdf)

Filmaffinity - Escape Room (https://www.filmaffinity.com/es/film747808.html)

Filmaffinity - La habitación de Fermat (https://www.filmaffinity.com/es/film615789.html)

Filmaffinity - Cube (https://www.filmaffinity.com/es/film741341.html)

10.1.2. Figures

Figure 5:
Sens Critique - Crimson Room
(https://www.senscritique.com/jeuvideo/Crimson_Room/36233264)

B Sting - Crimson Room (https://www.b-sting.nl/crimsonroom/solve.html)

Figure 6:
Filmaffinity - Cube (https://www.filmaffinity.com/es/film741341.html)

Figure 7:
Filmaffinity - La habitación de Fermat (https://www.filmaffinity.com/es/film615789.html)

Figure 8:
Im Joying - Bunker (https://imjoying.com/events/abd125cd-cc91-4086-9539-8898486fbd69)

Marguerite Ba - Magic circle from escape cube game of (https://www.xn--ngbkm8d.online/4212/magic-circle-from-escape-cube-game-of)

Figure 9:
Wiki How - Escape the red room
(https://es.wikihow.com/escapar-del-Cuarto-Rojo-(The-Crimson-Room))

Over Blog - Cube Escape Paradox chapter 2
http://uculcobuc.over-blog.com/2019/08/Cube-Escape-Paradox-Chapter-2-crack-32-bit.html

Figure 10:
How Long To Beat - The Room (https://howlongtobeat.com/game?id=10153)

Steam - The Room (https://store.steampowered.com/app/288160/The_Room/)

Figure 11:
Ubisoft - Escape the Lost Pyramid
(https://news.ubisoft.com/en-us/article/5FeI3K10xxDPL9PMtCwhIw/escape-the-lost-pyramid-in-a-vr-escape-room-set-in-the-world-of-assassins-creed-origins)

Real O Virtual - Ubisoft Escape Games: Analysis
(https://www.realovirtual.com/articulos/5375/ubisoft-escape-games-analisis/3/escape-lost-pyramid)

Figure 12:
New Atlas - Virtual reality escape room
(https://newatlas.com/vr/autronvr-virtual-reality-escape-room/)

Castellón Plaza - The first virtual reality escape room comes back to Castellón
https://castellonplaza.com/LaprimeraescaperoomderealidadvirtualvuelveaCastellconunanuevamodalidad

10.2. “Chatbots: The Game Master No One Thought About” Bibliography

10.2.1. Global Information

Expert - Chatbot (https://www.expert.ai/blog/chatbot/)

Chat Compose - What are chatbots (https://www.chatcompose.com/what-are-chatbots.html)

Paldesk - Chatbot: What is it and how to build one
(https://www.paldesk.com/chatbot-what-it-is-and-how-to-build-one/)

Tech Target - Definition of chatbot
(https://searchcustomerexperience.techtarget.com/definition/chatbot)

IBM - Watson Assistant (https://www.ibm.com/cloud/watson-assistant/)

Digital Trends - What is Amazon’s Alexa and what can it do
(https://www.digitaltrends.com/home/what-is-amazons-alexa-and-what-can-it-do/)

Talking To The Crowd - A conversational agent (http://talkingtothecrowd.org/index.html)

AAAI - Savenkov (https://aaai.org/ocs/index.php/HCOMP/HCOMP16/paper/view/14049) arXiv - Natural language processing (http://arxiv.org/abs/1708.05148)

Springer Link - Accessing an information system by chatting (https://doi.org/10.1007/978-3-540-27779-8_39)

Medical Futurist - Top health chatbots (https://medicalfuturist.com/top-12-health-chatbots/)

ChatterBot - About ChatterBot (https://chatterbot.readthedocs.io/en/stable/#)

Microsoft - Microsoft Bot Framework (https://dev.botframework.com/)

Google Cloud - Contexts (https://cloud.google.com/dialogflow/docs/contexts-overview)

Engage Hub - Considerations when implementing a chatbot (https://engagehub.com/blog/what-are-the-key-considerations-when-implementing-a-chatbot)

Chatbots Magazine - 5 core considerations for choosing your chatbot (https://chatbotsmagazine.com/5-core-considerations-for-choosing-your-chatbot-4f4bb0856ead)

Netomi - Worst chatbot fails and how to avoid them (https://www.netomi.com/the-worst-chatbot-fails-and-how-to-avoid-them)

Toptal - Chatbot fails (https://www.toptal.com/designers/ux/chatbot-fails)

Core DNA - Chatbot fail (https://www.coredna.com/blogs/chatbot-fail#1)

Servis Bot - Four reasons for enterprise chatbot failure (https://servisbot.com/four-reasons-for-enterprise-chatbot-failure/)

CIO - Three ways to mitigate chatbot risks (https://www.cioapplications.com/cxoinsights/three-ways-to-mitigate-chatbot-risks-nid-2221.html)

World Health Organization - HealthBuddy (http://www.euro.who.int/en/health-topics/health-emergencies/coronavirus-covid-19/news/news/2020/5/healthbuddy-a-new-chatbot-to-engage-with-communities-in-europe-and-central-asia-on-covid-19)

Acquire - Chatbot use cases (https://acquire.io/blog/chatbot-use-cases/)

Revechat - Chatbots use cases (https://www.revechat.com/blog/chatbots-use-cases/)

Bot Core - Top chatbot use cases in different industries
(https://botcore.ai/blog/top-chatbot-use-cases-in-different-industries/)

Jiyou Jia - NLML (https://arxiv.org/ftp/cs/papers/0404/0404018.pdf)

IEEE Xplore - The Uncanny Valley
(https://ieeexplore.ieee.org/document/6213238)

Oxford Academic - Computing machinery and intelligence
(https://academic.oup.com/mind/article/LIX/236/433/986238)

10.2.2. Figures

Figure 13:
Filmaffinity - 2001: A Space Odyssey (https://www.filmaffinity.com/es/film171099.html)

Wallpaper Name - 2001 A Space Odyssey HAL9000
(http://www.wallpapername.com/Fictional_characters/HAL/2001_a_space_odyssey_hal9000_2000x1300_wallpaper_51017)

Figure 14:
Flickr - Daniel Semper: ALICE
(https://www.flickr.com/photos/danielsemper/502694453)

Figure 15:
Slide Share - Virtual intelligent student counselor for APIIT
(https://www.slideshare.net/uldusg/virtual-intelligent-student-counselor-for-apiit)

10.3. "Unexpected Visitor's Game Master: The Chatbot Someone Thought About" Bibliography

10.3.1. Global information

Business Of Apps - Discord statistics
(https://www.businessofapps.com/data/discord-statistics/)