



UNIVERSITAT DE
BARCELONA

Treball final de grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques
Universitat de Barcelona

Design and Development of a
Virtual Reality Serious Game with
Editable Content

Autor: Alexander Dickson Roig

Director: Dra. Inmaculada Rodríguez Santiago

Realitzat a: Departament de Matemàtiques i Informàtica

Barcelona, September 14, 2020

Abstract (English)

This project is about the development and design of a Virtual Reality game to go along with the standard educational curriculum and make the activity of learning an even more fun experience. The game will be an escape room with trivia/multi answer questions introduced by the teacher that will condition the players to progress.

The game was built in the Unity game engine using `c#`. A companion web app was built using Javascript and Firebase, to be used by the teachers to upload their open subject questions for the game. The project was built with scalability in mind to make the project a starter for other students so it is highly encouraged that any students on different paths (business, psychology, or other computer science students), adopt this project and build on top it.

Abstract (Català)

Aquest projecte tracta sobre el desenvolupament i el disseny d'un joc de realitat virtual per anar junt amb el currículum educatiu estàndard i fer de l'activitat d'aprenentatge una experiència encara més divertida. El joc serà un escape room amb preguntes de resposta múltiple introduïdes pel professor que condicionarà el progrés dels jugadors.

El joc es va desenvolupar amb el motor de videojoc Unity mitjançant `c#`. També s'ha creat una aplicació web complementària amb Javascript i Firebase, que els professors utilitzaran per carregar les seves preguntes de tema obert per el joc. El projecte s'ha construït tenint en compte l'escalabilitat per fer del projecte un inici per a altres estudiants, de manera que es recomana que qualsevol estudiant en diferents carreres (empreses, psicologia o altres estudiants de Enginyeria Informàtica) adopti aquest projecte i construeixi a sobre.

Abstract (Español)

Este proyecto trata sobre el desarrollo y el diseño de un juego de realidad virtual para acompañar el curriculum educativo estándar y hacer de la actividad de aprendizaje una experiencia aún más divertida. El juego será un escape room con preguntas de respuesta múltiple introducidas por el profesor que condicionará el progreso de los jugadores.

El juego ha sido desarrollado con el motor de videojuego Unity mediante *c#*. También se ha creado una aplicación web complementaria con Javascript y Firebase, que los profesores utilizarán para cargar sus preguntas de tema abierto para el juego. El proyecto se ha construido teniendo en cuenta la escalabilidad para hacer del proyecto un inicio para otros estudiantes, por lo que se recomienda que cualquier estudiante en diferentes carreras (empresas, psicología u otros estudiantes de Ingeniería Informática) adopte este proyecto y construyan encima.

Contents

Index of figures	i
1 Introduction	1
2 Objectives	2
3 Background	3
3.1 Current state of VR	3
3.2 Similar projects	8
3.3 Related technologies	10
3.4 Dismissed technologies	12
4 Analysis	12
4.1 User analysis	12
4.1.1 Player	13
4.1.2 Teacher	14
4.2 Use cases	15
4.3 Use case descriptions	16
5 Game Design	17
5.1 Game description and interaction	17
5.1.1 Game concept	17
5.1.2 Objective	17
5.1.3 Gameplay	18
5.2 Player VR interaction	23
5.3 Class diagram	25

5.4	Class descriptions	27
6	Implementation	29
6.1	Architecture	29
6.2	Virtual Reality implementation: XR Interaction Toolkit	30
6.3	Interaction	30
6.4	Canvas interaction	31
6.5	Teacher website	32
6.6	Client side	32
6.7	Server side	33
6.8	3D modeling	36
7	Conclusions	37
8	Future work	37
9	References	39
10	Appendix I - Technical and user guide	41
10.1	Teacher	41
10.2	Player	41
10.3	Developers	42
10.3.1	VR	42
10.3.2	Firestore	42

Index of figures

1	Game preview	1
2	Virtualizer Elite	4
3	Screen door effect	5
4	Device manufacturer market share	5
5	Oculus quest headset	6
6	Kahoot player screen	8
7	Player picking part in Friday the 13th	9
8	Player doing mini game to place the part in the car	9
9	Player being chased by an enemy	9
10	Alex Wang persona	13
11	Ellie Smith persona	14
12	Use case diagram	15
13	Bank structure	17
14	Start of the game	18
15	One of the parts needed to escape through the door: luggage	18
16	Car exit and needed parts	19
17	Main door exit and needed parts	19
18	Helicopter exit and needed parts	20
19	One of the counters: a vent	21
20	Question canvas	21
21	Button appears when the 4 parts are obtained	22
22	Keyboard input	23
23	Locomotion as teleportation	24
24	Class diagram simplified	25

25	Class diagram extended	26
26	Client-Server architecture	29
27	Canvas preview	31
28	Website preview	32
29	Firebase tree like structure	33
30	Parsing flow diagram	34
31	Question download flow diagram	35
32	Steam workshop	38
33	Changing Unity version	42
34	Firebase values	43

1 Introduction

VR Trivia Escape, the game created in this project, will be a combination of trivia and an escape room game. The experience will be for Oculus Quest and developed in the game engine Unity using C#, with the 3D models acquired either through the Unity Asset Store, the Google Poly website, or made from scratch using Blender.

This game's purpose is to be a complementary activity to help students learn and evaluate themselves while being immersed in a videogame and not only be looking at plane multi-answer questions but to reward correct answers with progress on an escape room. I believe the education system should strive to include more alternative ways to learn and examine oneself other than just the regular means.

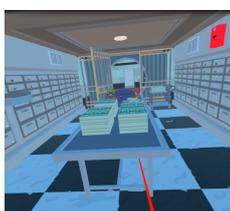


Figure 1: Game preview

One might think that to make a VR game, it is sufficient with just changing the character camera and controls, and maybe like this just adapt any desktop game to VR, but it is not that trivial. VR games come with a lot of complications related to the feeling of comfort of the player e.g. motion sickness and locomotion, related to the basic interaction with objects of the game and related with the HUD (Heads Up Display) and display of information, to name a few.

The motivation for this project was to tackle a part of Computer Science not present in the normal path, which is videogame making. There are a lot of things to learn from making videogames from scratch such as 3D modeling, optimization for good realtime performance, and more than everything, coming up with solutions and alternatives when things get stuck. Also when the idea was presented to me it looked like something I would use myself, a good idea for a game that would potentially be useful and fun to develop, so it was an easy election.

As an end note, this project has been the last step of my degree, where I've applied lots of the knowledge learned in the course of these 4 years, from good programming practices to human computer-interaction and software design.

2 Objectives

The main objective of this project is to develop an application i.e. a VR serious game, directed towards students on their last course of pre-college studies (even though the application will be extendable to other similar profiles). The game should be playable more than once and be a tool for students to reinforce their learning before an exam, so the structure of the game should be random and the placement of the parts also on different places every game, or the game would get repetitive.

Also, a server for uploading questions will be built using Javascript and Firebase, to make the question set modifiable and not be hardcoded in the application, an indispensable feature to make the game extendable to any subject. This website will be used by the subject's tutor to upload custom questions in Aiken[1] format, either by writing them at the moment in a text file following the formats guidelines or downloading them from Moodle. Moodle is the LMS (Learning-Management system) used in lots of institutions, where the questions of their own multi-answer questions tests are also in Aiken format, to make the job of uploading the question set as streamlined as possible. So in this way, they will be able to upload questions related to the content they explained in their class or even to get questions from other teachers from other courses.

A secondary objective is to build specifically a VR application, to jump on the trend and learn an emerging technology. This devices are becoming more and more consumer-friendly and in this way, we'll help the content available on the VR content market grow, which is one of the bigger problems on VR right now. These objectives work can be divided in:

- Competition study
- VR toolkit study
- Game design
- Game implementation
- Website implementation

3 Background

3.1 Current state of VR

We are still experiencing the rapid growth of virtual and augmented reality even though it has not been as steep as in the first stages of its presence on the mainstream market. This growth deacceleration is due to some barriers of adoption that haven't made the VR a must-have.

This technology is in its early stages of design as it is still a product for a niche market greatly because of its price. The current solutions are a big inversion for the day to day consumer (around 600 eur at the time of writing, for a standalone VR headset Oculus Quest) and do not offer any major advantages in relation to other types of products, that would make it a necessity in our lives. To achieve this type of necessity it should provide some functionalities that are only found in XR (Cross Reality i.e. VR and AR) for a use case in someones live, be it entertainment, training or productivity, and it still doesn't accomplish to do so or at least the market is not still aware.

Other factors that push back adoption other than price are, lack of content; the virtual reality stores offer relatively few options compared to other platforms, furthermore, few good titles are big hits and as popular as desktop games, also the size of the headset; the current models are still bulky and complicated and make it difficult to use in places other than a spacious empty room, so portability is not an option, and lastly, the lack of consumer awareness which makes it hard for the technology to show its potential.

Additionally, concerns in videogame communities, include the performance of the headsets on the current market, and the locomotion system, which is still premature and hasn't found a perfect solution for the movement inside the game. Locomotion solutions usually revolve around teleportation abilities (which feel unnatural and "cheaty" and may ruin the immersiveness of the experience or makes the game too static) or moving with a touchpad or a joystick, with is usually seen on desktop and console games, but in the case of virtual reality, it may cause motion sickness when actually stepping in real life or moving the head too fast.

Other new solutions are hardware made to make the player move/walk-in real life on a sort of 360-degree treadmill which obviously incurs an extra cost, even more now that they are small team projects and far from being a mainstream product and more for VR arcades. An example of this Virtualizer Elite for the same price as brand new VR device (See Figure 2).



Figure 2: Virtualizer Elite [20]

Another concerning fact, independently of the device sales, is its engagement on people who own one of these products. According Greenlight Ventures only 28% of VR set owners use them daily and 39% once a week, and in my personal case, the figures would be the same. This would be probably due to the difficulty in setting it up at the start of using it and because of the low variety of content.

Personally, additionally to these concerns I have found the Oculus Quest not really a polished product because of the resolution and the screen door effect (See Figure 5) which in my case was very noticeable, an artifact where you can see the empty black space between pixels due to low pixel density like viewing the world through a grid (displays such as the one present in the Valve Index that has more resolution are reportedly less bound to have this effect).



Figure 3: Screen door effect

However, the things that I took out of the headset experience was its pronounced immersiveness. At first, I thought the user would have to put a lot on his part to feel immersed in the virtual environment, but as soon as I tried the headset myself, on the scene I had been building the previous months, I could really feel that was my new reality and I felt like I wasn't using anything, even with the low poly assets on my project, that should theoretically feel less realistic. So the Oculus Quest has done a really good job with its hardware and even though there are all these obstacles, VR is greatly regarded by consumers and everyone is willing to at least try it, so there is a lot of future for the technology.

When it comes to devices sold by manufacturers, Sony was leading the charts in 2018, with Oculus just behind (See Figure 4).

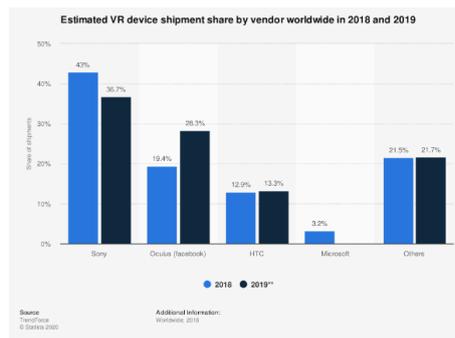


Figure 4: Device manufacturer market share

When it comes to the current year, the number of units sold wasn't so high mostly due to hardware production problems related to the COVID-19 pandemic, but the sales are still outstanding. While Sony got the landmark of 5 million units sold as of January 2020 (compared to 0,9 on February 2017 which is still a lot), Oculus has their most popular devices sold out (Oculus Quest 128GB and 64GB) sold out as of June 2020 and there have also been reports from Valve of having trouble meeting the demand for their higher-end headset Valve Index.

The reason the Oculus Quest was chosen for this project is because of its promising longevity, with Oculus making a bigger bet on the Quest and giving more updates to it than with the Rift. Its standalone capabilities were also a preference (most educational institutions do not have high-end systems to run games on the Rift), and also its low purchasing barrier for students that might not either have good platforms in their homes to run games on virtual reality.



Figure 5: Oculus quest headset

When searching the Oculus store the only educational or serious games are focused on specific topics on a subject and are not general like the one purposed on this project, like for example games to explore the human body, to learn math, or to learn the periodic table. One of the trivia games that exist on the platform is called VR trivia battle but is for purely entertainment content.

On the steam store, also compatible with Oculus headsets, the most similar app to our concept was “It’s quiz time”, a game for PC with a complex AI, and 25 000 questions on different random topics. Market pain reviews were as follows: easy questions, repeated questions, too difficult questions become a matter of guessing instead of knowing. We solve these market pains by making it easy to add questions by the admin, just by going to their portal and downloading any set of questions they already had on previous years, or by writing them in a file with “.txt” extension in any notepad editing software, in an Aiken format, hence there aren’t any similar applications by any means to the concept proposed in this project on the market for VR or desktop applications.

3.2 Similar projects

The following applications inspired some of the ideas of the game. As a lot of ideas are formed by the conjunction of other smaller ideas, this is the applications the concept of my game got born from. Kahoot , is a popular website for hosting quizzes in real-time in class, with the students using their devices to answer, with every student a username which climbs a leaderboard. This application would be the idea that would most resemble the philosophy of our game, VR Trivia Escape, about giving the students a quiz to evaluate them and have fun at the same time.

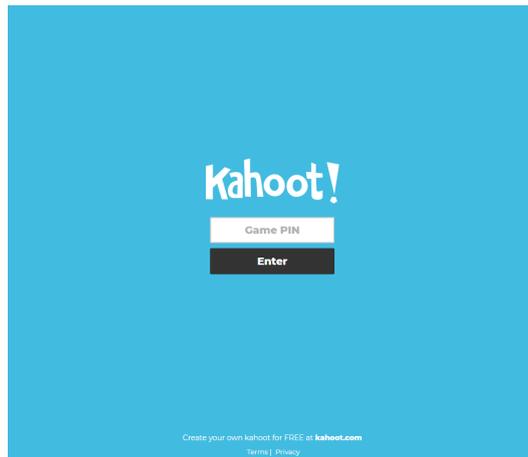


Figure 6: Kahoot player screen

The first idea of the game was a laberint first-person survival game, with the objective of escaping a maze, involving solving riddles whilst being hunted by a monster. Nevertheless, when prototyping a labyrinth with the VR experience, it gave a feeling of being trapped and close to the walls wich was very unpleasant in VR, so the enviroment had to be changed to an escape room. The part of the game about getting different parts to use an exit and mini-games on using objects of interest is taken from a recent multiplayer horror game called Friday the 13th (See Figure 7). In this game, players are assigned roles: either human or Jason, the killer. The humans have 3 inventory slots and can carry either objects to fight Jason, or components to find an exit. They can exit by car, boat or by calling the police. To place each part on the exit mean, they have to do small reaction minigames (See Figure. 8) while risking getting caught by Jason. The result of the game is mesured in how many kills Jason got and if each individual player has escaped.



Figure 7: Player picking part in Friday the 13th



Figure 8: Player doing mini game to place the part in the car



Figure 9: Player being chased by an enemy

3.3 Related technologies

This section describes the different languages, external libraries, and technologies that have been used to develop both the game and the web application.

Unity

Unity is a game engine to develop 2D and 3D games, simulations, animations, etc.. for a large variety of platforms such as WebGL, Desktop PC, Android, IOS, SmartTV, Consoles, and Virtual Reality platforms.

C#

Programming language derived from the C programming language family with similarities to Java and object-oriented. It was developed by Microsoft as part of its .NET initiative and was adopted to use as the language in unity to write scripts to define behaviors in Unity game objects.

HTML, CSS, JAVASCRIPT

The basic web programming stack is used. HTML to build the skeleton of the website i.e. what elements will form the website and where they go, CSS to style these elements or hide/show them, and Javascript to program the behavior of the components and access the database service.

Firebase

Firebase is a platform from Google integrated with the other existing platform Google Cloud Platform. Firebase is used to develop web and mobile applications offering tools like cloud messaging, user authentication, real-time database, or file storage for free (with limited use). It was also used to host the website as it provides a new hosting service for free. It is compatible with IOS, Android, Web applications, C++, and Unity.

Github

Github is a hosting service for git repositories. It is used to share, store, collaborate on, keep track of, and test experimental code. I used Github to store the code of both the website and the game [22][7].

3.4 Dismissed technologies

Unreal

Unreal is also a game engine, developed by Epic Games. It is written in C++ and it is open-source so it lets the user modify the base code to improve it. It is arguably a more powerful and complicated engine than Unity, and it always has stood up for its photorealistic environments. But with the addition of new addons in Unity like the HDRP(High Definition Render Pipeline) asset, it is getting more difficult to set them apart appearance-based, and both engines produce similar results.

The election of Unity over Unreal was simply because of its learning curve, related to both the editor and the programming language (Unreal uses C++). Also, Unity's presence online is way larger and there is a lot of great tutorials and forums.

4 Analysis

4.1 User analysis

There are 2 types of users in this system: the player or student and the teacher or instructor.

We have to define the characteristics of the player and the instructor to discover cues to engage the player in the game or to know what reasons would make the instructor want to try using the tool in class. We will be creating user personas to represent a group of people and define the users' characteristics and behaviors, so we can extract new information from their profile and build empathy with them.

4.1.1 Player

The person in this user group is a student between 10 years old and 18. These bounds start from the first age a student would be independent enough to be able to use the game correctly to an age where a student is still greatly interested in videogames, as older people are more resilient in trying new games. Also the general esthetic is important, as it is for an immature audience that likes colorful environments and low poly/pastel models.

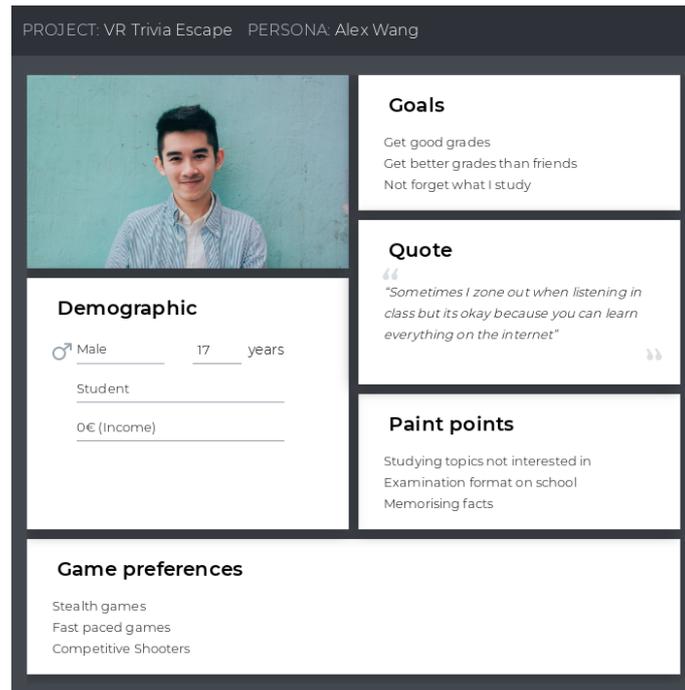


Figure 10: Alex Wang persona

Alex is a highschool student that is smart and motivated but gets bored easily. He enjoys playing videogames for its competitiveness and likes fast paced games that make him make decisions rapidly and get his blood pumping. This created a big disparity to what he gets in school. He feels like the classes are too slow, as they have to go at everyone's pace, and once he gets distracted, he gets lost and it is difficult to get back on track. He doesn't like the conventional educational system and thinks there are better ways to study subjects in school. When studying he often competes with his classmates to see who answers the questions of previous exams better, to have an incentive to study for the exam.

4.1.2 Teacher

The teachers that would use this game in their class would be someone between the ages of 25-45. This bounds start at the age of someone who has somewhat experience in teaching, and thinks outside the box on different ways of teaching. The bounds end on an age that a person has already cemented their practices and I think it would be unlikely that they would be willing to try new things. The teacher should also be employed on a institution with a high budget, depending on the number of devices they would like to use in class, as Oculus Quests can be very expensive.

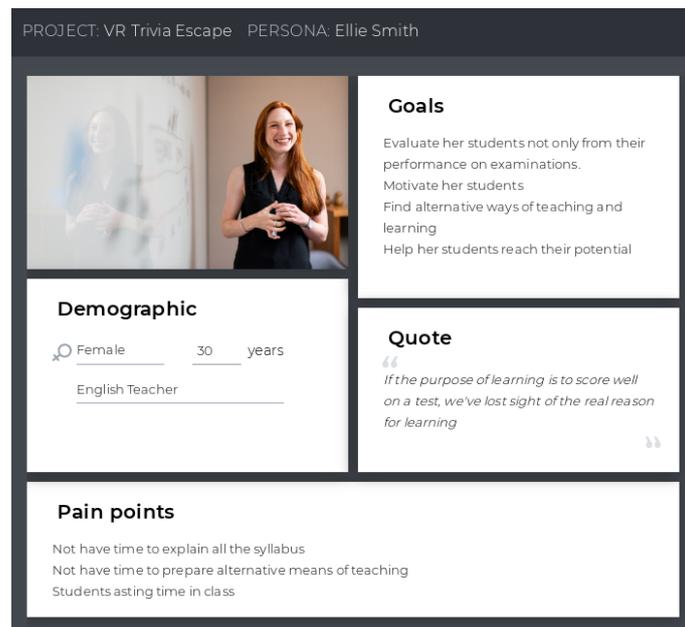


Figure 11: Ellie Smith persona

Ellie is a teacher around her 30's. She is energetic, enthusiastic and empathetic with her students. She is also somewhat experienced with technology and she likes to use it to work more efficiently. She has tried lots of alternative ways of teaching, some better than others, and she often rewards good performance in the games she organizes with some points in the next exam, to motivate them to do well. She values more understanding than memorising.

4.2 Use cases

In the following Use Cases Diagram I'll show the main use cases.

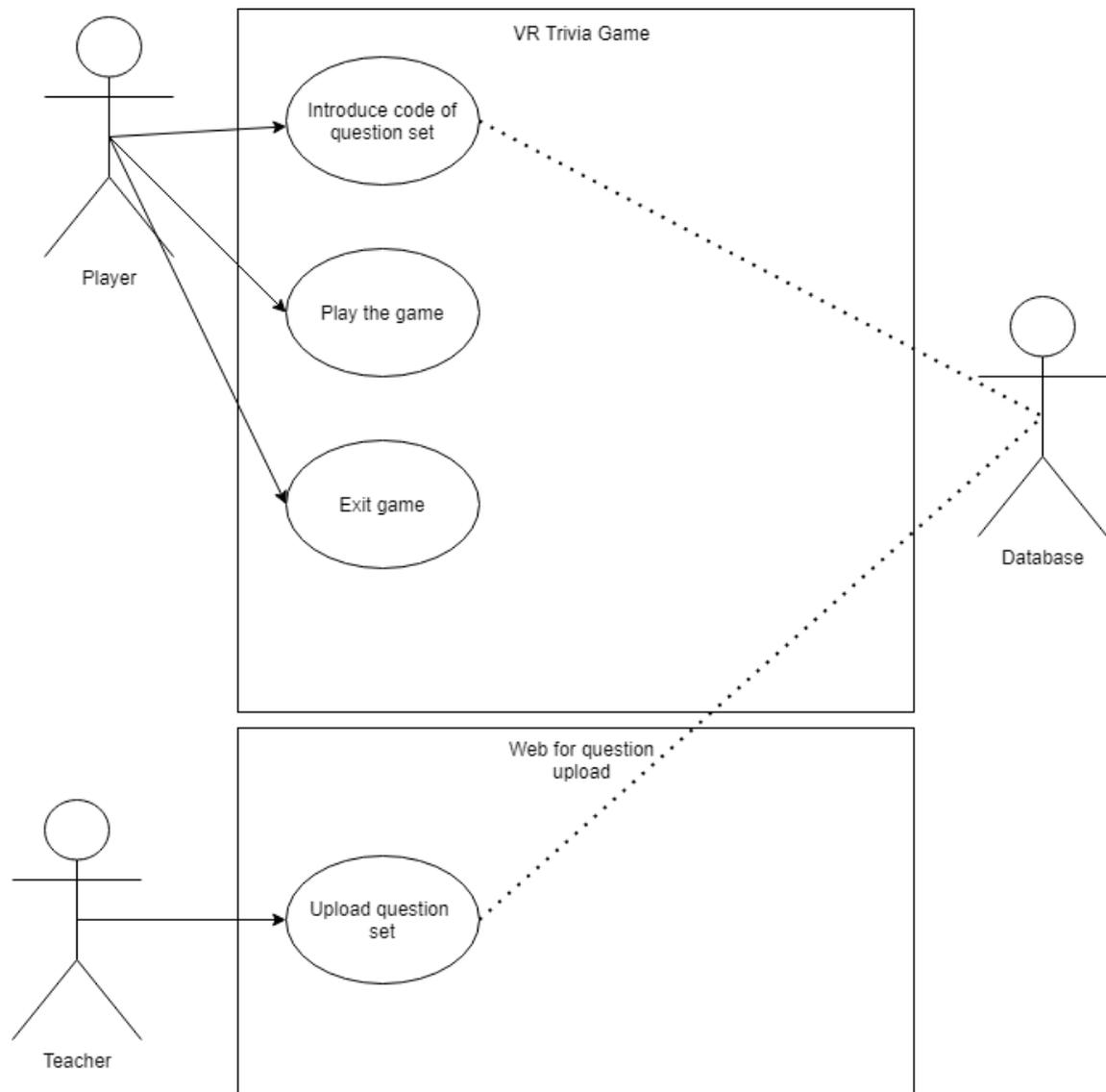


Figure 12: Use case diagram

4.3 Use case descriptions

Name:	UC1 - Set new questions
Actor:	Player
Description:	Player introduces the id and sets new questions set
Preconditions:	System must be connected to a network
Basic flow:	<ol style="list-style-type: none"> 1. The user selects the input field 2. The system displays the virtual keyboard 3. The Player introduces question id and presses the Accept button 4. The question set is downloaded correctly and the system displays success message on screen
Alternative flow:	3a. The id does not exist so the question set is not downloaded, then the system displays error on screen
Post-conditions:	<ol style="list-style-type: none"> 1. Play button is enabled and question set is saved in the questions class

Name:	UC4- Upload question set
Actor:	Teacher
Description:	Teaches uploads question file to databse
Preconditions:	System must be connected to a network
Basic flow:	<ol style="list-style-type: none"> 1. The teacher presses the "Choose question file" button 2. The system displays the file explorer pop up 3. The teacher chooses the correct .txt file 4. The teacher introduces unique id for his question set 5. The teacher presses the "Upload" button 6. The system shows success message
Alternative flow:	<ol style="list-style-type: none"> 1a. The teacher drags the file to the open website 4a. The file id is not unique, so the system displays error on screen 6a. The file has less than 12 questions, so the system displays error on screen 6b. The file has a question with more than 4 answers, so the system displays error on screen
Post-conditions:	<ol style="list-style-type: none"> 1. Question set uploaded to Firebase with introduced id

5 Game Design

5.1 Game description and interaction

5.1.1 Game concept

VR Trivia Escape, is a trivia game based on multiple choice answers, to be used by students, to evaluate their knowledge on a certain topic, and hopefully learn new facts from their mistakes. The focus of the game is not to actively teach the player from scratch, it is a tool to learn the answer to specific questions or train for an exam.

5.1.2 Objective

The objective of the game is to escape from a predefined structure (See Figure 13), in this case with the thematic of a financial bank, where the player will be acting as a bank robber who is trying to escape before being caught by a sentient enemy dressed as a police officer.

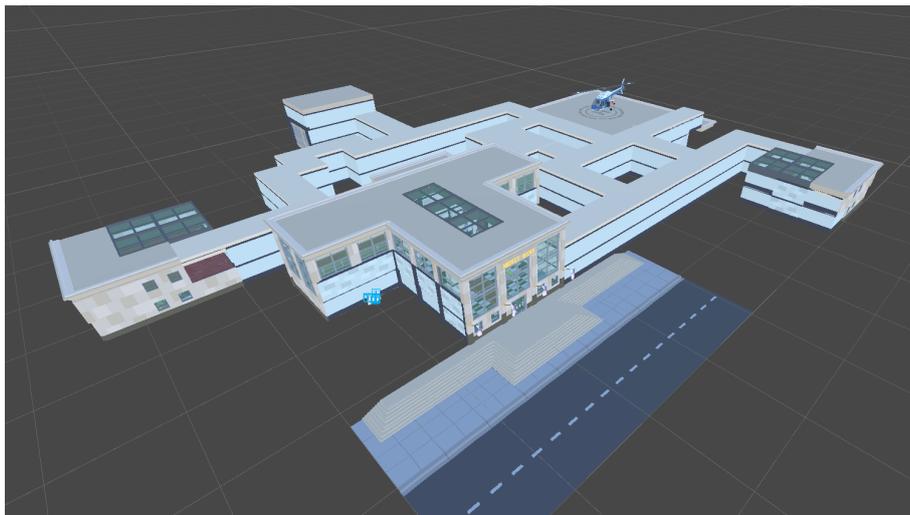


Figure 13: Bank structure

5.1.3 Gameplay

The story of the game takes place inside the fictional bank where the player finds himself inside, after supposedly committing a bank heist. The player (or thief) gains consciousness the moment he has already committed the crime and broken inside the vault and taken the money. The player can hear the alarm blaring and has to escape from the facility as soon as possible, without being caught by the police, which has already entered the bank and is already in his pursuit.

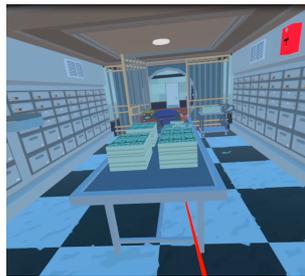


Figure 14: Start of the game

The player is alone and has three options to escape, either with a helicopter located on the opposite side of the bank, a police van left on the bank's garage, or by the main entrance door. To use any of these means to escape he first has to find additional parts (See Figure 15) to unblock these exits, as in a real-life scenario, where you wouldn't be able to use a car without gas or just exit the main door without the key.

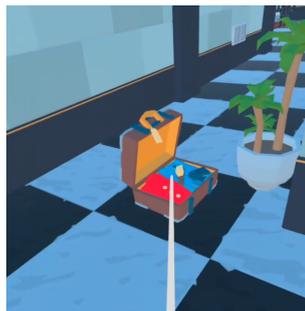


Figure 15: One of the parts needed to escape through the door: luggage

For instance, if the robber wants to escape with the car, he first has to find the *car keys* to turn the engine on, a *battery* to replace the old one to have power for the car, *gas*[8] for the combustible, and some *wire cutters* to set the alarm of the garage off.(See Figure 16)



Figure 16: Car exit and needed parts

In the case of the main door, the player will have to disguise as a normal citizen or hostage. He will have to find a *fake id*[6] in case the police stop him, a *mask* consisting of fake glasses and a mustache (yes, the police in this world are not really smart), some *fake luggage*[14], and a *door key*[5].(See Figure 17)

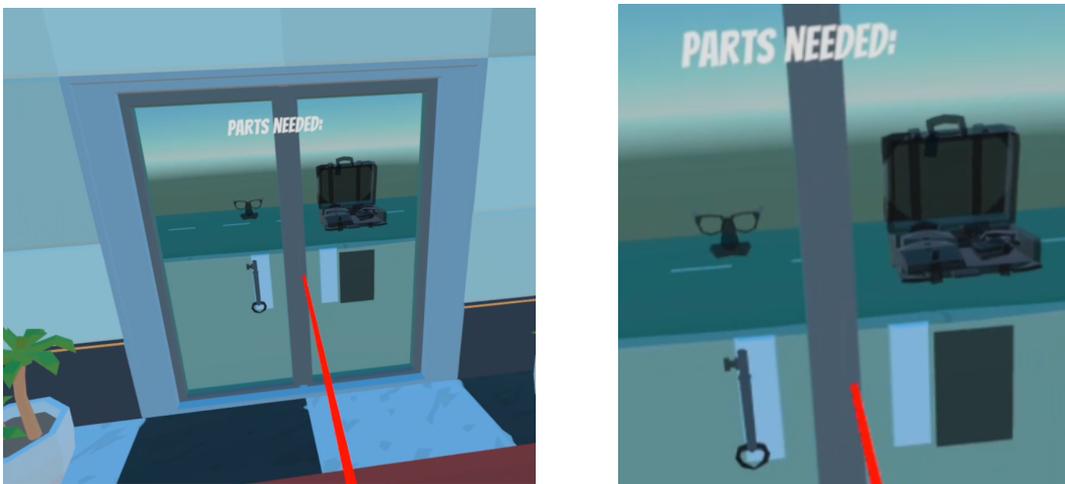


Figure 17: Main door exit and needed parts

Lastly, if he wants to escape with a helicopter, he first has to find a *crowbar* to force open the helicopter door, a “*How to drive a helicopter*” *manual*[4] as not everyone knows how to drive a helicopter, a *parachute* [2] inside a red bag with a symbol of a parachute in case the robber doesn’t fully grasp all the contents of the manual in time and *helicopter gear*[9][3][11] i.e. glasses, a cap, and some sound muffling earphones.(See Figure 18)

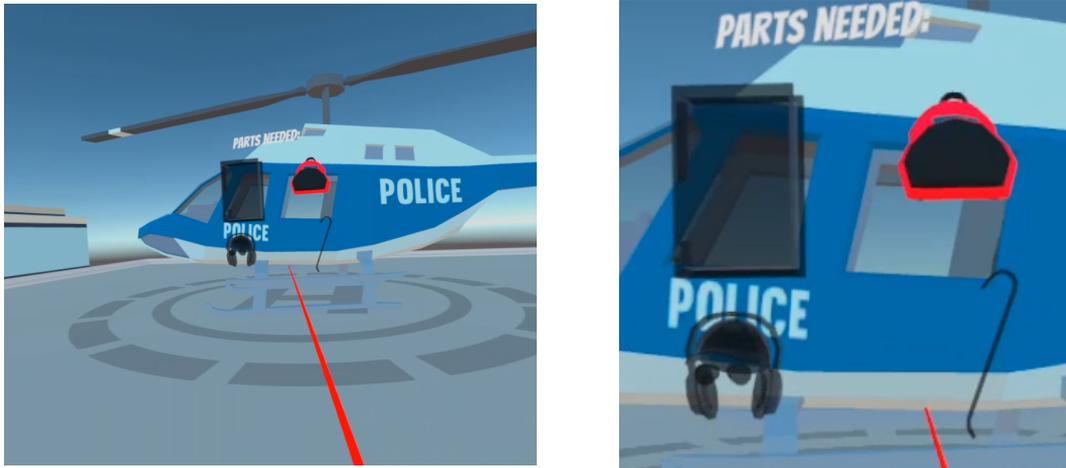


Figure 18: Helicopter exit and needed parts

So as the player can see, the parts make sense (some might not be obvious such as the wire cutters, in the case of the car, to set the alarm off) but he can go to each exit, once he finds it, and find a display with hints to what shape the required parts he has to find have, and what ones he has at that moment, already present and colored in the display (See gas tank in figure 16).

The enemy will actively be following the player with a pathfinding algorithm (it is not explained in this document because its the trivial pathfinding algorithm provided by Unity), so the player will have to constantly be aware of their surroundings and use strategies to fight or escape from the enemy which will try to reach the player and touch him (then we will assume the player will have been arrested).

To aid the player's fight against the enemies the player will have to use the so-called counterstrategy objects, randomly placed on the floor through-out the map, and once interacted with, will give a benefit to the player by delaying, distracting, or hiding from the enemy.(See Figure 19) In this case, there will be:

- Vents to teleport to another of the available rooms of the building with a paired vent
- Lockers to hide inside and make the enemy lose track of the player.[13]

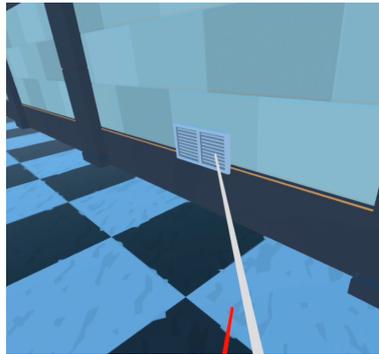


Figure 19: One of the counters: a vent

The acquisition of the escape components and the success of using the enemy counter objects such as the lockers to hide will be determined based on the outcome of one of the trivia questions that will appear near the object on a 2D canvas, on a timed constraint. (now 60 seconds). (See Figure 20)



Figure 20: Question canvas

- If the player answers the question correctly he will be awarded the part or the counter object temporary benefit.
- If the player answers the question incorrectly or the time runs out he will not be awarded the part and the respective exit that needs the part will be unobtainable, so he will have to resort to a different escape.
- In case of using a counter object, if he misses the question or there is no time left, the penalization will not be as harsh, they will be able to try multiple times until they get it right.

Once the player has obtained the 4 parts of an exit a button to exit will appear and he will win the game. If the player is caught before pressing the button, the game will be over. (See Figure 21). You can see the full gameplay on youtube [19].



Figure 21: Button appears when the 4 parts are obtained

5.2 Player VR interaction

The game starts with a game menu with a play button, and a button and input field to add a new question set. The play button first starts as disabled until a valid set of questions is loaded into the game instance.



Figure 22: Keyboard input

The player will use the controllers orientation to navigate the menu which will have a red ray to indicate where the controlleres are pointing. Once the controller is hovering a valid option, he will be able to press the trigger button (explained in the instructions where it is) to select objects. To load a new question set, the student or player will have to introduce the unique code the teacher introduced when uploading the questions to the question set server. To introduce this id, he will have to hover and press the input field, which will pop up a keyboard that is controlled using the head orientation, with a crosshair in the center of it. (See Figure 22).

To then select a letter, the player will have to remain more than 3 seconds inside the bounds of one of the keys of the keyboard, he will see the progress of these 3 seconds with a red loading circle around the crosshair. To finish the input text, the user will have to press the enter key using their gaze as with the other keys. Then if the question set is valid the play button will be enabled and he will be able to switch scene to the gameplay scene. The 2 sets of buttons are separated by horizontal space to give the user a distinction of steps, and if the player tries to press the disabled button, this will not react, so the process should be pretty straightforward.

The keyboard is imported from a plugin in the asset store (no standard VR keyboard is provided by unity)[12]

The mechanics once inside the game are simple. They consist of movement around the

bank with only one speed using the left controller's joystick, camera rotation using the headset orientation, interaction with the VR controllers orientation, and the trigger button, on the back of every controller, that will be used to interact with doors, objects, and canvas elements.

In regards to the locomotion of the game (the way the player moves) the only solution was to use continuous movement provided by the joysticks on the controller. It had to be continuous to make sense in a game where the player is being chased and has to escape, as for instance teleportation (See Figure 23) is not an option since the player could just teleport past the enemy if they felt cornered or teleport repeatedly and move a lot faster than the enemy.

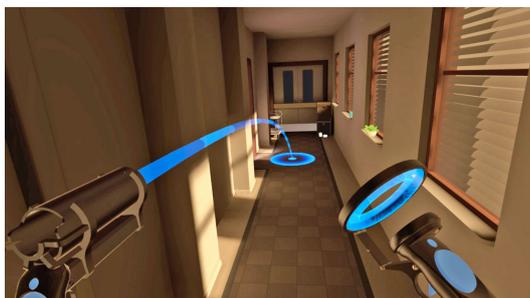


Figure 23: Locomotion as teleportation

The drawbacks of locomotion using the joystick are that it can produce a loss of balance/motion sickness and also immersiveness takes a pretty good toll. The reason why there is a problem with moving the player with a joystick is that it produces some sort of disparity between what the eyes of the player sees i.e. moving in space, and what the legs tell the brain it is doing i.e. not moving. So to eliminate this disparity, the brain then stops listening to the legs for movement, distancing itself with the movement and the hand takes control i.e. the joystick on the controller.

This is what makes it lose immersiveness, the brain ignoring its natural means of movement and using an artificial way to move: a plastic stick controlled by the fingers.

If the player does move in real life for instance when rotating to change the orientation of the body rotating our feet (we have to do this because our neck has only 180 degrees of freedom), if we are moving inside the game while doing this, the disparity comes back and this is what produces loss of balance and even major sensations of disorientation for a short period of time.

5.3 Class diagram

This section presents the class diagram with the most relevant classes, methods and properties. It is recommended to zoom in either by the controls of your pdf viewer or ctrl,+ or -, or see the original diagram in the project code: file “ClassDiagram1.cd”.

Disclaimer: the classes that appear are only c# classes done from scratch and applied as components to game objects from the scene, not classes provided from unity such as the XR Rig or the camera component. Also, some of the relationships are not shown because are indirect but exist for example in cases where one property of a class also happens to have as a component another class from the diagram. Some of the variables and functions of some classes are hidden because they are not really important such as the case of the “MovementProvider.cs”. Below, the class diagram with its inner parts hidden (See Figure 24), and on the next page, with the parts shown (See Figure 25).

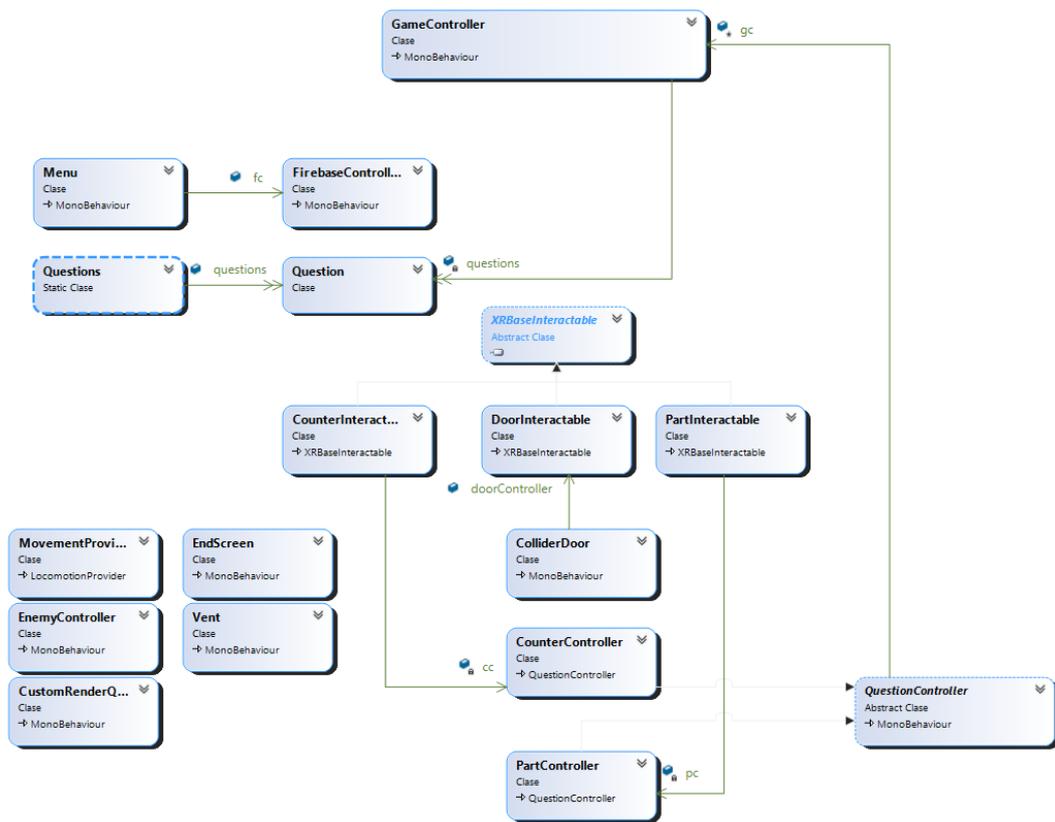


Figure 24: Class diagram simplified

5.4 Class descriptions

Some of the classes are briefly described in the following section and separated depending on the scene they are used on

Start menu scene

-**Menu.cs**: Class with the responsibility of controlling the canvas of the menu screen in the scene

-**FirestoreController.cs**: Class with the responsibility of fetching questions from the database. If they are in the database it means the format has already been checked so the only error-prone to happen is the id does not exist.

-**Questions.cs**: Static class that is accessible from the whole application that holds the set of questions of the game. The reason it is static is that then it can be accessed without losing its values from different scenes (when changing scenes everything is destroyed). If for whatever reason no question set is loaded, for example when testing scenes without having to go through all the steps (when debugging the game with some scenes deactivated) or if some sort of error occurs, a placeholder set of questions is returned.

-**Question.cs**: Container class that holds the information for a question i.e. the inquiry, options, and correct answer.

Game scene

-**GameController.cs**: Class that takes care of initializing the objects in the scene and managing the progress of the player.

-**Questions.cs and question.cs**: Explained above

-**Enemy controller.cs**: Class responsible for the AI of the enemy i.e. chasing the player, checking collisions and if the enemy is colliding with the player, end the game.

-**Question controller.cs**: Class for all the objects that have a question to use them. This class takes care of populating the canvas, setting the timer, and checking if the answer is correct.

-**Counter controller.cs**: Class for counter objects only such as vents and lockers. This class is an extension of questioncontroller.cs above and defines what happens when a question has been passed and the object is a counter. In case it is a vent it will teleport

the player to the paired vent and in case it is a locker, it will hide the player inside of it during a certain amount of time that will pass in a coroutine.

-Part controller.cs: Class for unlockable parts for the exit means. Also an extension of the questioncontroller.cs class and defines what happens when a question is answered correctly when you click a part, in this case, delegate the handling to the game controller which will process the type of object and then destroy it.

-Part interactable.cs and Counter Interactable.cs: Class extended from XRBaseInteractable that defines what happens when an object has interacted with the VR controller. In this case, when the part is selected i.e. hovered and while hovering pressing the trigger button, in this case calling the respective controller and telling it to start displaying the canvas.

-DoorInteractable.cs: Class that extends XRBaseInteractable and defines what happens when the door is hovered and then selected. If the door is open it closes it, if closed it opens it. It also handles the opening of the door in case the enemy collides with its bounding area and it opens it in case it is closed, so he can go through the doorway.

-DoorController.cs: Class with oncollide definition for the invisible bounds collider on the doors that will trigger when the enemy is close to the door. This is to open the doors for the enemy and the bounds are big to give him time to go through the doorway and open the door.

-Vent.cs: Class that holds the information about the paired vent.

-MovementProvider.cs: The XR Interaction Toolkit doesn't provide the movement with the joystick implemented, this class implements it. Written following a tutorial from youtube user VR with Andrew [21].

-CustomRenderQueue.cs: Class in charge of changing the material to draw a game object on top of everything in the z-buffer. Used in the canvas for example.

Scene EndScreen and EndScreenGameOver

-EndScreen.cs: Class that controls the menu of the end screen after winning/losing a game.

6 Implementation

6.1 Architecture

In Unity, the only design pattern is CB (Component-Based), where each component is an independent system that takes part in a bigger role. In Unity, the game and even the menu's are all contained in scenes. Then every scene has its game objects which are equivalent to objects in the game like for example enemy's, walls, lights, cameras, and even special effects.

To give functionality to a `GameObject`, components have to be attached to this game object, as in the CB pattern. These components can be already in Unity or can be created from scratch as `c#` scripts. An example of a component would be a script controlling the movement of the player based on the keys pressed, and then we would attach this script to the player model in the scene. So in Unity, we will find the case where multiple components control the behavior of only one game object. There are some exceptions though, as, in this project where there is an empty game object (with no model) called *GameController*, that is only a container for a script to manage the game, like for example spawning all the parts of the scene in random places or changing between scenes. This sort of game object is a pretty extended practice between Unity programming.

A client-server architecture was also used to build the website to upload questions. The parts involved are the game, the website and the database. (See Figure 26)

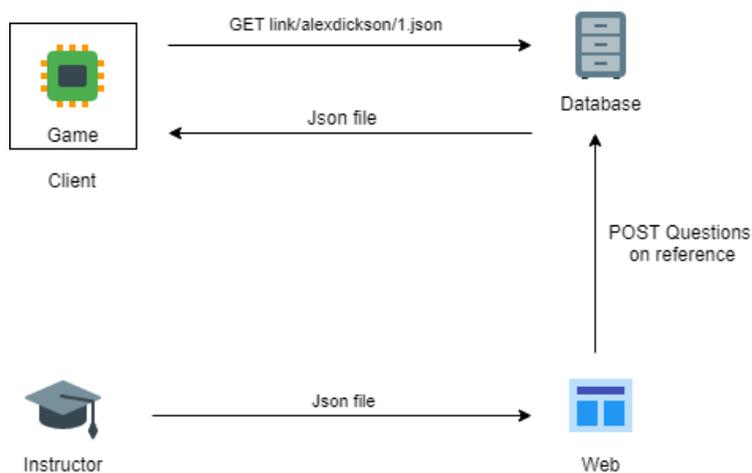


Figure 26: Client-Server architecture

6.2 Virtual Reality implementation: XR Interaction Toolkit

When developing for the Oculus Quest there are 2 options: OVR package from Oculus which is for Oculus devices only and the new XR Interaction Toolkit for versions 2019.3 and later, just released as a preview by Unity but fully backed by them and the state of the art for VR development from now on. In this project I used the XR Interaction Toolkit.

This package is high-level so it supports cross-platform XR controller input i.e. HTC Vive Controllers at the same time as Oculus and other hardware available, which is perfect for our requirement of making the game as widely available and general as possible. The package is a series of components to be attached to game objects to be able to interact with VR, and some game objects like the XRRig, containing the camera, a camera offset, and 2 controllers, and finally the XR interaction manager game object.

To be able to use this plugin containing the XR Interaction Toolkit, you must install it via the packet manager under “Window” > “Packet Manager”.

6.3 Interaction

On the objects that need interaction with the VR controllers, we will add a script that extends from *XRBaseInteractable*. This gives us a lot of functions from events to implement, and define what they should do. These functions are *onHoverEnter* and *onHoverExit*, *onSelectEnter* and *onSelectExit*, *onActivate* and *onDeactivate*. Then on the *ProcessInteractable* function, we can check what events have been triggered and do some process. It is basically the `update`¹ function for interactables.

Once you implement all the handlers you have to attach the script to the object and attach an XR Interaction Manager. This is one of the bad parts of using the new toolkit, sometimes you have to attach components that are trivial manually that without them nothing will work.

¹Update is the function that is called every frame in game objects extending monobehaviour that defines behaviour that changes over time

6.4 Canvas interaction

A canvas is where UI elements are drawn in videogames (See Figure 27). To use a canvas in VR you create the canvas and canvas components, as if you would be developing the canvas for a desktop game. The only difference is you have to set the canvas render mode to World Space, because a fixed canvas on-screen space like a HUD, is a bad practice for VR as it can get annoying to have something stick to your face wherever you move.

Additionally, to change this setting to view the canvas correctly, you have to set a component called “Tracked device graphic raycaster” to be able to interact with the canvas with the controllers. Also, do not forget to have an EventSystem in the scene or nothing will work.

To display the canvas in front of everything (the canvas appears facing the player’s orientation, regardless of what is in front) we use the script mentioned above called “Custom Render Queue”. This changes the shader to draw the element (text or image it is attached to) in top of everything by changing the priority in the Z-buffer (in charge of calculating what triangles to render on top of which).

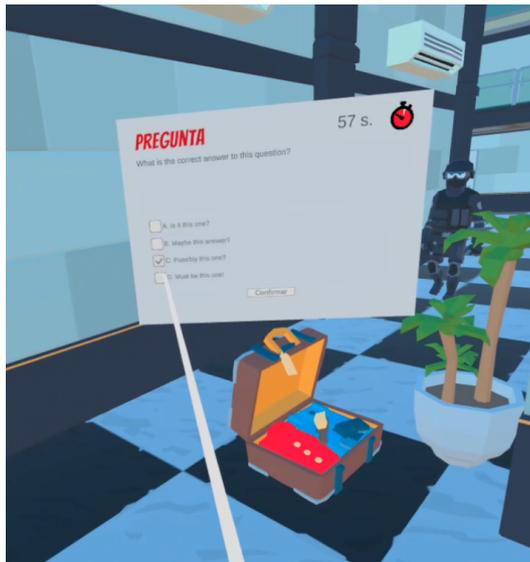


Figure 27: Canvas preview

6.5 Teacher website

6.6 Client side

As introduced in section 1, the website is where the teachers will access to upload the question set in a txt format. It was the only way to have a custom set of questions in the game because you obviously can't choose or drag file once inside the virtual reality game, so a separation had to be made. You can access the website at:

<https://vrtrivia-619c6.web.app/>

UNIVERSITAT DE BARCELONA

UB VR TRIVIA Questions upload database

1. Choose or drag your questions file (all questions in one file) in a aiken format (max.4 answers per question, minimum 12 questions to play and minimum 15 for best game experience) with the blue button.
2. Write a unique name for your question file that will be used to find the question set ingame.
3. Once step 1 is complete, the Upload button will activate, press it to upload the questions to the database.
4. Done!

NO numbering.

What is the Capital of Canada?
A. Ottawa
B. Vancouver
C. Toronto
D. Montreal
ANSWER: A

One line space.

What is the Capital of Alberta?
A. Ottawa
B. Calgary
C. Edmonton
D. Lethbridge
ANSWER: C

All capitalized.

1. No file chosen, yet.

2. Id for the questions: 3.

Figure 28: Website preview

The skeleton of the website is built using the regular stack of HTML, CSS, and javascript. The website consists of 2 buttons, one for choosing a file from the local file system, and an upload button, disabled until a file has been chosen (see Figure 28 or visit <https://vrtrivia-619c6.web.app/>) The whole page also has a dropbox to let the user drag the file directly inside the window. Also, some instructions are on the page, about the minimum number of questions (12) and the maximum number of answers (4) per question. (see Figure 28 or visit <https://vrtrivia-619c6.web.app/>)

6.7 Server side

The election of this type of database was pretty simple. Firebase has support for unity, and also the format they save data because it is a NoSQL type database with a JSON tree-like structure to save the data with matches with our type of data: questions and answers in text format (See Figure 29).

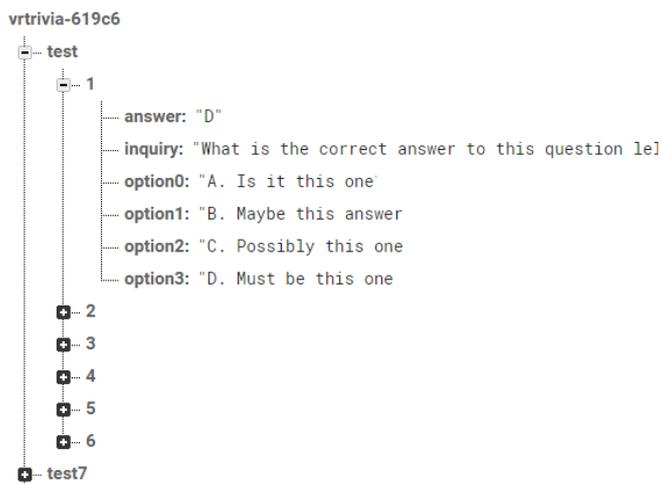


Figure 29: Firebase tree like structure

To parse the file uploaded, it is done on the client (website) side in javascript. We use the following parsing algorithm:(See Figure. 30)

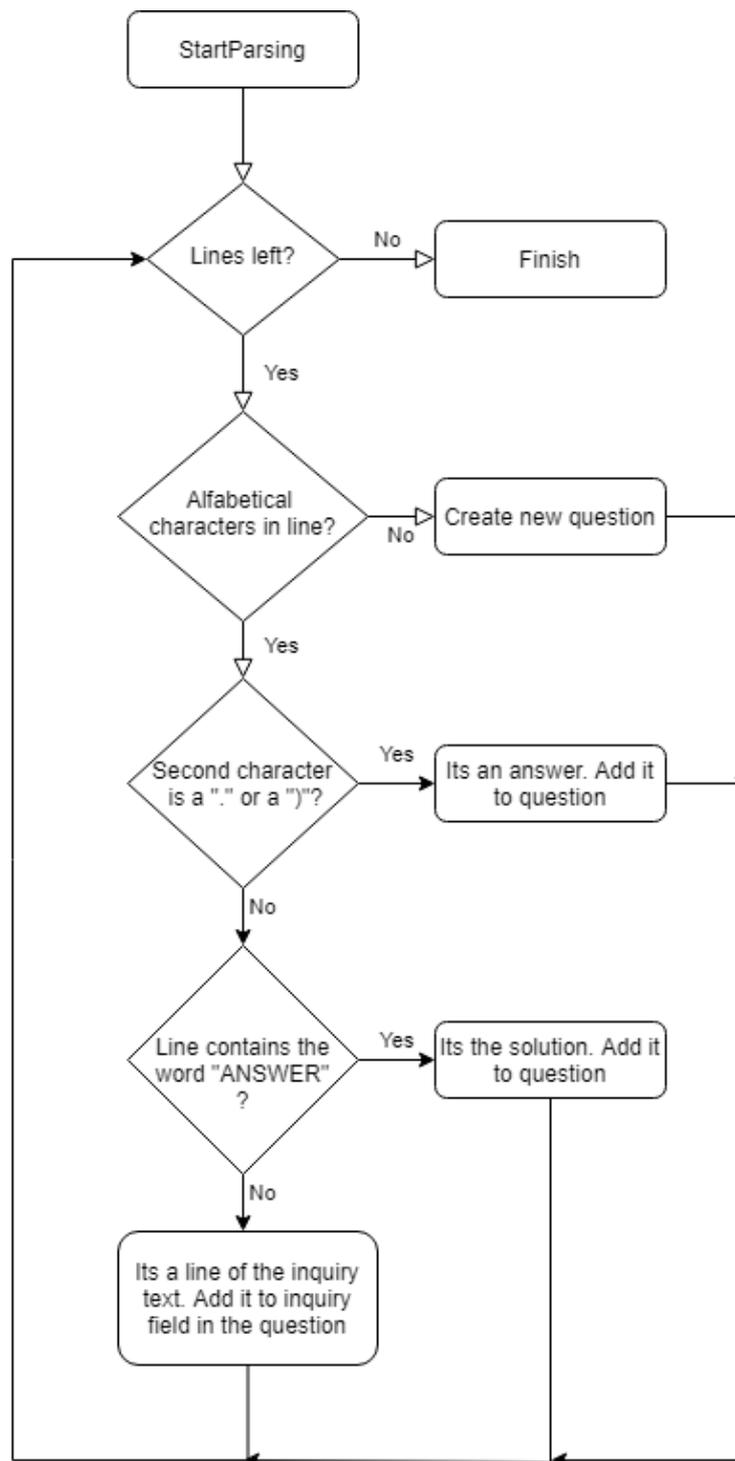


Figure 30: Parsing flow diagram
34

To manage the communication with the database from Unity, we will not be using the Firebase SDK for Unity as it is in beta version, limited to test builds and buggy on unity version 2019.3. So we will be using a user plugin that will access the REST API of the Firebase service, it is called Rest Client for Unity[16].

To access the API, we use the RestClient class and a GET request with the type of data we want to access. This will only work if the variables in the Question class are named the same as the keys of the data in Firebase, so in this way, it parses the response directly to our desired class.

We then define the handler for when a question has been successfully downloaded and the handler for when there isn't a response from the database (because of a wrong id or no more questions left, (See Figure 31).

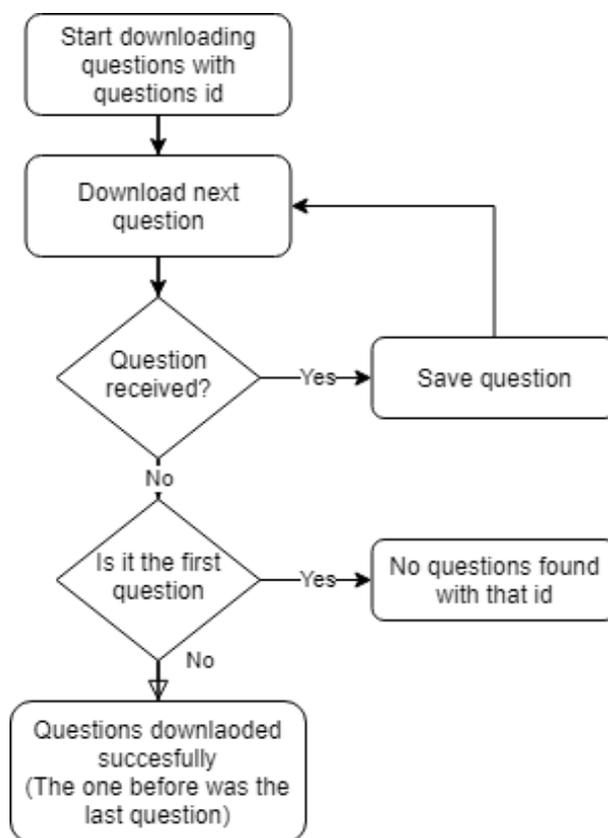


Figure 31: Question download flow diagram

6.8 3D modeling

One of the top difficulties when developing a game without a team is finding all the resources needed in terms of graphic modeling, in this case, 3D models. There are two ways to get a model you need, you either purchase it from 3rd parties like the Unity Asset Store , or you design it yourself with 3D modeling software. The problem with the first method is that not always you will find the a model you want or the models available don't fit with your game's aesthetics, are not the quality expected or are too expensive.

So the only alternative is to either contract someone to model it for you or to modify the game to no longer need that type of object (say for example you need a model of a truck, you don't find it so you change all trucks for vans). But sometimes sacrificing an idea like that is impossible, so 3D modeling experience is a must-have if there is a limited budget.

The third-party store I used to find my models were the Unity Asset Store [18] for big model packs and Google Poly for individual objects. In the Unity Asset Store, I imported the Bank Heist pack from Synty Studios[17] that contained the whole bank environment, some props, 3D characters, for instance, the one used for the enemy, and the vehicles. Google Poly[10] is a not very known site that is perfect to find assets, and I used it to get the majority of the parts of the escape room. The models are always free but with a CC-BY license where the author has to be mentioned (the name of the models I used are in the references page at the end of the document).

You just have to search a name of an object and once you find the model that suits you, you just download it with the right format, in our case for Unity, in .fbx format. If the fbx extension is not available you can still download it as a .obj file, open it with your preference on 3D modeling software and then export it to Unity as a fbx file.

For this project, I also learned how to use the basics of Blender. I learned how to navigate the scene, how to move (grab) objects, edges, points and faces and basic operations like extruding faces or beveling, and applying modifiers like differences or the mirror modifier. With only these operations you can go a long way. For instance, I couldn't find some models for some part in either the Asset Store or Google Poly like this key below (See Figure. ??), and I modeled it as 2 objects, the key, and the ring, and then colored it using UV mapping (in the right) by separating parts of the mesh in vertex groups and then dragging the vertex groups in the appropriate square of the UV mapping.

7 Conclusions

Looking back at the start of the project, there is a big difference in what I thought this project would be. Most of the objectives were achieved, but it took more time and planning than I first thought, something that often happens with software developers. We should always think everything will take double the time you expect to give margin to errors and our optimism. The only objectives that were not achieved were making the structure of the game procedurally generated due to some unexpected behavior with the collisions as explained above and posting the game on the Quest store (you have to present a submission to be reviewed by the Oculus team that is out of the scope of the application) so I just posted the apk on the client website. I am still satisfied with the results of the project though, and I think it has a lot of potential.

8 Future work

This project was built with its evolution in mind. It would be a nice followup to build different genres and stories for escape rooms, such as haunted house themes, a world of illusions themes, escape from an island, or a prison escape theme, to give some ideas, so the player could choose between different themes based on their personality or the themes already played. It could also be interesting to add new enemy agents and change their behavior with new Artificial Intelligence, and new game features like wildcards inside questions (like the ones in the trivia game "Preguntados" [15])

The first plan for the application was to have a randomly generated structure every game to make the game less repetitive and memorable. This would make the players maybe play more times. The algorithm used to generate a set of rooms and the corridors that connected them worked as follows: it started with the first room and iterated through the rooms left and saved its doorways. Then in each doorway, it tried to concatenate a room only if it could fit or in other words, it didn't collide with other rooms. But there was a problem with collisions not registering or registering in cases it shouldn't and it took more than 2 weeks so I had to resort to having a predefined structure, so it would be a great add-on for someone to finish implementing this algorithm.

A modding interface would also be a nice addition to the game for players to make their maps/themes and their parts for the community to use, as it is proven that this makes the game more interesting as some players have better creativity than maybe developers or have more time to work on things the devs can forget about thus giving them less work. To be able to build a community like this, it would be recommended to also add the game to the steam marketplace (See Figure 32) as there is already a workshop feature where users can share their maps and mods and include them in your game, then making building a platform for the mods unnecessary. A great example of this is Counter-Strike, an online multiplayer FPS, arguably the most popular videogame in history, which its maps and weapon skins are all made by the community.



Figure 32: Steam workshop

9 References

- [1] *Aiken format*. URL: https://docs.moodle.org/39/en/Aiken_format.
- [2] *Bag*. URL: https://poly.google.com/view/ems9KHrB_4x.
- [3] *Baseball cap*. URL: <https://poly.google.com/view/aaC5GgcWEhM>.
- [4] *Book*. URL: <https://poly.google.com/view/4S1nr7WmUxm>.
- [5] *Door key*. URL: <https://poly.google.com/view/dR-kItm5ihP>.
- [6] *Fake id*. URL: https://poly.google.com/view/OZXI8WCHi9_.
- [7] *Game github*. URL: <https://github.com/DicksonUB/juegoseriovrtfg>.
- [8] *Gas can*. URL: <https://poly.google.com/view/fmGMxckMykj>.
- [9] *Glasses*. URL: <https://poly.google.com/view/0Wsi-ygmiIX>.
- [10] *Google Poly site*. URL: <https://poly.google.com/>.
- [11] *Headphones*. URL: <https://poly.google.com/view/frvTEfwm9Yg>.
- [12] *Keyboard asset*. URL: <https://assetstore.unity.com/packages/tools/input-management/vr-keyboard-95936>.
- [13] *Locker*. URL: <https://assetstore.unity.com/packages/3d/environments/urban/low-poly-storage-pack-101732>.
- [14] *Luggage*. URL: <https://poly.google.com/view/023W-XcCmir>.
- [15] *Preguntados tutorial*. URL: <https://apperlas.com/comodines-de-preguntados-para-que-sirven/>.
- [16] *Restclient asset*. URL: <https://assetstore.unity.com/packages/tools/network/rest-client-for-unity-102501>.
- [17] *Synty studios*. URL: (<https://www.syntystudios.com/>).
- [18] *Unity Asset Store*. URL: <https://assetstore.unity.com/>.
- [19] *Video demo of game*. URL: https://www.youtube.com/watch?v=1gDgYomyV70&t=128s&ab_channel=AlexDickson.
- [20] *Virtualizer elite image*. URL: <https://www.cyberith.com/virtualizer-elite-setup3/>.
- [21] *VR with Andrew tutorial*. URL: https://www.youtube.com/watch?v=6N_0jeg6k0&t=545s&ab_channel=VRwithAndrew.

[22] Website *github*. URL: <https://github.com/DicksonUB/fileuploadserververtfg>.

10 Appendix I - Technical and user guide

10.1 Teacher

1. Upload questions on the website for the database or find an id for a question set appropriate for your class
2. Inform students of how the game works and give the id for the appropriate questions.
3. Give them the link to the website to download the game to their PC.
4. Inform them of the player steps in this technical guide

10.2 Player

1. Download the game from <https://vrtrivia-619c6.web.app/>
2. Install the downloaded apk file using the sideloading app for the Oculus Quest. You can follow the tutorial on: <https://uploadvr.com/sideload-quest-how-to/>
3. Go into applications, in Unknown Sources, and start the app named "Bank Run" (This name is like this because of the Unity project name)
4. Make sure you have a working internet connection on your headset
5. Set Oculus Quest in stationary mode when configuring the guardian. It is also recommended to be sitting down when playing the game, if you are sensitive to motion sickness. If you prefer to be standing up, remember to not rotate on top of yourself when moving at the same time with the joystick.

10.3 Developers

10.3.1 VR

The basic requirement is using Unity on a version higher than 2019.3 to be able to use the XR Interaction Toolkit. To update or add a new version to Unity, go to the Unity Hub Application and in Installs add a version equal to 2019.3 or higher.

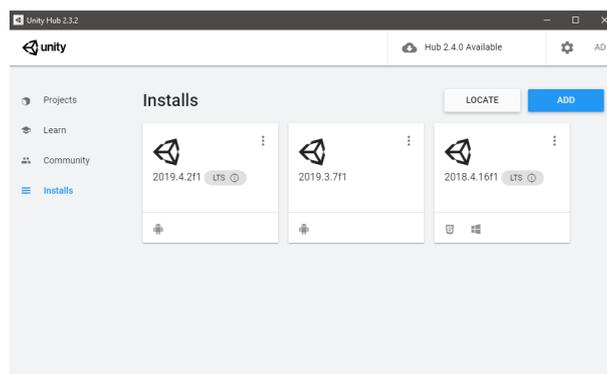


Figure 33: Changing Unity version

To set up the VR headset with Unity, you can follow the same tutorial I did: <https://learn.unity.com/tutorial/get-started-with-vr-beginner-the-escape-room?uv=2019.3&projectId=5e4abf44edbc2a09bf60dceb#5e4c0b4eedbc2a002125ac86> . This tutorial is also good if you are just starting developing for VR. Clone project from github link in Related Technologies section and develop from there.

It is highly recommended to invest in an Oculus Quest Link cable to be able to debug the application. If you set up a link connection, you can run the game in a matter of seconds (compared to more than 3 minutes without the cable) and see what the headsets see in real time on the PC. This cable makes the headset act like an Oculus Rift, which is not standalone.

10.3.2 Firebase

Remember to change the html values with the ones of your firebase account (See Figure 34). You can get them on the Firebase website when setting up the firebase realtime database.

```

<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.17.2/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/7.17.2/firebase-database.js"></script>
<!-- TODO: Add SDKs for Firebase products that you want to use
| https://firebase.google.com/docs/web/setup#available-libraries -->
<script src="https://www.gstatic.com/firebasejs/7.17.2/firebase-analytics.js"></script>

<script>
  // Your web app's Firebase configuration
  var firebaseConfig = {
    apiKey: "AIzaSyACyM4RY2L4r0rK1rZQbU1qDBIfpvMas3Q",
    authDomain: "vrtrivia-619c6.firebaseio.com",
    databaseURL: "https://vrtrivia-619c6.firebaseio.com",
    projectId: "vrtrivia-619c6",
    storageBucket: "vrtrivia-619c6.appspot.com",
    messagingSenderId: "712725552428",
    appId: "1:712725552428:web:f2cb33fad1a67b99926089",
    measurementId: "G-V16CV708RL"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
  firebase.analytics();

```

Figure 34: Firebase values

The hosting of the webpage is easier to do using Firebase as an option. It takes only 3-5 very detailed steps and you have the database in the same place. In case you want to make any changes to your already hosted website, go to the directory you initialized the firebase hosting on your PC and just enter the command “firebase deploy” again, after making the changes on the files you want.