

# Abstract

**Your Abstract**

# Acknowledgements

**Just want to give credit to friends, family and my supervisor who helped me develop ideas and the people who tested the game and provided feedback after development had concluded.**

# Table of Contents

[Abstract 0](#_Toc98610755)

[Acknowledgements 1](#_Toc98610756)

[Table of Contents 2](#_Toc98610757)

[Table of Figures 5](#_Toc98610758)

[Introduction 7](#_Toc98610759)

[Literature Review 8](#_Toc98610760)

[Chapter introduction 8](#_Toc98610761)

[Introduction to VR 8](#_Toc98610762)

[VR in past and present media 8](#_Toc98610763)

[The metaverse 8](#_Toc98610764)

[Early Virtual reality headsets and head mounted displays 8](#_Toc98610765)

[Sensorama 8](#_Toc98610766)

[Myron Krueger’s Videoplace 9](#_Toc98610767)

[Key elements of VR in 2003 9](#_Toc98610768)

[VR Possibilities outside of gaming 10](#_Toc98610769)

[VR in Education 10](#_Toc98610770)

[VR in medicine 10](#_Toc98610771)

[Game engines 10](#_Toc98610772)

[Tracking technology 12](#_Toc98610773)

[Xbox Kinect 12](#_Toc98610774)

[SLAM Overview 13](#_Toc98610775)

[Offline-SLAM for Map Acquisition a simple example of SLAM 13](#_Toc98610776)

[Oculus and SLAM 14](#_Toc98610777)

[APIS 14](#_Toc98610778)

[Rest API 14](#_Toc98610779)

[Web frameworks in python 14](#_Toc98610780)

[Django 14](#_Toc98610781)

[Flask 15](#_Toc98610782)

[WebSocket scripting 15](#_Toc98610783)

[Examples of VR puzzle games 15](#_Toc98610784)

[Tetris® Effect: Connected 15](#_Toc98610785)

[I Expect You To Die 2 16](#_Toc98610786)

[Methodology 18](#_Toc98610787)

[Requirements 19](#_Toc98610788)

[Functional Requirements: 19](#_Toc98610789)

[Non-functional Requirements: 23](#_Toc98610790)

[Design 25](#_Toc98610791)

[Introduction to chapter 25](#_Toc98610792)

[High Level Design 25](#_Toc98610793)

[Architecture diagram 25](#_Toc98610794)

[Use-Case diagram Website (Django) 26](#_Toc98610795)

[Use-Case diagram puzzle game VR (Unity) 27](#_Toc98610796)

[Database design 27](#_Toc98610797)

[Website Design 28](#_Toc98610798)

[Low Level Design 30](#_Toc98610799)

[Class Diagram puzzle vr game (Unity) Part 1 31](#_Toc98610800)

[Class Diagram puzzle vr game (Unity) Part 2 32](#_Toc98610801)

[8 Queens flowcharts: 33](#_Toc98610802)

[2048 game 37](#_Toc98610803)

[Reuse of existing code 38](#_Toc98610804)

[Implementation 39](#_Toc98610805)

[Sprint 0: 39](#_Toc98610806)

[Focus: 39](#_Toc98610807)

[Burn-down chart: 39](#_Toc98610808)

[Backlog: 39](#_Toc98610809)

[Sprint 1: 39](#_Toc98610810)

[Burn Down chart: 40](#_Toc98610811)

[Backlog: 40](#_Toc98610812)

[Explanation important code completed in sprint: 40](#_Toc98610813)

[Functional Requirements Testing 44](#_Toc98610814)

[Sprint 2: 45](#_Toc98610815)

[Focus: 45](#_Toc98610816)

[Burn Down chart: 45](#_Toc98610817)

[Back-log: 45](#_Toc98610818)

[Explanation important code completed in sprint: 46](#_Toc98610819)

[Functional Requirements Testing 48](#_Toc98610820)

[Sprint 3: 49](#_Toc98610821)

[Focus: 49](#_Toc98610822)

[Burndown chart: 50](#_Toc98610823)

[Explanation important code completed in sprint: 50](#_Toc98610824)

[Functional Requirements Testing 51](#_Toc98610825)

[Sprint 4: 53](#_Toc98610826)

[Focus: 53](#_Toc98610827)

[Burn-Down chart: 53](#_Toc98610828)

[Backlog: 53](#_Toc98610829)

[Explanation important code completed in sprint: 54](#_Toc98610830)

[Functional Requirements Testing 54](#_Toc98610831)

[Non-functional requirements testing: 55](#_Toc98610832)

[Project Evaluation 56](#_Toc98610833)

[Further Work and Conclusions 57](#_Toc98610834)

[Glossary 58](#_Toc98610835)

[Table of Abbreviations 59](#_Toc98610836)

[References / Bibliography 60](#_Toc98610837)

[Appendix A: First Appendix 63](#_Toc98610838)

# Table of Figures

[Figure 1 (Mishra, Shrawankar, 2016) ASSESSMENT OF ENGINES ON PERFORMANCE ASPECTS. 15](#_Toc98610839)

[Figure 2 (Zhang, 2012) Depth Map 16](#_Toc98610840)

[Figure 3 Body tracking (Zhang, 2012). 17](#_Toc98610841)

[Figure 4 The image provided is from the game Tetris® Effect: Connected (Monstars Inc. Resonair and Stage Games 2020). 20](#_Toc98610842)

[Figure 5 The image provided is from the game Tetris® Effect: Connected (Monstars Inc. Resonair and Stage Games 2020). 20](#_Toc98610843)

[Figure 6 The image provided is from the game I Expect You To Die 2 (Schell Games 2021). 21](#_Toc98610844)

[Figure 7 The image provided is from the game I Expect You To Die 2 (Schell Games 2021). 21](#_Toc98610845)

[Figure 8 Functional requirements. 27](#_Toc98610846)

[Figure 9 Non-functional requirements. 28](#_Toc98610847)

[Figure 10 Architecture Diagram. 29](#_Toc98610848)

[Figure 11 Website Use Case Diagram 30](#_Toc98610849)

[Figure 12 VR Game Use Case Diagram 31](#_Toc98610850)

[Figure 13 Database Design 31](#_Toc98610851)

[Figure 14 Home Page UI Design 32](#_Toc98610852)

[Figure 15 8 Queens Leader Board Page Design 32](#_Toc98610853)

[Figure 16 2048 Leader Board Page Design 33](#_Toc98610854)

[Figure 17 About Page Design 34](#_Toc98610855)

[Figure 18 VR Game Class Diagram Part 1 35](#_Toc98610856)

[Figure 19 VR Game Class Diagram Part 2 36](#_Toc98610857)

[Figure 20 PossibleMovesXAxis function flowchart 37](#_Toc98610858)

[Figure 21 Part of PossibleMovesXAxis function flowchart 37](#_Toc98610859)

[Figure 22 CalcDiagonalTopRight Function Flowchart 38](#_Toc98610860)

[Figure 23 CalcDiagonalTopRight Function Flowchart Loops 38](#_Toc98610861)

[Figure 24 CalcDiagonalTopRight Function Flowchart Loops, Add Chess coordinates. 39](#_Toc98610862)

[Figure 25 CalcDiagonalTopLeft Function Example 39](#_Toc98610863)

[Figure 26 Basic Game Logic For 8 Queens 40](#_Toc98610864)

[Figure 27 Cube merge and move algorithm 2048 Flowchart 41](#_Toc98610865)

[Figure 28 Burndown Chart Sprint 0. 43](#_Toc98610866)

[Figure 29 Burn Down Chart Sprint 1. 44](#_Toc98610867)

[Figure 30 Chess Coordinate List Part A. 45](#_Toc98610868)

[Figure 31 Chess Coordinate List Part B. 45](#_Toc98610869)

[Figure 32 PossibleMovesXAxis Code Snippet 45](#_Toc98610870)

[Figure 33 CalcDiagonalTopLeft Code Snippet 46](#_Toc98610871)

[Figure 34 Chess Board Example 46](#_Toc98610872)

[Figure 35 All Possible Moves Output Example. 47](#_Toc98610873)

[Figure 36 InLineOfSightQueen Code Snippet 47](#_Toc98610874)

[Figure 37 CalculateScore Code Snippet 48](#_Toc98610875)

[Figure 38 Functional requirements testing sprint 1. 49](#_Toc98610876)

[Figure 39 Burndown chart sprint 2. 49](#_Toc98610877)

[Figure 40 cubeMoveMerge Code Snippet. 50](#_Toc98610878)

[Figure 41 Figure 41 Figure 41 MoveMerge Code Snippet Part 2 51](#_Toc98610879)

[Figure 42 LeftButton Code Snippet. 51](#_Toc98610880)

[Figure 43 Functional Requirement Tests Sprint 2. 53](#_Toc98610881)

[Figure 44 Burn Down Chart Sprint 3. 54](#_Toc98610882)

[Figure 45 REST API Code Snippet 54](#_Toc98610883)

[Figure 46 HTTPRequest Code Snippet 55](#_Toc98610884)

[Figure 47 Functional Requirements Tests Sprint 3. 56](#_Toc98610885)

[Figure 48 Burn Down Chart Sprint 4. 57](#_Toc98610886)

[Figure 49 Keyboard Code Snippet. 58](#_Toc98610887)

[Figure 50 Image Of In Game Keyboard 58](#_Toc98610888)

[Figure 51 Functional Requirements Test Sprint 4. 59](#_Toc98610889)

[Figure 52 Non-Functional Requirement Tests 59](#_Toc98610890)

# Introduction

In this paper I will design and build a VR game that involves puzzles. More specifically the 8 queens puzzle, and other such chess related puzzles. The game will be created using a game engine, in this case unity, the language of choice is c#. The chosen platform is the quest 2 which uses a version of the android operating system. Both the headset and its operating system were developed by Meta formally known as Facebook. To add further complexity, I will develop a website a long side the game to display scores submitted by players. This paper will contain the research for said project, design information, testing and research.

# Literature Review

## Chapter introduction

This chapter of the report contains my research on VR technologies, web technologies and other such things that I will require to design and build my VR game

## Introduction to VR

There have been many representations of VR in the media, in the form of books and films.

### VR in past and present media

The first book, William Gibson’s Virtual light, in summary, is about a pair of glasses which allow a person to view information by feeding it into that person’s optic nerve (Gibson et al., 1993). Ready player one is a dystopian future in which people use virtual reality to escape the tough reality of life. The VR world they go to is called the Oasis (Cline et al., 2011).

### The metaverse

The metaverse can be described as a 3-dimensional virtual universe (Dionisio, III, Gilbert, 2013). This however differs from the concept of cyberspace that represents all of a shared online and virtual space across all dimensions (Dionisio, III, Gilbert, 2013). There are multiple different versions of the metaverse, the first example is an enlarged virtual world, and another is a large network of virtual world (Dionisio, III, Gilbert, 2013). The second example is wanted can be seen in ready player one. The virtual universe is split into multiple virtual worlds (Cline et al., 2011) that players can explore interact with, they are able to travel to other worlds inside the universe.

All of these examples show how virtual reality has been represented over the years, however in reality, today the technology is much more primitive compared the previous given fictional examples.

To illustrate how VR works in reality, one could look to early examples of VR headsets and VR like devices that have previously been developed in the past.

## Early Virtual reality headsets and head mounted displays

### Sensorama

An early example of a VR headset was the Sensorama( [Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), [R. Sherman](https://learning.oreilly.com/search/?query=author%3A%22William%20R.%20Sherman%22&sort=relevance&highlight=true),  [D. Will](https://learning.oreilly.com/search/?query=author%3A%22Jeffrey%20D.%20Will%22&sort=relevance&highlight=true), 2009). This piece of literature describes early VR technology. “The Sensorama was the brainchild of cinematographer and inventor Morton Heilig” ”( [Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), [R. Sherman](https://learning.oreilly.com/search/?query=author%3A%22William%20R.%20Sherman%22&sort=relevance&highlight=true),  [D. Will](https://learning.oreilly.com/search/?query=author%3A%22Jeffrey%20D.%20Will%22&sort=relevance&highlight=true), 2009). As described by the writer of this literature the system “was lacking a major component of the modern virtual reality system: response based on user’s actions” ”( [Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), [R. Sherman](https://learning.oreilly.com/search/?query=author%3A%22William%20R.%20Sherman%22&sort=relevance&highlight=true),  [D. Will](https://learning.oreilly.com/search/?query=author%3A%22Jeffrey%20D.%20Will%22&sort=relevance&highlight=true), 2009). Early headsets were very basic, this simply means that the system did not allow a user’s actions to have any impact on the virtual world unlike modern day headsets that allow a user to interact with the virtual world using controllers or in some cases the users’ hands.

### Myron Krueger’s Videoplace

“Krueger’s artificial reality provided a second-person view of a virtual world in which participants could watch themselves within the world” ”( [Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), [R. Sherman](https://learning.oreilly.com/search/?query=author%3A%22William%20R.%20Sherman%22&sort=relevance&highlight=true),  [D. Will](https://learning.oreilly.com/search/?query=author%3A%22Jeffrey%20D.%20Will%22&sort=relevance&highlight=true), 2009). [Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), [R. Sherman](https://learning.oreilly.com/search/?query=author%3A%22William%20R.%20Sherman%22&sort=relevance&highlight=true) literature also talks about another early VR headset such as Myron Krueger’s Video place. This headset gave the user a “second-person point of view” of themselves. No games could be played on this headset and the user did not have any devices or other sensors attached to their body( [Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), [R. Sherman](https://learning.oreilly.com/search/?query=author%3A%22William%20R.%20Sherman%22&sort=relevance&highlight=true),  [D. Will](https://learning.oreilly.com/search/?query=author%3A%22Jeffrey%20D.%20Will%22&sort=relevance&highlight=true), 2009) unlike some modern-day VR headsets that use full body tracking that require sensors to be placed all over the user’s body including head, legs and arms depending on what degree of tracking you want.

## Key elements of VR in 2003

There are a few key elements of VR that have been identified in the past, this information will help with the creation of the puzzle game and help to mitigate any mistakes that could be made during design.

The paper written by Sherman details four key elements of virtual reality.

A virtual world is the context of a medium(Sherman, [B.Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), 2003). In this case the case of this project this would refer to the game world. This will also tie into the writers next point regarding immersion. This is because the virtual world, if created correctly helps give a sense of immersion and improves the experience of the player.

It must have immersion, Immersion into a different reality or another point of view(Sherman, [B.Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), 2003). It must have both Physical and mental immersion, this can be split into two, mental immersion can often be referred to as having a mental presence within the world (Sherman, [B.Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), 2003) and physical immersion, a body entering into a medium (Sherman, [B.Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), 2003). Both points apply today, If the VR experience will be greatly affected if immersion is broken, so during development I must pay extra attention to this as one of the major causes of braking immersion is frame drops whilst playing a VR game, therefore it is imperative that my code is optimised.

It must have Sensory feedback, “VR allows participants to select their vantage point by positioning their body and to affect events in the virtual world” (Sherman, [B.Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), 2003). This refers to moving your arm for example in the VR worlds and have it match up with the real world. Technologies to make this more accurate already exist, with the rise of full body tracking for certain headsets. Sensors are placed on the body and tracked by the headset allowing for more accurate tracking of limbs creating the illusion of the player actually being in the virtual world.

The final key element according to the writer is, Interactivity. “For virtual reality to seem authentic, it should respond to user actions, namely, be interactive” (Sherman, [B.Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), 2003). In the case of current day VR this could be seen to be tied to the act of a user performing an action and getting a reaction. For example, a player pushes a box with there in game hand and the box moves back in response.

## VR Possibilities outside of gaming

### VR in Education

There are many applications for VR outside of gaming for entertainment, there also are applications for VR in the educational sector, for example in the research paper by (Holly, Pirker, Resch, Brettschuh, Gutl, 2021), it goes over how learning environments like Marroon (GameLabGraz), are environments designed to provide an interactive learning environment for students and allows student to perform experiments in a VR environment. The experiment that the researchers performed consisted of testing to test several factors for the game, Immersion, and engagement, learning value from both the teachers and learners’ respective perspectives (Holly, Pirker, Resch, Brettschuh, Gutl, 2021). In summary the response was positive in all areas of study, most students and teachers were happy with the level of immersion and engagement, and on average most students and teachers concluded that is had a high learning value (Holly, Pirker, Resch, Brettschuh, Gutl, 2021).

### VR in medicine

VR also has applications in medicine as out lined in this study by (Pourmand, Davis, Marchak, Whiteside, Sikka, 2018). The study delves into how VR can be used to distract patents from many types of pain, such as acute pain (Pourmand, Davis, Marchak, Whiteside, Sikka, 2018). It concluded that VR have become one important option for helping with pain management and how the evidence in said study shows how VR can be effective at distracting patients from many types of pain including actue and chronic short-term (Pourmand, Davis, Marchak, Whiteside, Sikka, 2018). The study references a paper that goes on to run some experiments on hospitalized patients, it was noted in the study that patents, had a significant drop in pain by around 24% in the VR group of patents compared to the 13% drop in pain in the control group (Tashjian, 2017).

## Game engines

The Wikipedia definition of a game engine is as follows, "A game engine is a software framework primarily designed for the development of video games, and generally includes relevant libraries and support programs. The "engine" terminology is like the term "software engine" used in the software industry."[Game engine 2021].

For further clarification this is what is defined as a game engine in Comparison between Famous Game Engines and Eminent Games journal article. Game engines are sets of tools that allow a programmer to, perform game related tasks like interpretation and physics related tasks and for focusing on the niceties that make the game great([Mishra](https://www.ijimai.org/journal/bibcite/contributor/5998), [Shrawankar](https://www.ijimai.org/journal/bibcite/contributor/5999), 2016).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ASSESSMENT OF ENGINES ON PERFORMANCE ASPECTS** | | | | | | |
| Game Engine | Platforms | Language Support | AI Engine | Physics Engine | Forward Compatibility | Backward Compatibility |
| CryEngine 3 | Win, X360, PS3, Wii U | C++, Visual Script, Lua | Lua-driven AI | Soft body | No | Yes |
| Hero Engine | Win | Hero Script | AIseek | PhysX | Partial | Yes |
| Source 2 Engine | Win, Mac, Xbox 360, Wii, Linux, Android | C++ | AI Director | Ipion | No | Partial |
| Unity 4 | BlackBerry, Win Phone, Win, OS X, Android, iOS, Apple TV, PS3/4, PS Vita, Xbox 360, Xbox One, Wii U, Wii | C#, JavaScript, Boo | RAIN | PhysX | Partial | yes |
| Unreal 4 Engine | Windows, OS X Linux, Xbox 360/ One, PS3/4, Wii U, Android, iOS, WinRT, PS Vita | C++, C#, GLSL, CG, HLSL | Kynapse | PhysX | Partial | Yes |
| Vision Engine 8 | Windows, Xbox 360, PS3, Wii, Wii U, iOS, Android, Win Phone, PS Vita | C++ | Kynapse | Bullet, ODE, PhysX | No | Partial |

Figure 1 ([Mishra](https://www.ijimai.org/journal/bibcite/contributor/5998), [Shrawankar](https://www.ijimai.org/journal/bibcite/contributor/5999), 2016) ASSESSMENT OF ENGINES ON PERFORMANCE ASPECTS.

As can be seen from figure one CryEngine, Hero Engine don’t support nearly as many platforms as the other four. Furthermore, Hero engine does not support any of the more popular programming languages like C++ or C#. In addition to this Hero Engine, Unity and unreal only have partial forward compatibility.

Unreal Engine has Blueprint Visual Scripting” (Chu, Zaman, 2021). It was created to support the workflow of designers and artist by giving them access to tools normally only accessible to programmers (Chu, Zaman, 2021). Blueprints is object oriented visual programming system which is used to create gameplay elements (Chu, Zaman, 2021). Other engines also have these capabilities, such as unity. A recent example of plugins that allow unity to obtain this functionality include, FlowCanvas, playmaker, Bolt and Amplify shader editor (Chu, Zaman, 2021).

## Tracking technology

### Xbox Kinect

Oculus uses state of the art tracking technology that developed from a massively successful technology known as the connect Kinect was an early example of SLAM being used, originally developed at Microsoft research Cambridge and then productized within the Xbox team at Microsoft. The Kinect is made up of 3 devices, the first being Depth sensor, the second being an RGB camera and lastly a 4-microphone array, this all gives 3D full body tracking, facial recognition and voice recognition (Zhang, 2012). The Kinect senses depth using its IR camara and IR projector which make up the depth sensor (Zhang, 2012). This works by the IR project projecting dots in an area, then if the dot in said image matches dots in the known dot pattern produced by the projector it is possible to rebuild it using 3D triangulation (Zhang, 2012).

A picture containing text

Description automatically generated

Figure 2 (Zhang, 2012) Depth Map

[Here](https://gyazo.com/9f3e31cf8efe38baab1eb00a047b4845) is a depth map produced by the Kinect, the darker the Gray the closer the object (Zhang, 2012).

This can then be used in conjunction with other techniques to track a person's body, figure 3 gives the general outline of this.

A picture containing diagram

Description automatically generated  
Figure 3 Body tracking (Zhang, 2012).

From the depth image a body part distribution is inferred, this is done per pixel as can be seen from the second part of the image above. ([Shotton](https://ieeexplore.ieee.org/author/37550019800), [Fitzgibbon](https://ieeexplore.ieee.org/author/37282639100), [Cook](https://ieeexplore.ieee.org/author/37951907600), [Sharp](https://ieeexplore.ieee.org/author/37408379000), [Finocchio](https://ieeexplore.ieee.org/author/37944916600), [Moore](https://ieeexplore.ieee.org/author/37950527600), [Kipman](https://ieeexplore.ieee.org/author/37944916100), [Blake](https://ieeexplore.ieee.org/author/37282602200), 2011). Local modes of this are then hypothesized to provide a good quality proposal for the joints in 3 dimensions, this all can also be done for multiple users([Shotton](https://ieeexplore.ieee.org/author/37550019800), [Fitzgibbon](https://ieeexplore.ieee.org/author/37282639100), [Cook](https://ieeexplore.ieee.org/author/37951907600), [Sharp](https://ieeexplore.ieee.org/author/37408379000), [Finocchio](https://ieeexplore.ieee.org/author/37944916600), [Moore](https://ieeexplore.ieee.org/author/37950527600), [Kipman](https://ieeexplore.ieee.org/author/37944916100), [Blake](https://ieeexplore.ieee.org/author/37282602200), 2011).

### SLAM Overview

SLAM stands for Simultaneous localization and mapping (MathWorks). There are many applications for SLAM including, using it with autonomous vehicles to produce maps of environments and pinpoint a vehicle in that map all at the same time (MathWorks). They can allow the vehicle to map environments and avoid obstacles and path find in that environment (MathWorks).

There are many methods of slam, the first is Visual slam, this type of slam utilises images obtained from cameras and other sensors and there are two main types of this version of slam, Spare methods that match features of images at then utilise an algorithm like ORB-SLAM and dense methods that use algorithms like LSD-SLAM (MathWorks).

The second method of SLAM is LiDAR SLAM this version of slam usually uses some form of distance sensor, for example a laser sensor (MathWorks).

Both types of SLAM have advantages and disadvantages, LiDAR is not affected by light level and its angular view is quite large. Unfortunately, however, on a large-scale LiDAR is not as viable in certain situations as it can get very expensive (Huang, Zhao, Liu, 2019). Visual SLAM, however, does not do so well in low light conditions and in low texture environments (Huang, Zhao, Liu, 2019). But visual SLAM can usually be implemented at a much lower price as RGB-D cameras, used to implement it can be obtained quite cheaply (Taketomi, Uchiyama, Ikeda, 2017).

### Offline-SLAM for Map Acquisition a simple example of SLAM

There are many ways to use SLAM, a simple example of SLAM involving robots is as follows; the first phase is map learning phase, as a robot moves through an environment the sensor data obtained, and labels are given such as place names. After this the SLAM algorithm creates a map from the data to robot collected, this data can then be used to plan paths and localization and so on (Frese, Wagner, Rofer, 2010).

### Oculus and SLAM

SLAM tracking technology is a tracking method utilized by the Oculus quest and rift s (Hesch, Kozminski, Linde, 2019). On both these headsets it uses three major types of sensors to work out the VR headset’s location, room aspects like where the floor is and track controllers with reference to a 3d map of the space around it that is generated continuously (Hesch, Kozminski, Linde, 2019).). The 3 sensors include, IMUs that track the headsets orientation and position, cameras to generate a 3d map of the room, and infrared emitters mitigating the controller drift cause by having a lot of the IMU sensors (Hesch, Kozminski, Linde, 2019).

## APIS

APIs are powerful tools that can be used in this project, to break down what an API is, we must first learn what it stands for, its stands for application programming interface([Uzayr](https://learning.oreilly.com/search/?query=author%3A%22Sufyan%20bin%20Uzayr%22&sort=relevance&highlight=true), 2016). Is a way in which you can hide the component specification from the actual implementation or the user of said components (Souza, Redmiles, 2009). One reason for doing this is it separates the modules in to public and private modules the public one being the API and the private one being the implementation of said API, there for any changes made to the private module does not impact the public module (Souza, Redmiles, 2009). Using APIs allows a company to give a developer functionality without said developer needing to know how it works (Souza, Redmiles, 2009).

## Rest API

To give context, using rest is one way we could implement a way to transmit data between a virtual reality head set and a webserver, in this case it would be the transmission of score data etc.

To summarize a rest API is a type of application programming interface that conforms to a set of constraints (Kotstein, Justus, 2021). Rest apis have two aspects, the first is that they are a type of architectural style that can dictate how distributed hypermedia systems are built and used like the world wide web (Kotstein, Justus, 2021). Furthermore, this journal article goes into further detail for example the second aspect it gives is that rest is like a guide that formalizes how web components like urls and http for example should be used when creating new applications that utilize them.

## Web frameworks in python

### Django

Django is a collection of python modules designed for building web applications (Yudin et al., 2020). The Django framework can map URLS to methods, it can also render HTML webpages and it can handle cookies, session and also web security (Yudin et al., 2020). Django can also support things like relational databases using object-relational mapping (Yudin et al., 2020). It has a build in database solution, known as SQLite (Yudin et al., 2020). It also contains tools that allow easy authentication and authorization (Yudin et al., 2020). It is also widely used, namely these websites use it, YouTube, Spotify and also many others (Yudin et al., 2020).

### Flask

Flask is a small framework and is small enough to be called a micro framework (Grinberg eta al., 2018). Flash was designed to be extendable, and you can pick and choose what you need to avoid bloat (Grinberg eta al., 2018). Flash has a few dependencies, including the routing, debugging and web server gateway (Grinberg eta al., 2018). However, it does not have support for accessing databases, validating web forms or authenticating users (Grinberg eta al., 2018). This is an issue for this project as we require a database to store score data from the game, also a way to authenticate users.

Comparing Django’s and Flasks features, Django is the most logical option as it allows the use of an SQLite database and has user authentication and authorization features which is lacking in the Flask framework.

## WebSocket scripting

A WebSocket is a transfer control protocol for web applications. It provides connection in both ways meaning data can be sent to and from a server and client. However instead of connecting to an internet host and a port, it facilitates connections using URLS instead. To establish a connection between a server and client the protocol is changed from HTTP to WebSocket protocol. Once a connection is established messages can be sent back and forth between the client and the server (Lubbers, Albers, Salim,2011). This is one way in which information could be sent between the vr headset and a webserver that stores and displays score data in a database. However, the connection only needs to be in one direction as, no messages are going to be sent to the client from the server.

## Examples of VR puzzle games

By looking at some high-quality games we can see what is expected from a high-quality VR game.

### Tetris® Effect: Connected

The first game we are going to look at is called Tetris® Effect: Connected. In summary, it is a game where the objective is to complete rows at the bottom of the screen, this is done by rotating different shapes to form these rows (Monstars Inc. Resonair and Stage Games 2020).

. A picture containing text, outdoor, light, night

Description automatically generated

Figure 4 The image provided is from the game Tetris® Effect: Connected (Monstars Inc. Resonair and Stage Games 2020).

As can be seen from the image above (figure 4) there are multiple game play elements, a scoring system, levels and elements in the scene, I.E the use of vibrant and interesting colours. Some of these elements would be useful to use in our VR puzzle game, this is because they create interesting and beautiful game play to immerse the user in the game.

A picture containing text, water, aquatic mammal, dolphin

Description automatically generated

Figure 5 The image provided is from the game Tetris® Effect: Connected (Monstars Inc. Resonair and Stage Games 2020).

As can be seen from this next image of gameplay (figure 5), the vibrant colours and simplicity of the UI is a consistent theme. Not overcrowding a user’s view and keeping things simple is another thing we must be careful of when developing a game.

### I Expect You To Die 2

The second game to analyse is called I Expect You To Die 2. In summary this game involves solving a selection of puzzles, the puzzles are split into missions for the player to complete where the player plays as an agent trying to defeat an enemy called Zoraxis (Schell Games 2021).

A picture containing graphical user interface

Description automatically generated

Figure 6 The image provided is from the game I Expect You To Die 2 (Schell Games 2021).

Graphical user interface, website

Description automatically generated

Figure 7 The image provided is from the game I Expect You To Die 2 (Schell Games 2021).

The gameplay above shows an intricate and immersive world with high levels of detail, this level of detail, is likely important for immersion. As can be seen from both images (figure 6 and 7) the user is able to interact will objects using there in game hands, picking objects up, looking at them closely to solve the puzzle, read instructions and other such actions. All these small features create a sense of immersion in the world the author has created, as is the nature with VR, if the game is implemented correctly, it can make a player feel intertwined with the story and provide a real sense of emergency to solve problems that occur, like with the image above, the user is required to disarm a bomb.

If all these elements are observed, it is possible to create an immersive and detailed world for a user to enjoy. These key elements will likely be very useful to incorporate into our game. The key elements to take away from these games, is allowing a user to interact with the world as much as possible.

# Methodology

To produce the VR game, we decided to use an agile development process. Agile methods of software developments use an iterative and incremental approach to software development (Kumar, Bhatia, 2012). The type of Agile method of development we have chosen is called, the scrum method. A scrum consists of sprints which are individual iterations that can last anywhere from 2 to 4 weeks each (Kumar, Bhatia, 2012).

Planning for a sprint starts at the beginning of a sprint and that time is also used to determine what can be delivered in the next one (Hron, Obwegeser, 2021). A review of the sprint is then conducted at the end of the sprint to show the outcome of said sprint to the customer and to obtain feedback for the next sprint, feedback will then be used in the next sprint (Hron, Obwegeser, 2021).

Now knowing the information presented above, we will be conduction 2 weeklong sprints for each stage of development.

Project setup Sprint 0:

This sprint will be 2 days long. In this sprint, basic project setup will be completed. Such as importing all libraries and packages that will be needed include Oculus integrations, creating a very basic prototype application to test all packages are working correctly and everything was setup and imported correctly.

Sprint 1:

Basic game development of the puzzles will be done in this sprint, the two puzzles giving a week to produce each puzzle during the sprint I.E 8 queens puzzle and so forth. This will then be tested according to the requirements of the system.

Sprint 2:

Further game development will be done here, including the creation of the puzzle 2048

Sprint 3:

Database setup will be performed in this sprint, the website will also be further developed. Testing to make sure data is stored correctly and is sent by the headset will also be done here.

Sprint 4:

User interface will be developed here according to the requirements specification. This will be tested along with the rest of the project to make sure everything is working correctly.

Sprint 5:

This sprint will consist of making sure everything is working correctly and working out any bugs that may have occurred and gone unnoticed during the systems development. Final testing of the system will also be done here, making sure the system properly satisfies all requirements.

# Requirements

This section of the report will contain 37 requirements in total. 27 Functional and 10 non-functional. We will be using this to illustrate the Priority of each of the requirements.

The Requirements are Identified with an ID and the type of requirement in said ID. For example:

FR:

-VR means a requirement related to the VR game itself.

* -VR-8Q is a VR requirement relating to 8 Queens
* -VR-2048 is a VR requirement relating to 2048

-WS means a requirement relating to the website.

* -WS-8Q is a website requirement relating to 8 Queens
* -WS-2048 is a website requirement relating to 2048

-VR/WS means it is a requirement that relates to both.

* -VR/WS-8Q is a requirement relating to both VR, the website and 8 Queens
* -VR/WS-2048 is a requirement relating to both VR, the website and 2048

Moscow stands for must have, should have, could have, won’t (Asghar, Bhatti, Tabassum, Shah, 2017, Page 303-313).

Must – The system must have this, and this is the highest priority.

Should – The system should have this, this is mid priority.

Could – This system could have this, however it is not necessary.

Won’t – This could be added in the future, but it won’t have this at this given time.

The requirement specification column gives more detail into the requirement and why it is necessary.

## Functional Requirements:

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement No. | Requirement | Requirements Analysis | Priority |
| FR1-VR-8Q | User must be able to pick up the chess piece using the controllers | Necessary for solving the puzzle as it allows the user to place the chess pieces on the chess board. | Must |
| FR2-VR-8Q | Chess board must be able to detect the location of queens | Required, as need to know where the chess pieces are, as their positions will be used to determine if the solution given by the user is valid. | Must |
| FR3-VR-8Q | Algorithm must be able to determine if an 8 queens’ solution is valid. | Necessary for a user to see if they solved the puzzle or not. | Must |
| FR4-VR-8Q | Timer that counts and starts when the user enters the puzzle level. | Necessary to determine a user’s score. | Must |
| FR5-VR-8Q | In the 8 Queens algorithm the score must be calculated using the time the user takes to solve the puzzle. | Necessary to allow the user to see how well they did compared to others on the leader board. | Must |
| FR6-VR-8Q | After 8 queens puzzle solved, the user is notified and taken back to menu in 5 seconds. | Necessary to allow the user to be removed from the level after the puzzle is complete. | Must |
| FR7-VR-2048 | 2048 Algorithm must move cubes in each direction when the corresponding button is clicked | When the user clicks a directional button with the controllers, the cubes will move in the direction indicated by said button. | Must |
| FR8-VR-2048 | 2048 Algorithm must merge cubes when there are of the same value | Necessary for the game to work. When two cubes are the same, i.e., 2 and 2, they are added together to produce a new cube, in this case, 4. | Must |
| FR9- VR-2048 | Must check to see if the user has lost the game before every move the user makes. | Ensures the moment the player loses they are notified. | Must |
| FR10-VR-2048 | In the 2048 algorithm when two cubes are merged and added together to make 2048 notify the user they have won. | Necessary to allow the user to win the game. When two 1024 cubes are merged, the player wins. | Must |
| FR11-VR-2048 | In 2048 the algorithm, the user must gain 10 points with every cube merge they make. | This is a method of keeping score, the more merges they make, the higher their final score will be. | Must |
| FR12- VR-2048 | After the user has either lost or won, a button must appear to allow them to travel back to the main menu | Necessary to allow a user to travel back to the main menu after they have completed the puzzle. | Must |
| FR13-VR-Menu | Button on the menu to take the user to the 8 queens puzzle | Necessary to travel to the 8 queens puzzle | Must |
| FR14-VR-Menu | Button on the menu to take the user to the 2048 puzzle | Necessary to travel to the 2048 puzzle | Must |
| FR15-VR-Menu | In game keyboard to allow a user to enter their name | Necessary to allow the user to input their name, so it can be used to display with their score on the website. | Must |
| FR16-VR/WS-8Q | 8 Queens - Website must receive Score data and username after the puzzle is solved | This will use a REST API to transmit data between the headset and the webserver in a JSON format. Necessary to allow score data to be displayed on the 8 queens leader board. | Must |
| FR17-VR/WS-2048 | 2048 – Website must receive score data and username if the user loses. | This will use a REST API to transmit data between the headset and the webserver in a JSON format.  The user still has a score value even if they lose that is why this requirement is necessary  Necessary to allow score data to be displayed on the 2048 leader board. | Must |
| FR18- VR/WS-2048 | 2048 – Website must receive and username score data if the user Wins. | This will use a REST API to transmit data between the headset and the webserver in a JSON format.  After the player wins, there score data will need to be sent to the web server to be displayed in the 2048 leader board. | Must |
| FR19-WS | The website must have a working navigation bar to take a user to the 4 pages. | The Website will have a working navigation bar to take the user to the 4 pages, Home, 8 Queens leader board and about. | Must |
| FR20-WS | On the 8 Queens leader board page score and usernames should be pulled from the database and displayed on the page | On the 8 Queens leader board page score data and player usernames should be displayed in the leader board table, highest to lowest. | Must |
| FR21-WS | On the 2048 leader board page score and usernames should be pulled from the database and displayed on the page | On the 2048 leader board page score data and player usernames should be displayed in the leader board table, highest to lowest. | Must |
| FR22-VR | Help could be provided if the user is struggling to sole a specific problem | The game will provide a hint if a user does not make a move within a given time frame | Should |
| FR23-WS | Allow a user to click a link to download the game | When the user clicks a link, they are taken to a place where they can download the game to play. | Should |
| FR24-VR | Be able to move around the world using hand tracking | The user should be able to make a hand gesture and move around the world freely | Could |
| FR25-VR | Data on the headset should be backed up to a server at the push of a button | Store games save data on dedicated server and pull data when game is loaded | Could |
| FR26-WS | A login system that allows a user to enter the website | Allow users to login to the website | Could |
| FR27-WS | Users can edit and delete their high scores if necessary once they are logged in | Allow users to manage their leader board data, I.E delete, change their name etc | Won’t |
| FR28-VR/WS | The user should be able to view their high score data within the headset | The user can pull their high score data from the website and view it | Won’t |
| FR29-VR | Allow full body tracking, allowing the user to use all limbs when playing the game | Allow full body tracking for input | Won’t |
| FR30-VR | Multiplayer allowing multiple users to join a session and solve puzzles with each other | Allow multiplayer – Allow users to join other user to solve the puzzles. | Won’t |

Figure 8 Functional requirements.

## Non-functional Requirements:

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement No. | Description | Requirement Specification | Priority |
| N-FR1 | A User interface is necessary to allow users to select different puzzles | The game must have a way to select between different puzzles in the form of a UI | Must |
| N-FR2 | The game must be maintainable enough to allow for the addition of new puzzles | Allow for the addition of new puzzles, it must be flexible | Must |
| N-FR3 | The game must be able to run at a reasonable frame rate on the quest 2, 60fps minimum | Must run on the Oculus Quest 2, Necessary as this is the target platform | Must |
| N-FR4 | Good maintainability is necessary to allow updates to the system. | Must be simple to maintain both quest 2 app and website. Splitting programs into Classes and functions etc must be observed to prevent the code from being hard to maintain | Must |
| N-FR5 | The game should be able to check the validity of a solution without any long durations of lag. | Be able to check a puzzle solution within 1 second | Must |
| N-FR6 | Multiple users are going to need to access the website at one time, the website should be robust enough for this to be allowed | Multiple users must be able to use website at the same time | Must |
| N-FR7 | There should be minimal delay when sending a message from the headset to the webserver | Be able to send data to the website from the headset and update it within 1 second | Should |
| N-FR8 | The game should also be available for headsets that require to be connected to a desktop machine | Cross compatibility with other headsets, the Oculus rift S for example. | Could |
| N-FR9 | If a user of the website does not have the game, they have no reason to access the website and make changes to scores | Users who do not own the game should not be able to access the website | Could |
| N-FR10 | The website must not be down for long periods of time | The website would not be down for more than 5 minutes during a fault | Wont |

Figure 9 Non-functional requirements.

# Design

## Introduction to chapter

In this chapter we will go over the design of the system, this will include a selection of diagrams of the systems classes, use cases, architecture, database design and UI design.

## High Level Design

Should have an introduction

### Architecture diagram

Diagram

Description automatically generated

Figure 10 Architecture Diagram.

The above diagram(fig 10) shows the system architecture. It models how the Oculus quest 2 will send data to the server using a REST API and how that data will be viewed using the clients web browser.

### Use-Case diagram Website (Django)

Diagram

Description automatically generated

Figure 11 Website Use Case Diagram

This is a use-case diagram for the website (fig 11). This use case diagram describes a given users interactions with the website itself, how they view different webpages. Users in this case, includes people who play the game, and general users who do not necessarily own the game but wish to view leader boards for a given puzzle, and the about and home pages.

### Diagram Description automatically generatedUse-Case diagram puzzle game VR (Unity)

Figure 12 VR Game Use Case Diagram

This is a use case diagram for the VR game (fig 12). This use case diagram describes the player actor’s interaction with the VR game. The goal of this use case is to send score data to the website and return to the main menu, after the chosen puzzle have been completed. The data is sent to the webserver.

### Database design

Diagram

Description automatically generated

Figure 13 Database Design

The above diagram (fig 13) is the database design for the website. The database will have two tables, one containing score data for the 8 queens puzzle and the other for 2048. All records in both tables will contain a username and a score value.

### Website Design

#### Home Page



Figure 14 Home Page UI Design

Above UI mock-up shows the design of the home page.

#### Eight Queens Leader board Design

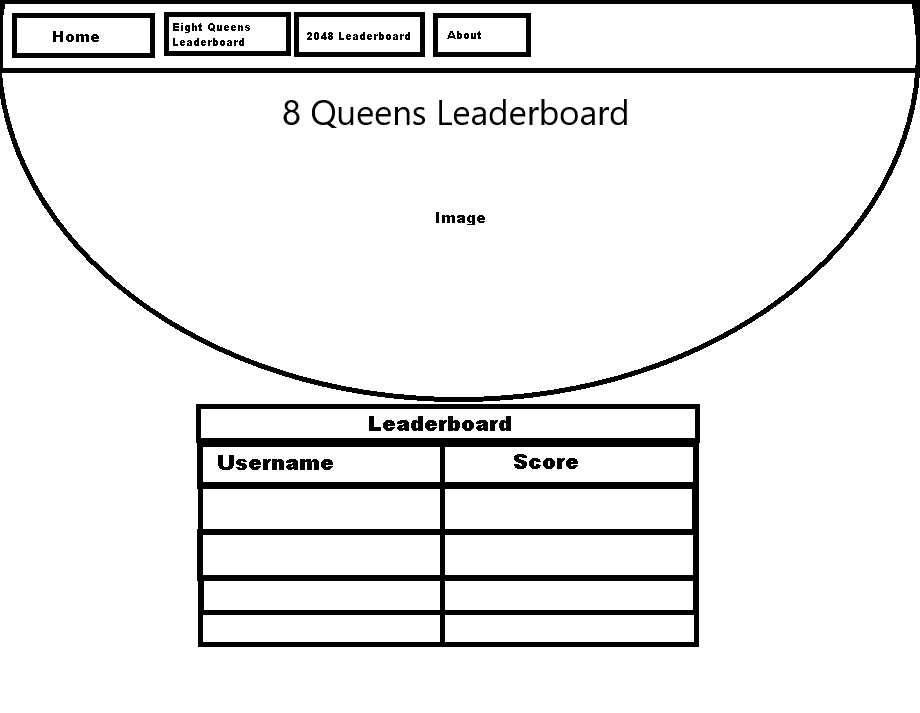


Figure 15 8 Queens Leader Board Page Design

Above UI mock-up shows the design of the 8 queens leader board page (fig 15). The page will contain a leader board, some text to describe the page and an image, the image will be a screen shot of the 8 queen’s puzzle.

#### 2048 Leader board Page design

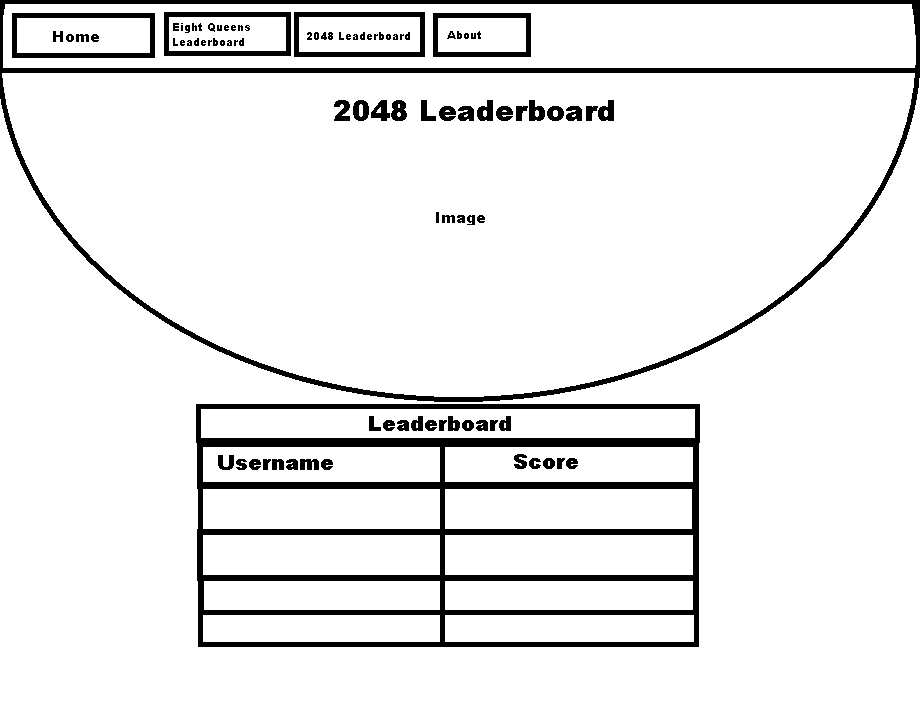


Figure 16 2048 Leader Board Page Design

Above UI mock-up shows the design of the 2048 leader board page (fig 16). The design of this page will be like the design of the 8 queens leader board page (fig 15). However, its leader board will contain 2048 score data, the image will be a screenshot of the 2048 puzzle in game.

#### About Page Design

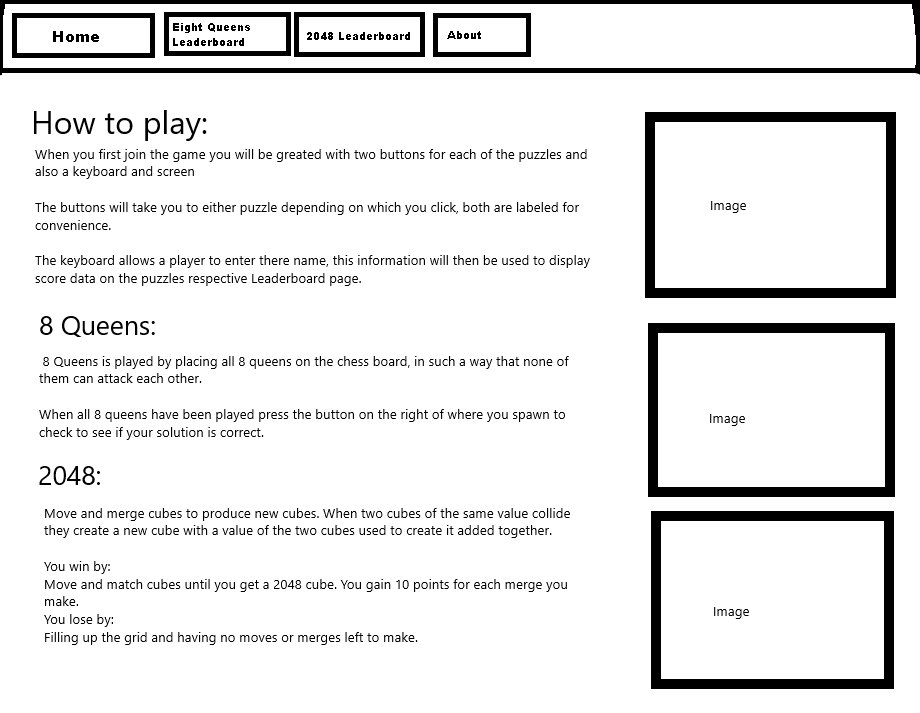


Figure 17 About Page Design

Above UI mock-up shows the design of the about page (fig 17). The page will contain images of the puzzle and a guide on how to play each puzzle, and other general instructions pertaining to the game, like how to input a user’s username.

## Low Level Design

### Class Diagram puzzle vr game (Unity) Part 1

Diagram

Description automatically generated  
Figure 18 VR Game Class Diagram Part 1

This diagram (fig 18) contains classes and the relations between them relating to the 8 queen’s puzzle. The class POST will be used to send data to the web server and the class KeyBoardMain class will be used by queensGame logic to obtain the username provided by the user in the main menu.

### A picture containing diagram Description automatically generatedClass Diagram puzzle vr game (Unity) Part 2

Figure 19 VR Game Class Diagram Part 2

This part of the class diagram (fig 19) contains classes that will be used by the 2048 puzzle and the relationships between them, as with the 8 queens puzzle the POST will be used to send score data to the web server and the KeyBoardMain class will be used to obtain the username inputted by the user in the main menu. Both classes are shared by both puzzles.

### 

### 8 Queens flowcharts:

#### PossibleMovesXAxis function flowchart.

Figure 20 PossibleMovesXAxis function flowchart

Diagram, engineering drawing

Description automatically generatedThere will also be another function like this implemented, that will be almost identical to calculate possible moves on the Y axis, however it will deal will Y coordinates instead of X. It will also use different lists and for example, instead of chessBoardA it will use chessBoard1 list (figure 19):

Chart, line chart

Description automatically generated

Figure 21 Part of PossibleMovesXAxis function flowchart

These lists contain all chess coordinates in a row for a given axis for example:

chessBoardA = new List<string>() { "A1", "A2", "A3", "A4", "A5", "A6", "A7", "A8" }

and chessBoard1 = new List<string>() { "A1", "B1", "C1", "D1", "E1", "F1", "G1", "H1" };

Figure 20 shows a flowchart that describes how the PossibleMovesXAxis function will work. This function gets all possible moves of a given queen on the x axis.

CalcDiagonalTopRight Function.  
Diagram

Description automatically generated  
Figure 22 CalcDiagonalTopRight Function Flowchart

This flowchart describes one of 4 very similar functions. It is used to calculate the diagonal top left possible moves from where the chess piece is placed. The other 3 functions are programmatically very similar. However, in the first two for loops:

Diagram

Description automatically generated

Figure 23 CalcDiagonalTopRight Function Flowchart Loops

The functions will have a different combination of functions (I.E placeInXAxis() and placeInYAxis() functions) these could be any combination of these functions:

placeInXAxis() or placeInXAxisReverced()

placeInYAxis() or placeInYAxisReverced()

And a corresponding list of chess coordinates within the loop so

chessCoordsXAxis or chessCoordsXAxisReverced

chessCoordsYAxis or chessCoordsYAxisReverced

Example in this function (CalcDiagonalTopRight):

Diagram

Description automatically generated

Figure 24 CalcDiagonalTopRight Function Flowchart Loops, Add Chess coordinates.

For example, this is what those loops would look like in the CalcDiagonalTopLeft function  
  
Diagram

Description automatically generated

Figure 25 CalcDiagonalTopLeft Function Example

The rest of the function is identical to the others. Depending on the direction dictates what combination will be used.

#### General 8 queens’ algorithm

Diagram

Description automatically generated  
Figure 26 Basic Game Logic For 8 Queens

This flowchart shows basic game logic for 8 queens. I.E what will happen when the calculate if valid solution button is clicked.

### 2048 game

#### Diagram, engineering drawing Description automatically generatedCube merge and move algorithm 2048

Figure 27 Cube merge and move algorithm 2048 Flowchart

This flowchart shows how a given row on the 2048 grid will be updated to move and merge cubes.

## Reuse of existing code

One major library we will be using is called, Oculus integrations (Meta), in contains all code, and assets required to start developing for the oculus quest 2. This asset pack includes character prefabs, basic VR assets and other such VR related C# scripts. Another framework that will be used in development, is Django, this will be used to create the website.

# Implementation

## Sprint 0:

### Focus:

Create unity project

Import oculus integration package.

Get basic scene set up.

Test to see if all is working using the provided oculus example scenes.

### Burn-down chart:

Chart, line chart

Description automatically generated

Figure 28 Burndown Chart Sprint 0.

### Backlog:

Import oculus integration package. Done

Get basic scene set up. Done

Test to see if all is working using the provided oculus example scenes. Done

## Sprint 1:

Focus:

1. Create Queens class
2. Write Detector script
3. Write Possible moves script
4. Write Queens game logic script

### Burn Down chart:

Chart, line chart

Description automatically generated

Figure 29 Burn Down Chart Sprint 1.

### Backlog:

1. Create Queens class Done
2. Write Detector script Done
3. Write Possible moves script Done
4. Write Queens game logic script Done

### Explanation important code completed in sprint:

To implement 8 queens, an algorithm is required to determine a few things, for example, the current position of a queen, its possible moves, thereby the queens it can attack.

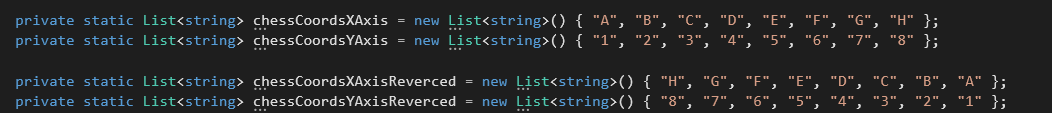
The first set is producing a set of arrays containing all positions on the chess board, I.E one list for each of the rows and columns:

Graphical user interface

Description automatically generated with medium confidence

Figure 30 Chess Coordinate List Part A.

The final two important lists contain the labels for each axis on the chess board and the same in reverse order:

  
Figure 31 Chess Coordinate List Part B.

After this has been implemented it is then possible to perform the required operations to determine which position a given chess piece is in and where it can move/attack.

To determine possible moves, each axis, is calculated separately, for example to calculate the possible moves for a given queen for the X axis this function is employed:

Text

Description automatically generatedFigure 32 PossibleMovesXAxis Code Snippet

This function takes the current Coordinates for a given queen, splits them into a char array, and uses those coordinates to determine every place in can move on the given axis, so if the queens is sitting on an “A” square, it will add all places it can move on that column to the possible moves array, excluding the square that the queens is placed on. Programmatically, the method of calculating possible moves for the Y axis is identical, however it will use a different, array containing all rows positions.

The final set of steps in determining where the queen is placed, is determining possible moves diagonally, as the queen can move diagonally, this is done with a set of 4 functions, one of them is given bellow:

Text

Description automatically generatedThis code work as follows, first the possible positions on the X coordinates are found for the diagonal section and so is the Y.

Next to prevent uneven arrays, which when the coordinates are merged, I.E “A” and “1” to produced “A1” this can happen, this for loop is employed to go through the array and remove any extra Y or X coordinates that have been added.

The beginning of each set of coordinates is then removed to prevent the actual position of the chess piece from being added to the final output list.

The individual coordinates are then merged to produce a list of a set of locations on the chessboard for that diagonal section.

Figure 33 CalcDiagonalTopLeft Code Snippet

The above code shows how the possible moves are calculated for the top left diagonal section. The diagonal possible moves section is split into 4 different sections with the square that the queen is placed on as the centre for example:

A picture containing shape

Description automatically generatedFigure 34 Chess Board Example

Programmatically the function above for calculating the top left diagonal section is very similar to the other diagonal sections as show above, with a few key differences. Depending on the direction of the diagonal line section they may use a different initial coordinate list, for example calcDiagonalBottomLeft(string location) uses chessCoordsXAxisReverced and chessCoordsYAxisReverced lists instead of chessCoordsXAxis and chessCoordsYAxisReverced like in calcDiagonalBottomRight(string location). And use a different combination of the place in axis functions, like with the two examples given above calcDiagonalBottomLeft(string location) uses placeInXAxisReverced(string location) and placeInYAxisReverced (string location) and calcDiagonalBottomRight(string location) uses placeInXAxis(string location) and placeInYAxisReverced(location). Which placeInAxis function used depends on which direction calculateDiagonal function need to calculate coordinate for.

Here is example debug output for the above-mentioned functions with a queen on coordinate “F2”:

Graphical user interface, text

Description automatically generatedFigure 35 All Possible Moves Output Example.

All these functions are called in the getPossibleMoves(string currentLocation) function, this function combines the results for the above functions to create a list with all possible moves for one queen. These steps are repeated for all queens until each queen has a list of all possible positions they can move.

To check if a given queen can attack another this function is employed:

Text

Description automatically generated  
Figure 36 InLineOfSightQueen Code Snippet

If the current queens’ coordinates match one of the coordinates in another queens’ possible moves, then they can attack each other, and the solution given by the user is invalid.

Score is calculating based on the time taken to solve the puzzle, and the score modifier:

Text

Description automatically generated

Figure 37 CalculateScore Code Snippet

Formula: PossibleScore – (timeTaken \* scoreModifyer)

This ensures that the longer time a user takes to come to the solution, the less of a pre-set maximum score they will receive.

### Functional Requirements Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement | Expected result | State of system | Actual Result | Pass/Fail |
| FR1-VR-8Q | User Is able to pick up a chest piece | player is in game, in the 8 queens’ scene | Chess piece was able to be lifted using controller by pressing the grip button | Pass |
| FR2-VR-8Q | Game detects chess piece when placed on a given chess piece on the board, sensor shows the chess queen in sensor check box ticked | player is in game, in the 8 queens’ scene | Sensor detected chess piece, queen in sensor check box ticked | Pass |
| FR3-VR-8Q | Chess pieces are placed in incorrectly on the chess board, the game must register this as an incorrect solution. | player is in game, in the 8 queens’ scene. With puzzle completed | Game flagged the players solution as invalid | Pass |
| FR3-VR-8Q | Chess pieces are placed in correctly on the chess board, the game must register this as a valid solution. | player is in game, in the 8 queens’ scene. With puzzle completed | Game flagged the players solution as valid | Pass |
| FR4-VR-8Q | Timer must start the moment the player enters the puzzle. It must start counting up. | Player clicked the 8 queens puzzle button in the main menu and is taken to the 8 queens puzzle level | Timer stated counting correctly | pass |
| FR5-VR-8Q | Score is calculated based on the time it takes to solve the puzzle, puzzle is solved in 20 seconds, expected result is possibleScore - (timeInput \* scoreModifyer)  Therefore;  1000 – (20 \* 10) = Score = 800 | player is in game, in the 8 queens’ scene. With puzzle completed, clicked submit score button | Score outputted as 800 | pass |
| FR6-VR-8Q | After the 8 queens puzzle is solved, the user is taken back to the main menu in 5 seconds | player is in game, in the 8 queens’ scene. With puzzle completed, clicked submit score button | User was taken to the main menu after 5 seconds. | pass |

Figure 38 Functional requirements testing sprint 1.

## Sprint 2:

### Focus:

1. Set up scene for 2048
2. Crate spawning script for initial 2 cubes
3. Create buttons to move cubes in each direction and code to allow for this
4. Create cubeHandle script to facilitate cube movement and merging
5. Implement system to allow a user to lose
6. Implement system to allow a user to win

### Burn Down chart:

Chart, line chart

Description automatically generated

Figure 39 Burndown chart sprint 2.

### Back-log:

1. Set up scene for 2048 Done
2. Crate spawning script for initial 2 cubes Done
3. Create buttons to move cubes in a given direction and code to allow for this Done
4. Create cubeHandle script to facilitate cube movement and merging Done
5. Implement system to allow a user to lose Done
6. Implement system to allow a user to win Done

### Explanation important code completed in sprint:

There is one main function within the 2048 code that facilitates movement and cube merges:

This function has 2 applications. Its main function is it calculates a row of cubes moves I.E if they can merge or move, and this can be used it 2 ways:

The first is facilitating cube movement and merging, when the flagAllowUpdateGrid flag is set to true, the cubes will move on the grid based on the functions calculated state of a row.

The second use case is to detect whether moves and merges are possible in each row and column, if there are moves and merges possible the function will return false, otherwise true to indicate no changes to the row can be made, when the flagAllowUpdateGrid flag is set to false this can be used to check whether or not the player has lost without actually making any physical moves on the grid.

Text

Description automatically generatedThis first section of the code figures out what cubes can be merged in each row based on a few rules, the first is the cubes must have identical values and the second is that if a cube of a different value is in between two cubes of the same value a merge can’t occur.

Figure 40 cubeMoveMerge Code Snippet.

Instead of creating a function that moves and merges cubes as an when it finds a match, we opted to calculate all possible movements and merges in one go and then update the hole row at once. This was done due to synchronisation issues that occurred when changes to the grid were done as an when the algorithm determined there needed to be a merge or cube move, this is because when merging the old two cubes, must be deleted, when this happened, the algorithm, still thinks the cube still exists and tries to delete it again causing an error. This method of implementation is one way to avoid this error. The win condition is checked as the program is checking for merges, if the addition of two cubes equates to 2048 then the player has one, however before the player is notified of this the algorithm finishes running and the board updated with the new cubes and locations to allow the user to see the winning cube on the board otherwise they would get the win message without the 2048 cube appearing on the board.  
Text

Description automatically generated

Figure 41 Figure 41 Figure 41 MoveMerge Code Snippet Part 2

The second part of this function as displayed above facilitates, the movement of cubes, the first part however if(flagMergesMade == false) determines where it is needed to use the output list created by merging, if no merges took place then the original input list can be operated on to facilitate movement, this is done by removing all empty spaces from the list and appending them to the end of the list. This function is universal and can be used on every row and column, irrespective of the direction the user chooses to move I.E up, down, left, or right.

The code bellow is the button script for the left button, when this button is pressed the cubes will be moved and merged to the left:

Figure 42 LeftButton Code Snippet.

Text

Description automatically generatedThe first set of function calls populates the corresponding lists with the UpToDate positions of the cubes in the hole grid.

If the button is pressed then possible moves and merges is calculated for all rows and columns, however only the cubeMoveMerge() that are operating on rows or columns that correspond with the button pressed are actually updated in game on the grid.

This is done to allow for lose condition checking, no matter which direction use user selects to move, the reason this is checked in the code of each button as is the other way of doing this is continuously checking the board to check to see if the

player has lost. This may have a great impact on performance of the game, therefore, it should only be check when any of the directional buttons are pressed.

As can also be seen from the code above the Invoke(“AllowForButtonPress”, 2); function is used to allow for a delay in button presses. This is done to avoid the button being pressed multiple times unintentionally if the user leaves the controller inside the button for two long, allowing this to happen resulted in the game crashing during testing.

Score in this puzzle is calculated by adding ten to the score for each merge the user completes.

### Functional Requirements Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement | Expected result | State of system | Actual Result | Pass/Fail |
| FR7-VR-2048 | When left button is clicked all cubes move to the left on the board. | 2048 puzzle loaded; game is ready to play | All cubes moved in the correct direction. | pass |
| FR7-VR-2048 | When right button is clicked all cubes move to the right on the board. | 2048 puzzle loaded; game is ready to play | All cubes moved in the correct direction. | pass |
| FR7-VR-2048 | When down button is clicked all cubes move down on the board. | 2048 puzzle loaded; game is ready to play | All cubes moved in the correct direction. | pass |
| FR7-VR-2048 | When up button is clicked all cubes move up on the board. | 2048 puzzle loaded; game is ready to play | All cubes moved in the correct direction. | pass |
| FR8-VR-2048 | Cubes must merge in the direction of the button the user clicked. Two “2” cubes are merged to produce 4. Button tested left. | 2048 puzzle loaded; game is ready to play. | Cubes merged correctly to produce a 4 cube in their place. | pass |
| FR8-VR-2048 | Cubes must merge in the direction of the button the user clicked. Two “2” cubes are merged to produce 4. Button tested right. | 2048 puzzle loaded; game is ready to play. | Cubes merged correctly to produce a 4 cube in their place. | pass |
| FR8-VR-2048 | Cubes must merge in the direction of the button the user clicked. Two “2” cubes are merged to produce 4. Button tested up. | 2048 puzzle loaded; game is ready to play. | Cubes merged correctly to produce a 4 cube in their place. | pass |
| FR8-VR-2048 | Cubes must merge in the direction of the button the user clicked. Two “2” cubes are merged to produce 4. Button tested down. | 2048 puzzle loaded; game is ready to play. | Cubes merged correctly to produce a 4 cube in their place. | pass |
| FR9- VR-2048 | Once no new movements of cubes or merges can be made the user gets a message saying they have lost. | 2048 puzzle loaded; game is ready to play, no more merges or cube movements can be made | Game was declared a loss. | pass |
| FR10-VR-2048 | After two 1024 cubes are merged the user is show a win message. | 2048 puzzle loaded; game is ready to play, two cubes are about to be added to make 2048 | Two 1024 cubes are added to make 2048, user was notified of the victory | pass |
| FR11-VR-2048 | 3 cubes are merged to create a score of 30. | 2048 puzzle loaded; game is ready to play, two cubes are about to be added. | Cubes are combined and score is now 30. | pass |
| FR12- VR-2048 | After the game is won, button appears to take user back to the main menu | 2048 game won | Button appeared to take user back to main menu. | pass |
| FR12- VR-2048 | After the game is lost, button appears to take user back to the main menu | 2048 game won | Button appeared to take user back to main menu. | pass |

Figure 43 Functional Requirement Tests Sprint 2.

## Sprint 3:

### Focus:

1. Install Django using anaconda
2. Set up database models
3. Set up static files
4. Create html and CSS for Home page
5. Create html and CSS for eight queen’s leader board
6. Create html and CSS for 2048 leader board
7. Create html and CSS for about page
8. Set up views create and create views for all pages, including setting up urls.
9. Add code to each view for the leader boards pages to pull score data from the database and display on their respective webpages.
10. Set-up Rest API and Create POST class within

### Burndown chart:

Chart, line chart

Description automatically generated  
Figure 44 Burn Down Chart Sprint 3.

Backlog:

1. Install Django using anaconda Done
2. Set up database models Done
3. Set up static files Done
4. Create html and CSS for Home page Done
5. Create html and CSS for eight queen’s leader board Done
6. Create html and CSS for 2048 leader board Done
7. Create html and CSS for about page Done
8. Set up views create and create views for all pages, including setting up urls. Done
9. Add code to each view for the leader boards pages to pull score data from the database and display on their respective webpages. Done
10. Set-up Rest API and Create POST class within Done

### Explanation important code completed in sprint:

Figure 45 REST API Code Snippet

Text

Description automatically generatedHere are the two rest api classes for both puzzles, each class contains only handles post requests as the game does not require get requests as not data is required from the database by the game. The code accepts post requests and then takes the username from the json data and stores it in the database to be utilised by the website, to display user scores.

Some code is shared between the two puzzles most notably the POST class which allows score data to be sent to the webserver and stored in the database.

Text

Description automatically generatedAs can be seen from the code, the is facilitated using the System.Net class, allowing json data to be sent to the webserver in the form of a http Web Request.

Sent is both the username of the player and the score they obtained in the respective puzzle.

Figure 46 HTTPRequest Code Snippet

### Functional Requirements Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement | Expected result | State of system | Actual Result | Pass/Fail |
| FR13-VR/WS-8Q | After 8 queens puzzle is solved, score data and username are sent to the webserver and scored in the database  (The username in this test is hard coded)  Test data:  Username: “Test1”  Score Data: “1000” | 8 queens puzzle is solved | 8 Queens score and username was sent to the database | pass |
| FR14-VR/WS-2048 | After 2048 puzzle is lost, score data and username are sent to the webserver and scored in the database  (The username in this test is hard coded)  Test data:  Username: “Test2”  Score Data: “300” | 2048 puzzle lost, no moves or merges to be made, user is notified of lost | Score data and username received by headset | pass |
| FR15- VR/WS-2048 | After 2048 puzzle is won, score data and username are sent to the webserver and scored in the database  (The username in this test is hard coded)  Test data:  Username: “Test3”  Score Data: “400” | 2048 puzzle won, two 1024 cubes added to win the game | Score data and username received by headset | pass |
| FR16-WS | Navigation bar home button takes user to the website home | User is currently on the 8 queens leader board page | After clicking the home button user is taken to the home page | pass |
| FR16-WS | Navigation bar 8 queens leader board button takes user to the 8 queens leader board page | User is currently on the home page | After clicking the 8 queens leader board button user is taken to the queens ‘leader board page | pass |
| FR16-WS | Navigation bar 2048 leader board button takes user to the 2048 leader board page | User is currently on the home page | After clicking the 2048 leader board button user is taken to the 2048 leader board page. | pass |
| FR16-WS | Navigation bar about button takes user to the about page | User is currently on the home page | After clicking the about button user is taken to the about page | pass |
| FR17-WS | On the 8 Queens leader board page score and usernames should be pulled from the database and displayed on the page.  Test data:  Username “test” score “1000” | User is currently viewing the 8 queens leader board page | User can view score data pulled from the database; test data is visible on webpage | pass |
| FR18-WS | On the 2048 leader board page score and usernames should be pulled from the database and displayed on the page.  Test data:  Username “test2” score “300” | User is currently viewing the 2048 leader board page | User can view score data pulled from the database; test data is visible on webpage | pass |

Figure 47 Functional Requirements Tests Sprint 3.

## **Sprint 4:**

### Focus:

1. Create menu scene
2. Create two buttons in scene allowing travel to the 8 Queens puzzle and 2048 puzzles
3. Implement keyboard in menu scene so the user can input their name

### Burn-Down chart:

Chart, line chart

Description automatically generated

Figure 48 Burn Down Chart Sprint 4.

### Backlog:

1. Create menu scene Done
2. Create two buttons in scene allowing travel to the 8 Queens puzzle and 2048 puzzles Done
3. Implement keyboard in menu scene so the user can input their name Done

### Explanation important code completed in sprint:

Text

Description automatically generatedThis code is a small extract from the keyboards on trigger enter function, this script is attached to the controller. When the controller collides with a given key is corresponding letter or number is concatenated onto the typed Text string variable, allowing user input to be collected and due to this being a public static variable, this variable can be accessed in other scenes allowing it to be used when score data needs to be sent to the webserver.

Figure 49 Keyboard Code Snippet.

Here is an image of said keyboard.A screenshot of a video game

Description automatically generated with medium confidence  
Figure 50 Image Of In Game Keyboard

### Functional Requirements Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement | Expected result | State of system | Actual Result | Pass/Fail |
| FR13-VR-Menu | When the 8 queens’ button is clicked, the user is taken to the 8 queens puzzle | User is in main menu | User is taken to the 8 queens puzzle | pass |
| FR14-VR-Menu | When the 2048 button is clicked, the user is taken to the 2048 puzzle | User is in main menu | User is taken to the 2048 puzzle | pass |
| FR15-VR-Menu | The user can type the username “user1” and it appears on the in-game screen. | User is in main menu | User data appeared after being typed out using the in-game keyboard. | pass |
| FR15-VR-Menu and FR14-VR/WS-2048 | 8 Queens - Username is retrieved from the main menu keyboard and sent with the score data to webserver | User in 8 queens with puzzle complete | Username is retrieved from menu and sent along with score data to webserver | pass |
| FR15-VR-Menu and FR14-VR/WS-2048 | 2048 - Username is retrieved from the main menu keyboard and sent with the score data to webserver after the user loses and presses the main menu button | User in 2048 with puzzle Lost | Username is retrieved from menu and sent along with score data to webserver | pass |
| FR15-VR-Menu and FR14-VR/WS-2048 | 2048 - Username is retrieved from the main menu keyboard and sent with the score data to webserver after the user wins and presses the main menu button | User in 2048 with puzzle Won | Username is retrieved from menu and sent along with score data to webserver | pass |

Figure 51 Functional Requirements Test Sprint 4.

## Non-functional requirements testing:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement | Expected result | State of system | Actual Result | Pass/Fail |
| N-FR3 | The game must be able to run at a reasonable frame rate on the quest 2, 60fps minimum | Game loaded in puzzle | Game achieves above 60 fps | pass |
| N-FR5 | The game should be able to check the validity of a solution without any long durations (over 5 seconds) of lag. | Game loaded in puzzle 8 Queens | Only a brief second of lag when the solution validity is being calculated | pass |
| N-FR5 | 2048 checking if lost. The game should be able to check the validity of a solution without any long durations (over 5 seconds) of lag. | Game loaded in puzzle 2048 | No lag present when testing, game runs smoothly, no delay when moving and merging cubes due to prior calculation of win loss state | pass |

Figure 52 Non-Functional Requirement Tests

# Project Evaluation

This section will include a full project evaluation, the issues that were faced, the solutions to those issues and what improvements could be applied to make the product better will be discussed here.

## Research

During the research section, we researched many topics to do with VR, how to implement it and the technologies we would use to do so. However, the research was a little too narrow in some areas, particularly in terms of game engines, and web technologies, for example, likely due to personal bias, research was focused on unity, and web frameworks for the programming language python. This was not a wise strategy as it is likely that better solutions could have existed and would have been found if the search criteria were broadened.

## Requirements

The methodology used however for prioritizing requirements “MoSCoW” proved to be very useful as it made planning sprints much easier, as identifying priority requirements before development, made important features that must have been implemented first clear. Furthermore, it resulted in a fully playable product as no time was wasted implementing less important features. Requirements section does have some flaws mainly, although all the “must have” requirements were implemented and tested, they “should have” requirements and “could have” requirements were not implemented due to time constraints.

## Methodology

As discussed in the methodology section the methodology selected was agile scrum. Even though they method is usually used in groups, it was found that during this project it worked quite well with solo work as well. It was found that structuring development in this way helped to organised and implement work efficiently, setting time frames for sprints with set work to do within them, made the large amount of work that was needed to be done easier to handle and manage within those sprints.

## Design

In the case of this project, time management was of the utmost importance, making the design section extremely important as, not being prepared in this area when implementing could result in time being wasted, due to poor planning. Due to this, this section could be considered a success as because of the diagrams, the implementation went smoothly and efficiently. Particularly the flow charts in this section proved very useful making some of the more complex algorithms in the project much quicker and easier to implement.

## Implementation and testing

During the development of the game, we obtained much feedback from the people that tested it, one of major issues that users faced was if they were completely new to VR, it was very difficult for them to play the game, without instructions from someone with more experience, this is likely because with the headset on, you can’t look at the controls for reference. This issue could be rectified with an addition of a tutorial with general instructions about how to operate the headset, such as controls. Controls in VR games as was found during research and testing of different VR games, tend to be similar. For example, the grip button is used to grab objects in most VR games. Due to this fact, this product, because of its relatively simple concept could be used, and adapted as more of a learning aid, to allow users to get used to the basics of VR as because of its simple gameplay, it is not very intense.

Another major point that was discovered when allowing users to test is that, as a developer, We did not consider the possibility of users playing the game unlike we would, for example, when we play we know exactly what we are doing, as the games developer we know that to play 8 queens, for example we know that we need to place all the chess pieces on the chess board so they can’t attack each other, so we didn’t account for situations in which the user would go against this directive and walk up to a virtual wall and sticking their head through it seeing the incomplete world outside of the game area. Although at first this does not sound like an issue, it brought on a realisation that there are bugs and flaws in my game that we may not have been able to find because when we test, we are testing to see if a given feature works. Not if a feature can be used to break the game. This, it indicates that our approach to testing was flawed, as we only tested my game against the requirements of the project, and only used external testers after I had finished the large majority of implementation. A better approach would have been to give the game to external testers after each sprint, so any unusual bugs that they found could be fixed in the next. This testing method would have likely resulted in a more polished game with fewer bugs.

## Supervisor Feedback and PiP Day

During the project, feedback from both the project supervisor and the projects second marker on project in progress day. Advice from both people were implemented during the creation of the project, for example the supervisor’s advice on research ideas and game features were both utilised and advice from the second marker regarding external testers. Due to this it is believed that this advice was utilised effectively.

# Further Work and Conclusions

# Conclusion

To conclude this report, it should be stated although a lot of things went very well, a few things did not. The main project aim, to create a VR game that has two puzzles with a web site to display scores was a success. This was all done using unity and Django (python). Extending this would not be difficult in the future, as the game is set up in a way where adding new puzzles would be very easy. For example, using readymade unity prefabs and code that were created during the course of the project for other puzzles. This report and the game implemented gave an in depth look into the rising technology VR and demonstrated one way in which it could be applied and used. During the reports research section other applications for VR were identified, VR as an emergent technology has a lot of potential applications because of this future work on this project would allow for further understanding of VR capabilities and limits.

## Future work

Although due to time constraints only the main requirements, the “MoSCoW” “Musts” were implemented, given more time, further additions could have been added. For example, if work on this project were to continue, more puzzles should be added to the game. More features like a login system for the website and the VR headset allowing players to update their existing scores rather than simply adding a new one to the leader board. Furthermore, creating a version of the game that will also run on desktop based VR headsets like the Oculus Rift S is another idea that could be considered.

# References / Bibliography

Green = peer reviewed

Purple = books

Blue = trusted websites

Orange = unsure if peer reviewed

Red = Not peer reviewed

Yellow = Video games

Brown = Unused/Not finished with

**Temp Colour coding**

1. Souza, Redmiles, C.R.B.S. , D.F.Redmiles. (2009) On the Roles of APIs in the Coordination of Collaborative Software Development. Computer supported cooperative work [online]. Volume 18, Page 445-475. [Accessed 21 December 2021].
2. Kotstein, Justus, S.K, J.B. (2021) Which RESTful API Design Rules Are Important and How Do They Improve Software Quality?. A Delphi Study with Industry Experts[online]. [Accessed 21 December 2021].
3. Holly, Pirker, Resch, Brettschuh, Gutl, M.H, J.P, S.R, S.B, C.G. (2021) Designing VR Experiences--Expectations for Teaching and Learning in VR. Educational technology & society [online]. [Accessed 27 December 2021].
4. Pourmand, Davis, Marchak, Whiteside, Sikka, A.P, S.D, A.M, T.W, N.S. (2018) Virtual Reality as a Clinical Tool for Pain Management. Current pain and headache reports [online]. Volume 22 [Accessed 27 December 2021].
5. Kumar, G.K, Bhatia, P.K.B (2012) Impact of Agile Methodology on Software Development Process, International Journal of Computer Technology and Electronics Engineering [online]. Volume 2[Accessed 31 December 2021].
6. Hron, M.H, Obwegeser, N.O (2021) Why and how is Scrum being adapted in practice: A systematic review, Journal of Systems and Software[online]. Volume 183[Accessed 31 December 2021].
7. Tashjian, V.C.T. (2017) Virtual Reality for Management of Pain in Hospitalized Patients: Results of a Controlled Trial [online]. Volume 4, pages e9-e9 [Accessed 27 December 2021].
8. Zhang, Z.Z. (2012) Microsoft Kinect Sensor and Its Effect. IEEE Computer Society [online]. Volume 19, pages 4-12 [Accessed 28 December 2021].
9. [Shotton](https://ieeexplore.ieee.org/author/37550019800), J.S, [Fitzgibbon](https://ieeexplore.ieee.org/author/37282639100), A.F, [Cook](https://ieeexplore.ieee.org/author/37951907600), M.C, [Sharp](https://ieeexplore.ieee.org/author/37408379000), T.S, [Finocchio](https://ieeexplore.ieee.org/author/37944916600), M.F, [Moore](https://ieeexplore.ieee.org/author/37950527600), R.M, [Kipman](https://ieeexplore.ieee.org/author/37944916100), A.K , [Blake](https://ieeexplore.ieee.org/author/37282602200), A.B . (2011) Real-time human pose recognition in parts from single depth images. Anon, [CVPR 2011](https://ieeexplore.ieee.org/xpl/conhome/5968010/proceeding) , Colorado Springs, CO, USA, 2011, IEEE [Accessed 28 December 2021].
10. Frese, U.F, Wagner, R.W, Rofer, T.R. (2010) A SLAM overview from a user’s perspective. KI [online]. Volume 24, pages 191-198 [Accessed 28 December 2021].
11. Huang, B.H, Zhao, J.Z, Liu, J.L. (2019) A Survey of Simultaneous Localization and Mapping with an Envision in 6G Wireless Networks. CoRR [online]. Volume abs/1909.05214[Accessed 28 December 2021].
12. [Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), [R. Sherman](https://learning.oreilly.com/search/?query=author%3A%22William%20R.%20Sherman%22&sort=relevance&highlight=true),  [D. Will](https://learning.oreilly.com/search/?query=author%3A%22Jeffrey%20D.%20Will%22&sort=relevance&highlight=true), [A.C](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true) , [W.S](https://learning.oreilly.com/search/?query=author%3A%22William%20R.%20Sherman%22&sort=relevance&highlight=true), [J.W](https://learning.oreilly.com/search/?query=author%3A%22Jeffrey%20D.%20Will%22&sort=relevance&highlight=true) (2009) Developing virtual reality applications : foundations of effective design [online]. Location: Elsevier Science & Technology. [Accessed 06 November 2021].
13. Yudin, A.Y. (2020) Building Versatile Mobile Apps with Python and REST: RESTful Web Services with Django and React [online]. No place: Apress. [Accessed 13 November 2021].
14. Grinberg, M.G. (2018) Flask web development: developing web applications with Python

[online]. Second edition. No place: O’Reilly. [Accessed 13 November 2021].

1. [Uzayr](https://learning.oreilly.com/search/?query=author%3A%22Sufyan%20bin%20Uzayr%22&sort=relevance&highlight=true), S.B.U. (2016) Learning WordPress REST API [online]. Basingstoke: Palgrave Macmillan. [Accessed 05 December 2021].
2. Lubbers, Albers, Salim, P.L, B.A, F,S. (2011) Pro HTML5 Programming[online]. no place: [Apress](https://learning.oreilly.com/library/publisher/uuid/42edee2f-ff66-471b-b851-c41366e2b462). [Accessed 25 December 2021].
3. Gibson, W.G. (1993) *Virtual Light*. US, UK, Canada: Bantam Spectra, Viking Press, Seal Books.
4. Cline, E.C. (2011) *Ready player one*. US : Crown Publishing Group.
5. Hesch, Kozminski, Linde. J.H, A.K, O.L. (2019) Powered by AI: Oculus Insight. Available from: https://ai.facebook.com/blog/powered-by-ai-oculus-insight/ [Accessed 21 December 2021].
6. MathWorks. (no date) Powered by AI: Oculus Insight. Available from: https://uk.mathworks.com/discovery/slam.html#slam-with-matlab [Accessed 21 December 2021].
7. Sherman, [B.Craig](https://learning.oreilly.com/search/?query=author%3A%22Alan%20B.%20Craig%22&sort=relevance&highlight=true), W.S, A.C. (2003) Understanding Virtual Reality: Interface, Application, and Design [online]. Location: Morgan Kaufmann. [Accessed 06 November 2021].
8. Chu, Zaman, E.C, L.Z. (2021) Exploring alternatives with Unreal Engine’s Blueprints Visual Scripting System. Entertainment Computing [Online]. Volume 36[Accessed 13 November 2021]
9. Dionisio, III, Gilbert, J.D, W.I, R.G. (2013) 3D Virtual worlds and the metaverse: Current status and future possibilities. ACM computing surveys [online]. Volume 45, Issue 3. [Accessed 21 December 2021].
10. Taketomi, T.T, Uchiyama, H.U, Ikeda, S.I (2017) Visual SLAM algorithms: a survey from 2010 to 2016. IPSJ transactions on computer vision and applications [online]. Volume 9. [Accessed 28 December 2021].
11. Asghar, A.R.A, Bhatti, S.N.B, Tabassum, A.T, Shah, S.A.A.S (2017) The Impact of Analytical Assessment of Requirements Prioritization Models: An Empirical Study International Journal of Advanced Computer Science and Applications [online]. Volume 8. Page 303-313 [Accessed 28 December 2021].
12. [Mishra](https://www.ijimai.org/journal/bibcite/contributor/5998), [Shrawankar](https://www.ijimai.org/journal/bibcite/contributor/5999), [P.M](https://www.ijimai.org/journal/bibcite/contributor/5998), [U.S](https://www.ijimai.org/journal/bibcite/contributor/5999). (2016) Comparison between Famous Game Engines and Eminent Games. International journal of interactive multimedia and artificial intelligence [online]. Volume 4, page 69. [Accessed 13 November 2021].
13. Game engine. (*2021*) Wilipedia The free Encyclopedia [online]. 24 December. Available from: https://en.wikipedia.org/wiki/Game\_engine [Accessed 25 December 2021].
14. GameLabGraz (No Date) Maroon [Video game]. Graz University of Technology. Available from: https://maroon.tugraz.at/ [Accessed 27 December 2021].
15. Monstars Inc. Resonair and Stage Games (2020) Tetris® Effect: Connected [Video game]. Enhance. Available from: https://www.oculus.com/experiences/quest/3386618894743567/?locale=en\_GB [Accessed 01 January 2022].
16. Schell Games (2021) I Expect You To Die 2 [Video game]. Schell Games. Available from: https://www.oculus.com/experiences/quest/2970998659623177 [Accessed 01 January 2022].
17. Meta (No Date) *Oculus integrations* (37.0) [Unity asset Pack]. Available from: https://assetstore.unity.com/packages/tools/integration/oculus-integration-82022 [Accessed 03 March 2022].

# Appendix A: First Appendix