University of Plymouth

School of Engineering,

Computing, and Mathematics

PRCO304

Final Stage Computing Project

2019/2020

High Ground VR

Jack Griffiths

10547816

BSc (Hons) Computing and Games Development

Acknowledgements

Firstly, I'd like to thank Mr James Hayter for his support and feedback over the course of this project, which continued even during a global pandemic. I would like to thank the entire ISS staff for their continued support at all stages of university, especially to my personal tutor Dr Dan Livingstone.

I would like to thank my family and friends for their unwavering support throughout everything that's happened during my time at university.

Finally, I'd like to thank my classmates, especially Aden Webb, Jack Brewer and Avebry Houghton -Vowles for continuously testing this project and providing feedback.

Abstract

This report covers the development of High Ground VR, a single player tower defence game in virtual reality. The project was undertaken as an exploration into the possibilities of a virtual reality tower defence game, aiming to approach this using recognised interactions and input methods. It was also targeted as a challenging programming project, with quite a large scope considering the 14-week development time.

This report begins with laying out the aims and objectives of the project and goes on to compare key players that fall within this genre. The basic functional requirements of the project are laid out, which were written after an extensive Game Design Document had been completed. The methodologies and project management approach are also discussed, as to describe the method of approach to completing the game.

The main body of the report covers the actual development of the project, broken down into weekly sprints. The architecture of the project is also outlined, and the various iterations of the game's systems explored. This section aims to present the implementation behind every feature and provides code and game screenshots to aid in explanation.

The final section of the report reviews the success of the project's objectives, through appraisal and reflection. Performance graphs are also present, which aim to provide some quantitative data to evidence the success of the project management and its development. The report ends with conclusions on the project's success and what it discovered about the genre of virtual reality tower defence.

Table of Contents

Table of Figures

Word Count

9881

Code Link

https://github.com/JackDotGriffiths/PRCO304-High-Ground

Look at Appendix 1 for more links relating to the project

1 - Introduction

1.1 - Overview

High Ground VR is a single player virtual reality tower defence game built for the HTC Vive VR system, using Unity and SteamVR. The game takes place on a hexagonal terrain, on which the player can choose to view scaled down on the board itself or scaled up as if the game it's taking place in miniature. The player has a set amount of time during a building phase to build defences to protect the centre of the map, and then enemies start to spawn.

1.2 - Audience

This project aims to fall within the PEGI 7 category, as it appeals to a wide audience with its minimal implied violence and cartoon style. The main reason to pursue this rating was that the game can be played in more of a 'couch co-op' setting. This of course means that players require an HTC Vive to play the game, as well as a computer that can run VR programs at high frame rates [1]. The project is intended as an exploration into tower defence, rather than a game devoted to it, so the game may not appeal to absolute fans of the genre. However, the market of tower defence in virtual reality is relatively unsaturated, so it will still appeal to those players looking for something new.

1.3 - Purpose

The aim of High Ground VR is to explore a balance between strategic tower defence combat and challenging board game mechanics, through the interactive technology available through Virtual Reality. The game also aims to research the gameplay opportunities available through unpredictable enemies in a more strategic setting, and how the player will react to more stressful situations. This research came in the form of usability testing at various stages of the project (See Appendix 10).

2 - Background, Objectives and Deliverables

2.1 - Project Background

Virtual Reality (VR) can be defined as a simulated experience that can be similar to, or completely different from, the real world. VR systems usually use headsets to create realistic images, sounds and other sensations that stimulate a user's physical presence in a virtual environment. The VR market is growing, at rapid pace, and has become an increasingly accessible medium for players and developers to experience games (*Virtual Reality (VR)* -

[1] https://www.vive.com/uk/support/vive/category_howto/what-are-the-system-requirements.html : HTC Vive system requirements

Statistics and Facts, 2020). This project aims to take inspiration from key players within the VR industry, and replicate interactions and art styles that players may be familiar with.

The genre of tower defence is very well established in gaming. In short, the goal of tower defence games is to defend a player's possessions by obstructing incoming enemy attackers, through placing defensive structures on or along their path of attack. This genre of games is known for being strategic, requiring players to think methodically about their next plan of action. Hence, tower defence games require longer periods of gameplay. The typical adaptation of the genre is not appropriate for virtual reality, as most players of VR are uncomfortable wearing the headset for too long. This project aims to explore the possibilities of this genre within VR, and how it can be adapted for shorter periods of gameplay.

2.2 - Competitor Analysis

Figure 1 below shows a brief competitor analysis of similar titles.

Title	Platform	Differentiating Mechanics	Art Style	Average Level Time
Castle Must Be Mine	SteamVR	● Command a hero to attack enemies ● Use controllers to throw attacks ● Cast traps to aid in defence	● Cartoon ● Bright Colours ● Simplistic lends well to power limitations of VR	5-10 minutes
Alchemists Defender	SteamVR	● Teleporting down onto the game board ● FPS Style combat with enemies ● Building relocation	● Medieval ● Dark Lighting ● Realism	5-10 mins
Hex Defense	SteamVR	● Hexagonal Terrain ● Large game board ● Huge array of defences	● Sci-Fi ● Modern Warfare ● Realism	5-10 mins

Figure 1: Brief competitor analysis of similar titles

For a financial viability market research document, see Appendix 11.

2.3 - Project Objectives and Deliverables

The objectives of the project are as follows:

- Deliver a bug free gameplay loop, with one session lasting between five and ten minutes.
- Explore the possibilities of a VR tower defence game.

- Create a cohesive user experience through art, audio and VR interactions.
- Produce original 3D and 2D assets and use appropriately licensed third-party audio assets.

The result of this project will be a vertical slice of gameplay, on top of which further gameplay could be developed. The game will be bug free, provided as an '.exe' file, and will function with an HTC Vive. The entire development of High Ground VR will be logged and analysed, including weekly reports and social media posts.

2.4 - Functional Requirements

These functional requirements for the project were written alongside the game design document (See Appendix 4).

- Basic Virtual Reality viewer and controller tracking
- Generated hexagonal Game Board
- Randomly placed enemy spawns
- Player can place buildings
- Teleport down onto the game board and around the hexes
- Building and Attack phase announced
- Enemy Spawning
- A* Pathfinding
- Combat
- Sieges
- Game over sequence
- Physical button UI for building selection
- Point and click UI for restarting the game
- Player gets a score

For more in-depth functional requirements, see Appendix 6.

3 - Method of Approach

3.1 - Methodologies

There were a variety of methodologies used in the development of this project. Agile development strategies were used, meaning feature implementation was iterative and incremental. Development was test driven, as continuous functionality and usability testing led every stage of the project. New features were written with many exposed parameters, so they could be tested and altered during runtime of testing the game.

The project management used a Kanban workflow. Kanban is a minimalistic and visual methodology to project management that enlists ‘To Do’, ‘Doing’ and ‘Done’ columns on which tasks are moved between throughout the sprint (Brechner, 2015).

3.2 - Technologies

For a breakdown of the software and hardware used during the development of this project, see Appendix 12.

3.2.1 - A Pathfinding*

A* Pathfinding is the primary technology used to drive this project. It’s one of the most used pathfinding algorithms, as it’s relatively efficient and very reliable (Bourg and Seemann, 2004). A* is a best-first search, so it creates a weighted graph, starting from a specific node. It aims to find the path to a goal node, by taking the path of smallest cost. On each iteration of its main loop, it determines which of its paths it needs to explore. It does this based upon the cost of the path and an estimate of the cost required to extend the path all the way to a goal node. A* aims to minimise the following:

$$f(n) = g(n) + h(n)$$

where;
n = next node on the path
g(n) = cost of the path from the start node to n
h(n) = heuristic function that estimates the cost of the cheapest path from n to the goal

The implementation of A* Pathfinding in this project uses a custom heuristic distance built for hexagonal grids. The distance between nodes is equal to the greatest of the absolute values of : the difference along the x-axis, the different along the y-axis, or the difference of these two differences (Drake, 2010). In addition to this calculation, a cost is added based upon the status of the node, e.g. nodes that involve combat should be avoided.

3.3 - Project Management Approach

The entire 14-week development of this project was broken down into phases, each of which contained up to three one-week long sprints. The phase lengths were determined by the project functional requirements list (See Appendix 6). Phases were defined within a ‘Roadmap’ column, which contained a functionality list to be completed within that phase. At the start of a new phase, all required tasks were created. These remained in ‘Backlog’, or ‘Approved from Backlog’ if they had been fleshed out.

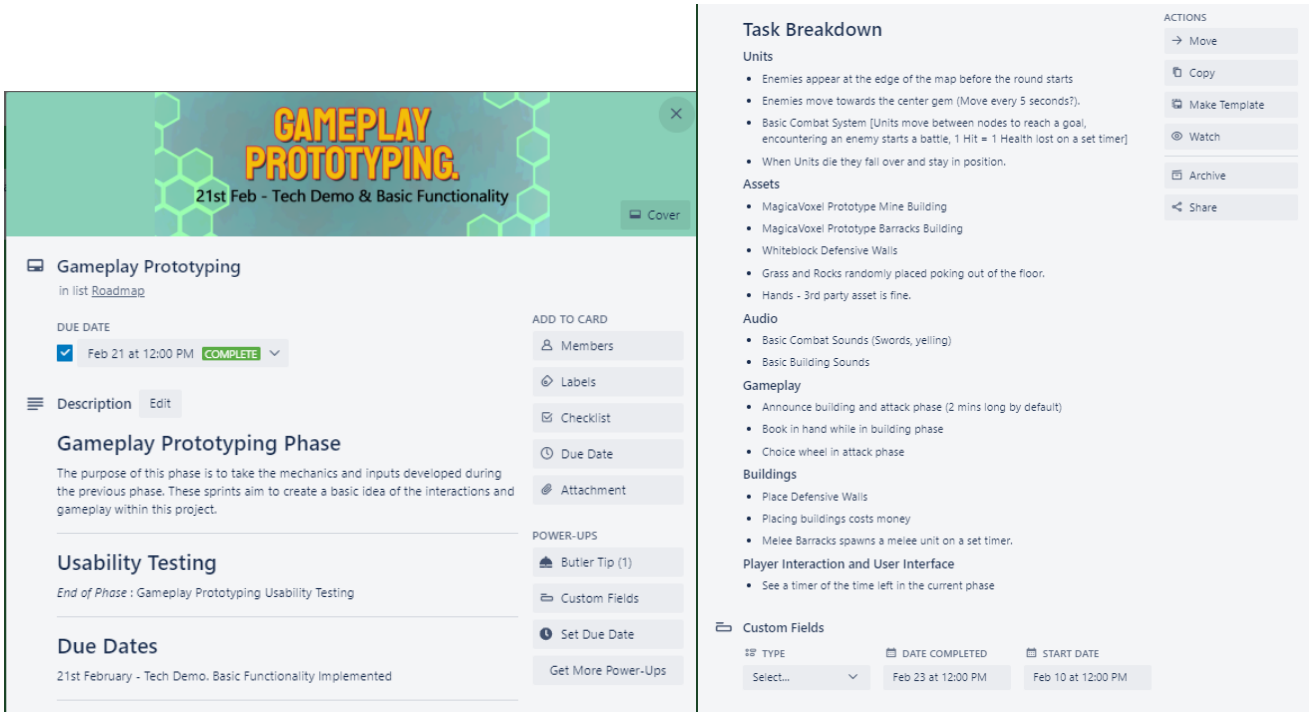


Figure 2: An example task from the project roadmap

Each sprint consisted of a list of approximately five tasks which were to be completed during that week, alongside a sprint conclusion checklist (See Appendix 7). This checklist was integrated into the development cycle as it helped ensure the project was tested to a suitable standard and that progression was logged. When a sprint was complete, it was archived in a separate board so that it was easier to track the progress of development.

As clear in figure 3, All the tasks created in the Trello board were given due dates, descriptions, estimated points value (hours) and were categorised accordingly.

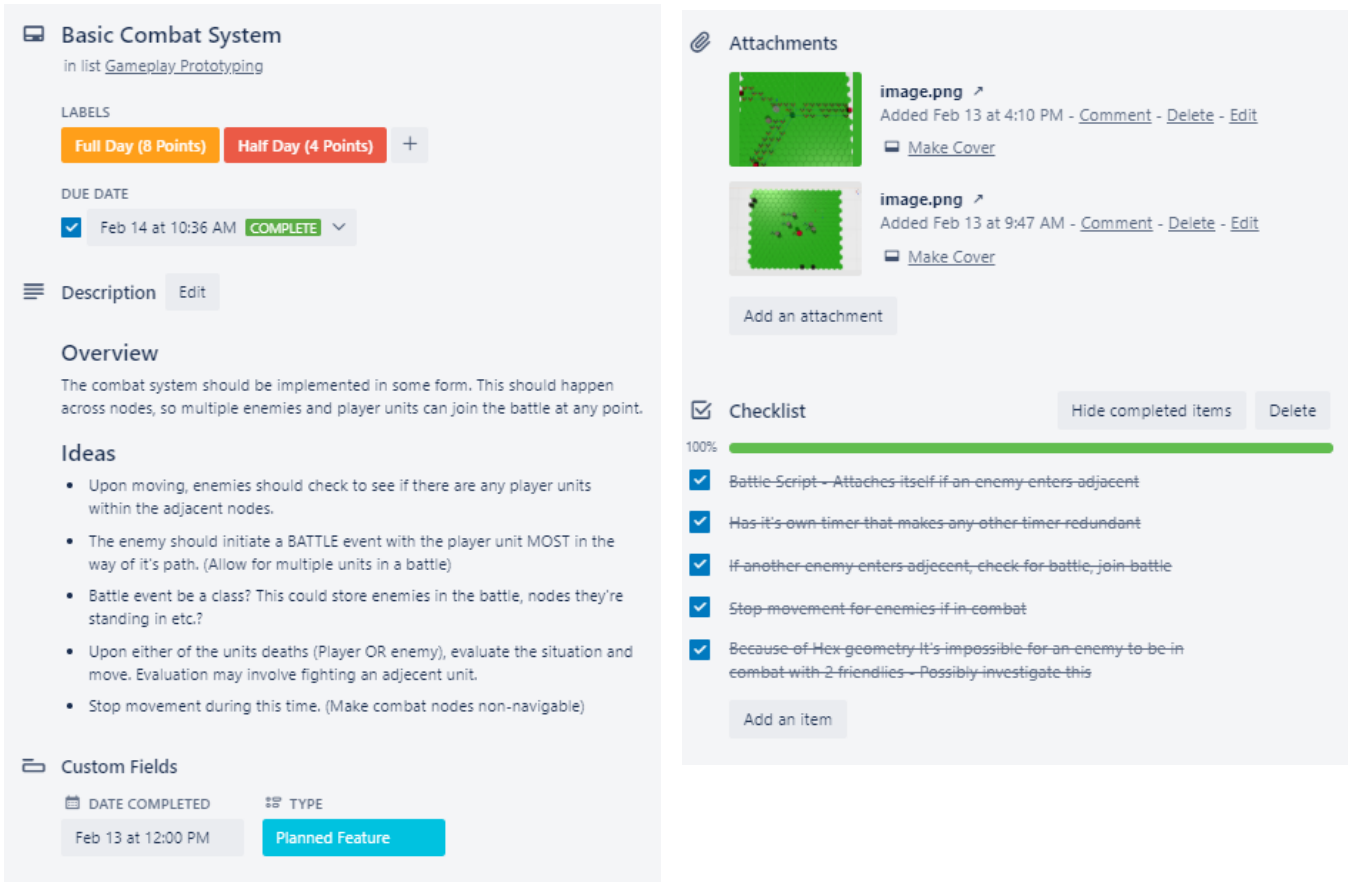


Figure 3: An example task

Development of a sprint took part on a “staging” branch on the repository, and then tested at the end of the week and merged into the master branch. This ensured there was always a working version of the project available that was at least up to date with the previous sprint’s development. When it came to name a commit, the following standard was implemented:

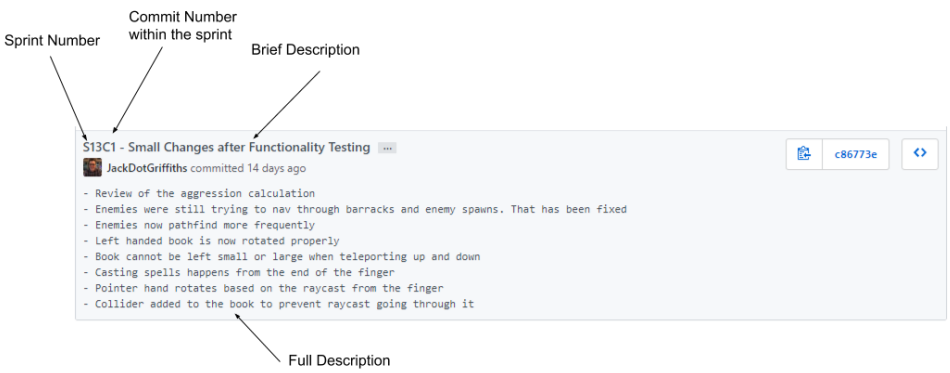


Figure 4 : An example commit

3.4 - Testing methods

A variety of testing was undertaken to ensure the success of this project. Functionality testing took place on every single feature implemented during each week. Mechanics were tested intrinsically and extraneously, to ensure they worked as intended.

For more integral systems, such as pathfinding, custom frameworks were developed to test them. During the testing, the number of enemies would be increased beyond realistic to gameplay, and game time was increased to speed up the simulation. Any bugs or issues with these systems would cause the simulation to stop, and hence easily debugged.

Custom editors were added to all key features in the project, which was useful when a VR headset wasn't available. Usability Testing was also a large part of this project's success, with all feedback and responses being analysed to help inform the development (See Appendix 10).

4 - Issues and Considerations

4.1 - Legal

High Ground VR is being developed under a 'Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License'[2]. This license allows for free sharing of the project, however, requires appropriate credit and doesn't allow for the material to be used for commercial purposes. All the assets produced for this project are created using appropriately licensed software, all of which allow for commercial distribution of the assets created using them. When any external assets were used, appropriate licensing was found. These assets included a skybox (Avionx, 2019), an audio effects pack (Imphenzia, 2017) and a game music pack (French, 2020).

4.2 - Social

An issue present during the development of High Ground VR was that there wasn't a wide network of people available to test the project. This was especially present during the coronavirus pandemic, in which it became much more difficult to find play testers with VR headsets. In order to run usability testing, both a VR and non-VR version of testing was created, so that some helpful feedback was still produced for analysis.

In order to meet the classification of a tower-defence game, certain features needed to be present. These were laid out at the start of the project, and their development time assessed to see which could be integrated into the game. Due to the potential release of the project, it was important that the tower defence genre was met in enough features to not attract negative response from customers.

[2] https://creativecommons.org/licenses/by-nc-nd/4.0/ : Read more about the license on the creative common's website.

4.3 - Ethical

In order to comply with the ethics policy laid out by the university, all play testers of the project were university students who had agreed to a consent form. This form laid out what would take place during the session, and how their responses will be used to aid development. It was important that protection from harm measures were in place during testing. While players are wearing the VR headset, they are unable to see. To combat this health and safety risk, the area was clear of objects that players may trip over, and steam's chaperone feature was turned on to show the player the edge of the play area.

4.4 - PEGI Rating

This project follows the PEGI rating system. This system provides age classifications for video games in 38 European countries, hence is the most applicable system to use for projects being produced for the UK market [3]. In order to create an accessible project and portfolio piece, the target rating for this game is PEGI 7. This requires that only a very mild form of violence is implied, hence the battle scenes in this project allow for this classification.

5 - Project Management Tools

5.1 - Project Management

Trello was used for the implementation of an agile Kanban workflow. The vertical columns in Trello, called 'lists', were used as the 'to do, doing, done' columns, and 'cards' were used as the tasks which moved from column to column. Figure 5 shows an example screenshot of Trello during development, with more screenshots available in Appendix 3.

[3] https://pegi.info/ : PEGI (Pan European Game Information) rating system.

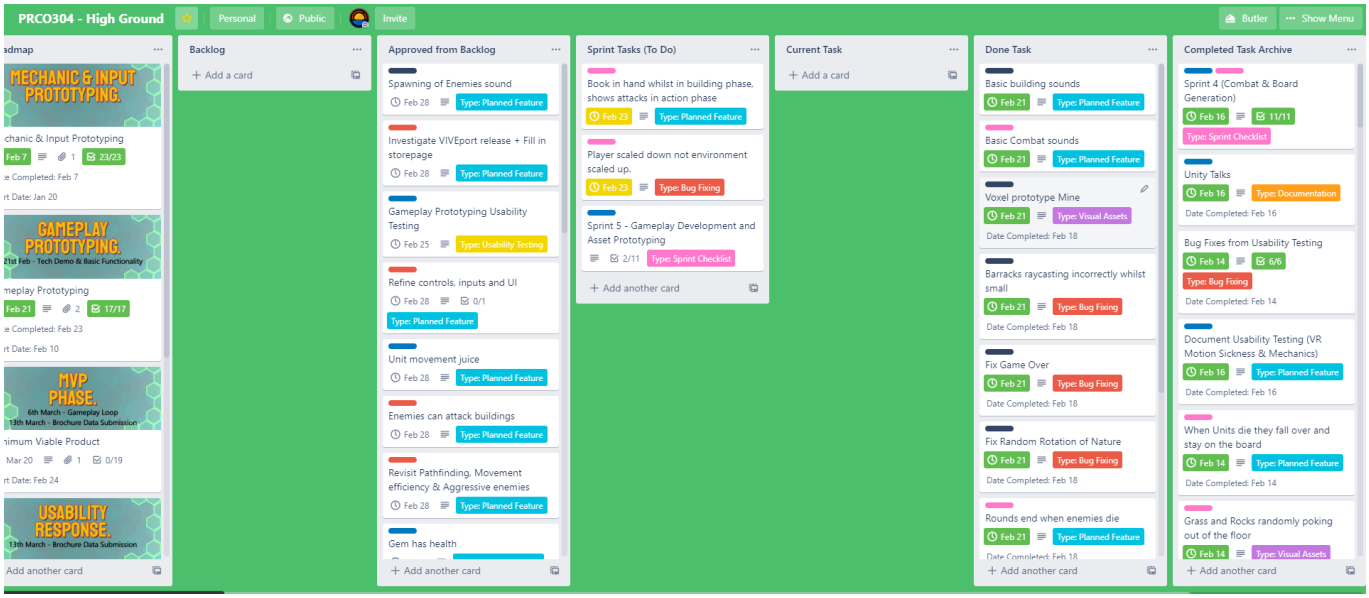


Figure 5: A screenshot of the Trello board during development of Sprint 5

5.2 - Version Control

The version control system used for this project was Git, through GitHub. The desktop client was used as it's a very accessible interface, yet also allows for use of Git in the command line for edge cases. The branch management allowed for a master branch, on which the most stable version of the software was held, and then a staging branch which was used for the development of each sprint. These were then merged, when tested, at the end of the week.

5.3 - Weekly Development Reports

Part of completing a sprint was to fill out Weekly Development Reports (See Appendix 9). These reports were used to track the success of a sprint. When a task was completed, the time taken was entered and compared against its assigned point value; this discrepancy was then used to help guide the point values assigned to tasks in the future. As seen in figure 6, the reports were also used to amalgamate other information on the sprint, including its functionality testing success rate and extra notes.

3	Mechanic & Input Prototyping		3rd Feb 2020 - 7th Feb 2020	Supervisor Meeting Date : 04/02/2020 11:30			
	Sprint 3 - Core Gameplay Mechanics		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	100.00%	Node System	Planned Feature	12	13	-1
	Devlog Link	https://ja	Units are Classes	Planned Feature	2	0.5	1.5
	Coding Standards Complete	☑	VR User Interface Research + Mood board	Documentation	8	2	6
	Trello Screenshot Upload	☑	Left hand book UI	Planned Feature	4	3.5	0.5
	Next Sprint Tasks	☑	Melee Units can be spawned	Planned Feature	4	4	0
	Roadmap Updated	☑	Place buildings	Planned Feature	4	2	2
	Branches Merged	☑	Mines generate money	Planned Feature	4	1	3
			VR Motion Sickness reasearch	Documentation	2	1	1
	Task time more than assumed	2	Centralize Terrain Generation	Bug Fixing	0.5	0.5	0
	Task time correct	2	Game Design Document Outlined	Documentation	4	3	1
	Tasks time less than assumed	7					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3	4	-1

Notes: Alot of time spent on the Node system and pathfinding this week, so much so that I spent the weekend before the sprint planning and researching the process. Scriptable objects were going to be used for a unit, however using classes seemed much more appropriate. A lot of framework built behind placing buildings, leaving room for verifying a buildings location before placing it. GDD planned out during this phase, as well as a lot of art styles and other documentation. Whilst testing, noticed lagging issues when board is rotating with all the spaces full. This could potentially become a bigger issue when geometry is not this simplistic, so the physics board rotation will be replaced with transform rotation which is much less intensive.

Figure 6 : An example weekly development report from Sprint 3

A development blog post was also created for every week of development. This acted as more of a front-facing record of development. It can also be used as a helpful archive of the development of the project (See Appendix 8).

6 - Implementation

6.1 - Mechanic and Input Prototyping Phase

6.1.1 - Sprint 1: Input and Project Preparation

Overview

This sprint started with the setup of the Unity project, importing all required packages and setting up the SteamVR Input system. It was a primary requirement that basic virtual reality controls were implemented during this phase, so that other features could be developed upon them. The hexagonal game board also needed to be generated this sprint, using an axial coordinate system.

Sprint Objectives

1. Set up the development environment and documentation templates
2. Create a game board built up of hexagons
3. Implement virtual reality interactions through the SteamVR package
    a. Prototyped movement systems
        i. Pointing and teleporting around the game board using the centre trackpad button
        ii. Press the grab button and swipe the controller to move the player
    b. Point and click at a hexagon on a game board to return info on that hexagon

Implementation Methods

The board generation uses an axial coordinate system. It is an alternative to the cube coordinate system for hexagon grid mapping; however, it's only built by using two of the three coordinates. (*Red Blob Games: Hexagonal Grids*, 2013).

The implementation, shown in figure 7, keeps track of the offset on both the X and Z axis, and multiplies that within a for-loop until a desired length or width of hexagons is reached. This system allows for any size of game board to be generated. Once the offset is calculated, a hexagon prefab is instantiated in the correct position. A length and width of 17 hexagons is used for the game, which was determined as a perfect size after usability testing sessions (See Appendix 10).

```
/// <summary>
/// Calculates the position of each hexagon required from the m_hexGapSize and the current index in the graph.
/// </summary>
public void generateRec()
{
    bool _offsetColumn = false;

    //Generate game board based upon a desired lenth and width.
    for (int i = 0; i < m_length; i++)
    {
        m_currentZ = 0;
        for (int j = 0; j < m_width; j++)
        {
            m_currentX = i * m_hexagonalWidth; //The current X offset from the origin point.
            m_currentZ = j * m_hexagonalHeight; //The current Z offset from the origin point.
            if (_offsetColumn == true)
            {
                m_currentX += m_hexagonalWidth / 2; //Column number is odd, offset on the X by half the width.
            }
            _offsetColumn = !_offsetColumn; //Alternate Offset column so the hexes fit together.


            placeHex(i.ToString(), j.ToString()); //Place a hexagon in the desired position.
        }
        _offsetColumn = false;
    }
}
```

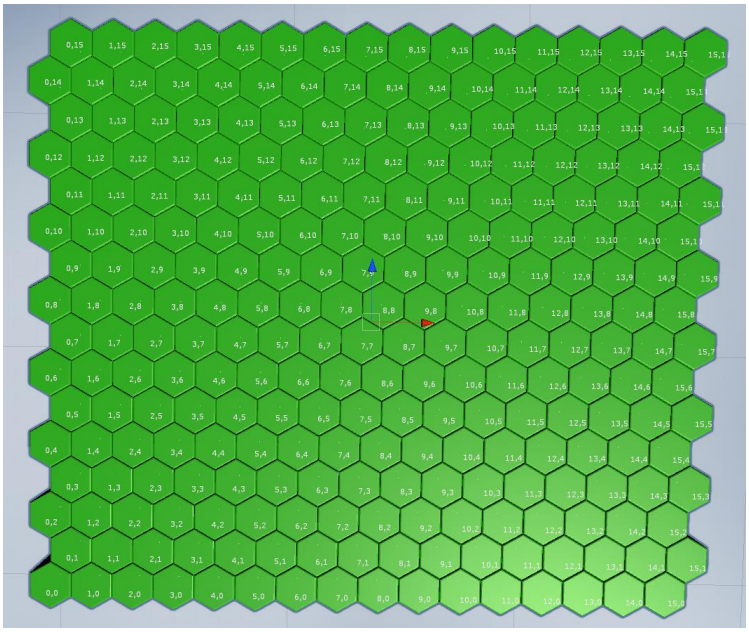


Figure 7: Code-snippet and outcome of game board generation

The SteamVR package contained a prefab that could be imported to enable tracking of the headset and controllers. The package also had a default controller layout and input system built in, which was modified to suit the needs of this project. For the player to be able to point and click at a chosen hexagon, a Raycast is emitted forward from the tip of the controller. This then hits an object and returns the data of that object to the input manager. The returned object is checked as to whether it's a hexagon, and appropriately highlighted to indicate the player is pointing at it. If the centre trackpad button is pressed whilst the player is pointing at a hexagon, the entire camera rig is scaled down and moved to the position of that node. Whilst small, the

player is still able to point at other nodes to move the camera rig around, and point at the sky and press the button to become large again.

```
//Raycast from the pointer position, out forwards and at a downwards angle. This is a different angle when the player is large.
if (Physics.Raycast(m_pointerPosition.transform.position, MainController.transform.forward - (MainController.transform.up * 0.5f), out _hit, 1000
{
    //If it hits an environment Hex, highlight.
    if (_hit.collider.gameObject.tag == "Environment")
    {
        removeHighlight(); //Remove the highlight from all objects.
        m_enlargePlayer = false; //Player is not about to teleport and scale up.

        m_currentlySelectedHex = _hit.collider.gameObject; //Keep track of the currently selected hex.

        MeshRenderer _hitMesh = _hit.collider.gameObject.GetComponent<MeshRenderer>(); //Get the mesh of the _hit object.
        updateObjectList(_hitMesh);  //Update the list of hexagons to apply the default material to. Exludes the hex the player is pointing at.


        //Show the laser pointer
        m_mainPointer.startWidth = 0.05f;
        m_mainPointer.endWidth = 0.00f;

        //Turn laser colour to blue when you correctly collided with environment.
        m_mainPointer.startColor = Color.blue;
        m_mainPointer.endColor = Color.blue;

        //Apply outline material to the selected object
        Material[] _matArray = _hitMesh.materials;
        List<Material> _matList = new List<Material>();
        _matList = new List<Material>();
        _matList.Add(m_outlineMaterial);
        _hitMesh.materials = _matList.ToArray();
```

Figure 8: Code-snippet of the point and click functionality.

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 1

6.1.2 - Sprint 2: Input Refinement and Basic Gameplay Elements

Overview

Sprint 2 contains a review of all the inputs and interactions in the project. It is also the start of visual research to inform asset prototyping. It was a requirement that these assets be made, as it helps with visualizing the outcome whilst developing gameplay mechanics. Most of the assets will remain as cubes for the meantime, with only a few needing more advanced 3D models.

Sprint Objectives

1. Produce some temporary 3D assets
    a. Hexagonal Prism
    b. Barracks
    c. Mine
    d. Cloud Shader -
2. Research and documentation aspects of user experience
    a. User Story
    b. Mood Board

Implementation Methods

All the 3D models were created within the unity editor. The Barracks model used the existing hexagon model, along with cubes to create crenellations on top. The Mine model is a

representation of a staircase going downwards into the ground. All other required models are going to remain as coloured cubes until later in the project.

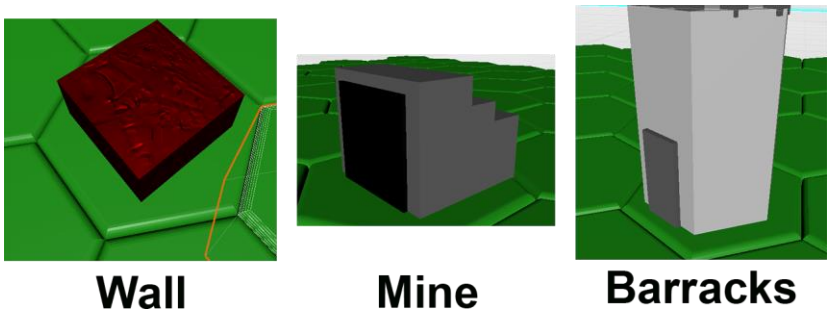


Figure 9: Images of the prototype buildings

The Universal Render Pipeline was imported into the unity project, which added the shader graph tool. This allows for visual shader scripting whilst within the unity editor. The graph created for the cloud environment uses a variety of gradient noise to create a scrolling Y axis displacement on a flat plane. This emulates a wavy effect, and alongside displacement-based colouring, creates an effective stylised-cloud platform for the player to stand on.

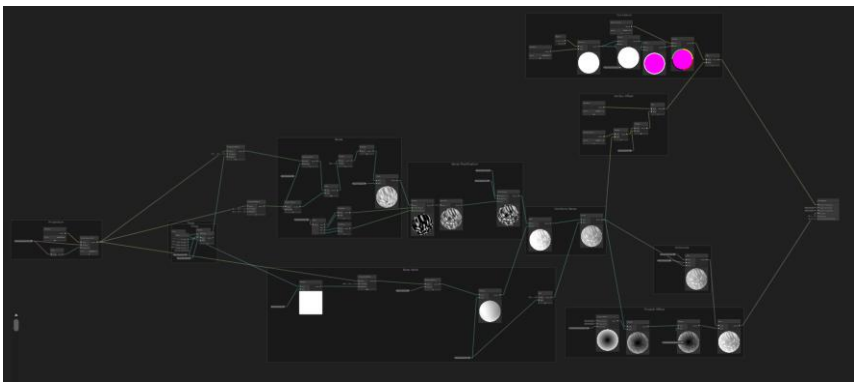


Figure 10: The shader graph

The user stories and mood boards will be used continuously whilst creating assets and planning the user experience of new mechanics. They are kept in the Game Design Document (See Appendix 4).

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 2

6.1.3 - Sprint 3: Core Gameplay Mechanics

Overview

The aim of the third sprint was to implement a hexagonal based node system, on which the entire basis of gameplay would rely. It was a requirement that pathfinding was functional by

the end of this sprint, as well as a simplistic user interface that the player could use to interact with the game. This would then be used to select one of the 3 buildings in the game, and then the player would be able to place them onto the board.

Sprint Objectives

1. Hexagonal node system
    a. Which nodes are adjacent to it?
    b. What it contains (e.g. wall, mine, barracks, enemy unit etc.)
    c. It's physical location in the game
2. Pathfinding needs to be able to create a path
3. A book user interface
    a. Point and click to select the building to place
4. Place a building on the game board using point and click
5. A basic money system
    a. Buildings cost an amount of money to place
    b. Mines, once placed, need to generate money over time.

Implementation Methods

The entire basis of the game's functionality resides around this node system; therefore, it was implemented to a near-full extent during this sprint. The class that was created holds properties relating to a node itself and the hexagon Game Object associated with it. As seen in figure 11, the class also has a constructor, which is used when the game board is generated at the start of each game.

```
public class Node
{
    public string label = ""; //Label of the node. Essentially the name.
    public nodeTypes navigability; //The navigability state of the node;
    public GameObject hex; //GameObject hexagon associated with this node.
    public List<Node> adjecant = new List<Node>(); //Adjecent nodes to this node.
    public int x; //x position of the node in the graph.
    public int y; //y position of the node in the graph

    /// <summary>
    /// Node Constructor
    /// </summary>
    /// <param name="label">Label, or name of the node</param>
    /// <param name="x">x position of the node in the graph.</param>
    /// <param name="y">y position of the node in the graph</param>
    /// <param name="hex">GameObject hexagon associated with this node.</param>
    /// <param name="navigability">The navigability of the node</param>
    1 reference
    public Node(string label, int x, int y, GameObject hex, nodeTypes navigability)
    {
        this.label = label;
        this.x = x;
        this.y = y;
        this.hex = hex;
        this.navigability = navigability;
    }
```

Figure 11: Node Class and it's constructor

In order to implement pathfinding, it was essential that each node had a populated list of nodes adjacent to it. By using the implemented axial coordinate system, each node was populated with a list of its adjacent nodes. The entire game board is then stored within a public 2D-Array on the GameBoardGeneration singleton.

The pathfinding implementation calculates which adjacent node reduces the distance between the current position and a goal position the most, and then chooses that node to navigate to. This algorithm returns a list of nodes, which will be used in the future for enemy navigation. This resulted in enemy behaviour that would occasionally move straight towards the goal position, disregarding any cost it could have saved by directing around an obstacle rather than towards it. Figure 12 shows an example of the pathfinding algorithm functioning at this stage.

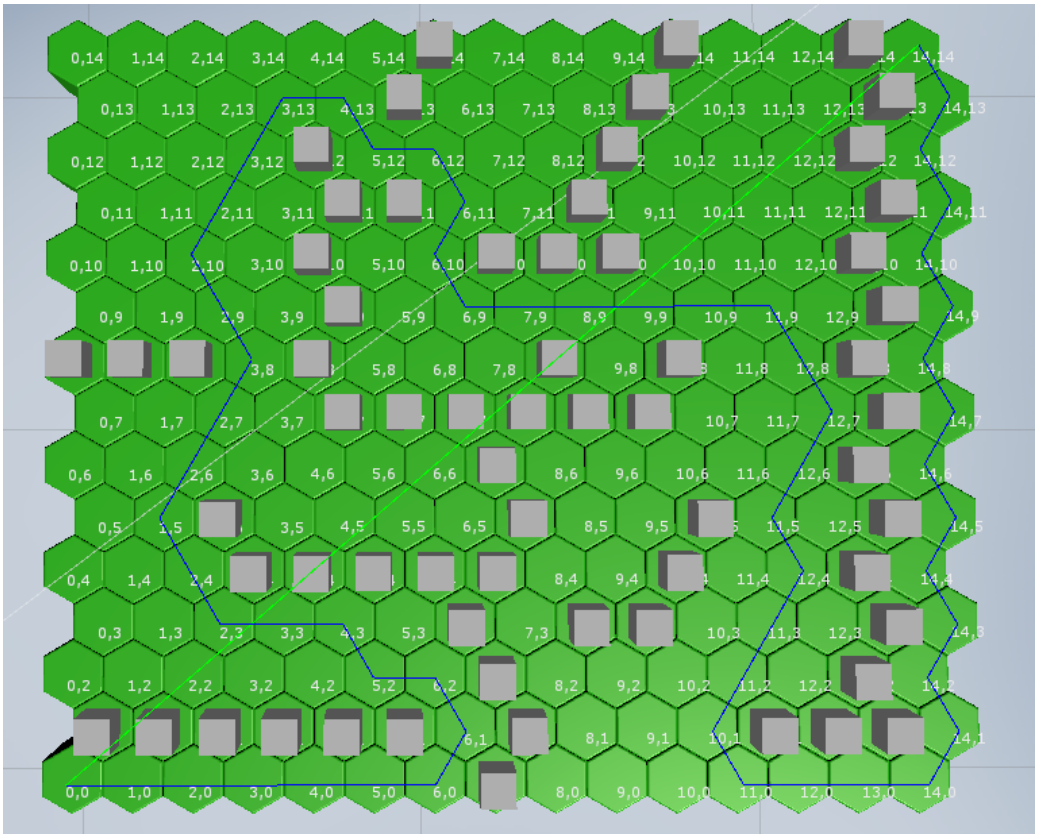


Figure 12: Example of Pathfinding

The user interface and building placement both utilized the point and click functionality added in the first sprint. The prototype UI takes the form of a book which is placed in the player's left hand, allowing them to point and click at various images on the book to choose which building they wish to place. Once a building is selected, the player can then point and click at a node, which then instantiates the selected building at the target location. The node type, or navigability, of that node is updated so that the graph is up to date with placed buildings.

In order to implement a simple money system, a game manager was created. This used the singleton design pattern, so there could only be one active instance of this script at any one time. This could then be referenced by other scripts to return variables or functions that were defined in that manager. Two integers, ‘startingGold’ and ‘currentGold’ were added, as well as functions for spending money and earning money.

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 3

6.2 - Gameplay Prototyping Phase

6.2.1 - Sprint 4: Combat and Board Generation

Overview

The aim of this week was to take the node system added in sprint 3 and use it to start developing some key elements of gameplay. It was a requirement that enemies spawn, were able to move around using a calculated path, and then that they were also able to take part in combat with friendly units that are automatically spawned from a player’s barracks.

Sprint Objectives

1. Enemy navigation, using the pathfinding to move around the board.
2. Enemies need to spawn
    a. On adjacent nodes to randomly placed enemy spawns
    b. Done on a timer.
3. Combat system needs to be implemented
    a. Player units spawn on the node in front of the barracks
    b. Enemies adjacent to players start combat
    c. Each unit has its own health and damage
4. Gras s and rocks placed randomly on the game board.

Implementation Methods

Enemy navigation uses the list of nodes from pathfinding to create a list of positions they need to move onto. This functions on a timer, so as the timer ticks after 2 seconds, the enemies target position becomes the next node on the list. It then runs a coroutine that moves the enemy towards the hexagon's position in the world. In order to prevent the use of animations or physics objects, the ‘hop’ movement is done using the code shown in figure 13. This function is called within the Update loop; hence it is called every frame. It causes the enemy to move towards the next node, using the sin of the percentage of its journey from the previous node as an offset on the Y axis, creating an arching movement.

```
/// <summary>
/// Moves the GameObject towards m_targetPosition. This needs to be called in update to function.
/// </summary>
1 reference
void MoveEnemy()
{
    float _yOffset = 0;
    if(currentStepIndex != 0)
    {
        //Move the enemy along it's path. Calculate distance from the goal as a percentage.
        float _maxDistance = Vector3.Distance(m_groupPath[currentStepIndex - 1].hex.transform.position, m_targetPosition);
        float _currentDistance = Vector3.Distance(transform.position, m_targetPosition);
        float _percentage = _currentDistance / _maxDistance;
        //Sin of the percentage creates a 'hop' like movement.
        _yOffset = 2 * Mathf.Pow(Mathf.Sin(_percentage),2);
    }
    Vector3 _hopPosition = new Vector3(m_targetPosition.x, m_targetPosition.y + _yOffset, m_targetPosition.z); //Position the enemy needs to move to.
    transform.position = Vector3.Lerp(transform.position, _hopPosition, m_movementSpeed * GameManager.Instance.GameSpeed * timePerception); //Lerp the enemy towards it's target position.

    try
    {
        RotateEachUnit(); //Rotates each unit towards the next position, so they're facing where they're headed.
    }
    catch { }
}
```

Figure 13: Code-snippet of enemy's movement

Utilizing the graph stored within the 'GameBoardGeneration' singleton, a list of nodes on the outer edge of the graph was generated in the game manager. This was used to randomly place any number of enemy spawns around the rim of the game board, by instantiating a temporary prefab in a node's location and updating that node's navigability. Enemy spawns were given a 'EnemySpawnBehaviour' script, which holds a function that spawns an enemy unit in any adjacent node to the enemy spawn. Game manager would then run a coroutine, which would call that function on any of the enemy spawns, and then wait for a timer to pass before trying again. This was done until a maximum enemy count was reached.

A unit class was also added at this stage, which holds information on a unit such as health, damage and whether it's a friendly or enemy. The stats on these units were randomly generated.

The implementation of the combat system started with adding a 'BarracksBehaviour' script, which was added onto the barracks prefab. Once a barracks is placed, it fires a raycast from the top of the doorway downwards, in order to locate the node on which it needs to place friendly units. It then instantiates friendly units on that node, and generates their health and damage using the constructor. For the purposes of combat, the barracks keeps a track of how many units it currently has, and how many it should have, and runs a respawning coroutine which gradually respawns units, the code of which can be seen in figure 14.

```
/// <summary>
/// Controls the respawn of units from a barracks.
/// </summary>
/// <returns></returns>
0 references
IEnumerator RespawnUnits()
{
    //Wait for a set timer.
    yield return new WaitForSeconds(m_unitRespawnDelay);
    //If the target node is completely clear , Spawn a unit.
    if (BarracksUnitNode.hex.transform.childCount != 0)
    {
        if (BarracksUnitNode.hex.transform.GetChild(0).GetComponent<EnemyBehaviour>() == null) //Ensure an enemy isn't in the way
        {
            BarracksUnitNode.navigability = nodeTypes.playerUnit;
            SpawnAUnit();
        }
    }
    //If there is a player unit currently there, add another.
    else if (BarracksUnitNode.navigability == nodeTypes.navigable || BarracksUnitNode.navigability == nodeTypes.playerUnit)
    {
        BarracksUnitNode.navigability = nodeTypes.playerUnit;
        SpawnAUnit();
    }
    m_respawning = false;
    yield return null;
}
```

Figure 14: Code-snippet of the units respawning

Within the Update loop of the barracks, a check is constantly running to see if there are any enemies in adjacent nodes. If so, a ‘BattleBehaviour’ script is instantiated on the barracks, and the movement of enemies is paused. As seen in figure 15, both the friendly and enemy units are added to lists on the battle ‘event’, and then a timer is started which causes each unit to attack once it ticks over. In the background, the barracks update loop is still looking for enemies moving into adjacent tiles to the battle, and it will add them to the event if they’re found. The battle ends once either all enemy units die, or all the friendly units die. This is also when the BattleBehaviour script is removed from the barracks, and enemy’s movement resumes.

```
/// <summary>
/// Starts the battle.
/// </summary>
/// <param name="_friendlyUnits">List of friendly units to have in the battle.</param>
/// <param name="_enemyUnits">List of enemy units to have in the battle.</param>
1 reference
public void StartBattle(List<Unit> _friendlyUnits, List<Unit> _enemyUnits)
{
    enemyGroups = new List<EnemyBehaviour>(); //A list of enemy units within the battle
    friendlyGroups = new List<BarracksBehaviour>(); //A list of friendly units within the battle.
    friendlyUnits = _friendlyUnits;
    enemyUnits = _enemyUnits;

    //Seperate timers for enemies and friendly units, which is used later for spells effecting the time perception of an enemy.
    m_currentFriendlyTimer = m_battleTimer;
    m_currentEnemyTimer = m_battleTimer;
    m_battleOccuring = true;
}
```

Figure 15: Code-snippet of BattleBehaviour

The addition of board detailing was simply a case of instantiating prefabs of grass and stone in random positions when the board was generated. This is done on a percentage chance, so that a density value was available as a slider within the Unity Editor.

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 4

6.2.2 - Sprint 5: Gameplay Development and Asset Prototyping

Overview

Sprint 5 was the last sprint of prototyping, before moving into the MVP phase. It was a requirement that the player was able to lose the game, and that an attack phase started once a timer was up, and the building phase started once all enemies were killed. Building placement rules also needed to be implemented, so that they matched the rules planned out in the Game Design Document. (See Appendix 4)

Sprint Objectives

1. Balance health and damage of enemies and players units.
2. Building placement rules implemented
3. Building and Attack Phases announced
    a. Building phase ends after a set amount of time
    b. Attack Phase ends after all the enemies are killed

Implementation Methods

Through playtesting, suitable bounds were found for the health of the units. Friendly units have a smaller but higher value range to choose from, and enemies have a higher range to choose from, but with a lower minimum and higher maximum. This creates an evenly matched enemy that still has an opportunity to defeat the player but doesn't in all circumstances.

A script called 'ValidateBuildingLocation' was added, which is called upon when a building is placed. This enforces a variety of rules, and returns a bool based on whether the target location is valid or not. Figure 16 shows an example of the verification rules needed to pass to be able to place a mine.

```
/// <summary>
/// Verify the position of a Mine
/// </summary>
/// <param name="_targetNode">Node on which to place a Mine</param>
/// <returns>True or false based on whether the targetNode can accept a Mine.</returns>
2 references
public bool verifyMine(Node _targetNode)
{
    bool _validLocation = false;
    if (nodeEmpty(_targetNode) && !adjacentToEnemySpawn(_targetNode) && !adjacentToGem(_targetNode) && checkGemAccessible(_targetNode))
    {
        _validLocation = true;
    }
    return _validLocation;
}
```

Figure 16: Example of the mine placement validation

The Game Manager was enlisted to manage phase changes. A timer is used to give the player a length of time as a building phase, and after that runs out enemies start spawning and the attack phase begins. The game manager also has a count of the number of enemies spawned during a round, and when an enemy dies it is taken off that count. Once the count reaches 0, the attack phase is over, and the building phase begins again. In order to calculate how many enemies should spawn per round, the equation shown in figure 17 is used.

$$\textbf{Enemies to spawn} = (\textbf{Rounded to Integer})\ 3 * \sqrt{\textbf{Round Counter}}$$

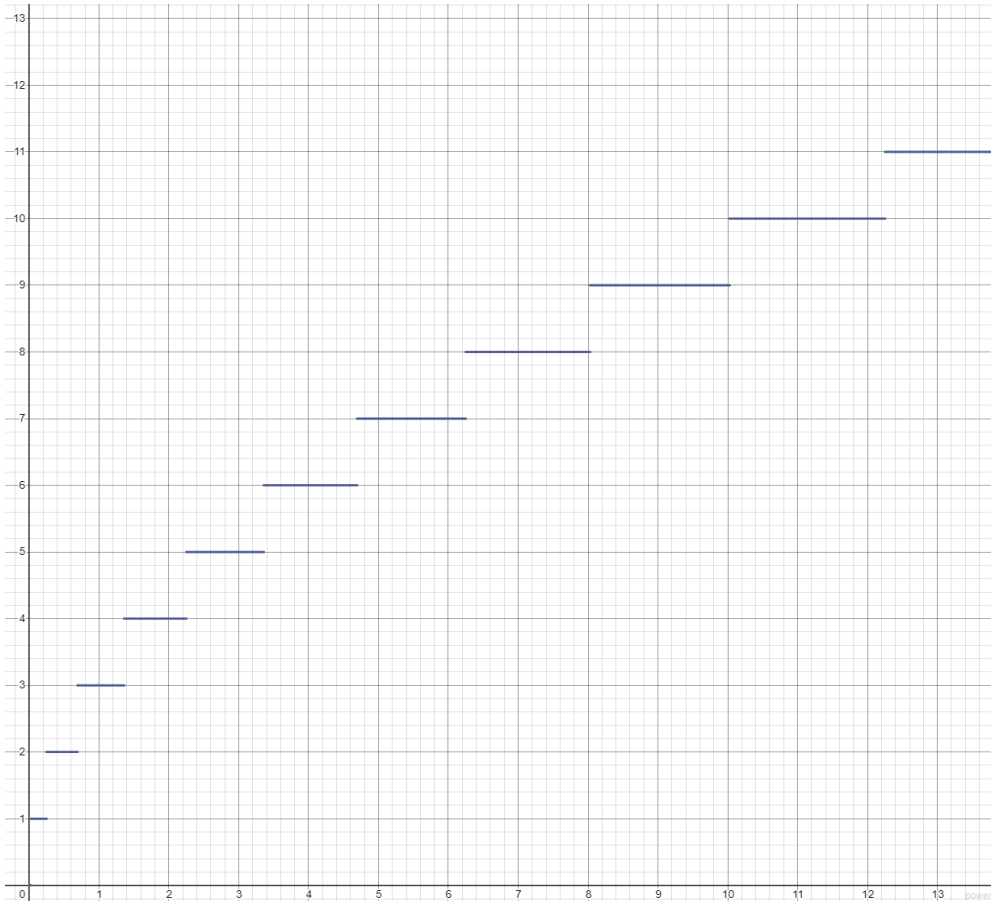


Figure 17: Enemy Spawning Calculation & Graph

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 5

6.3 - Minimum Viable Product Phase

6.3.1 - Sprint 6: Pathfinding Refinement and Movement Development

Overview

Usability testing after the Gameplay Prototyping Phase didn't reveal many fixes or changes to work on, so this sprint could be devoted to adding new features which were planned for this phase. Firstly, an Audio Manager needed to be added, so audio effects could be played

from various scripts around the project. The second addition was related to the planned ‘aggression’ value that enemies use to inform their pathfinding and behaviour around the board. It was a requirement that aggressive enemies were then able to ‘siege’ a building, if it was in their way.

Sprint Objectives

1. An audio manager
2. Aggressive enemies
    a. Enemy groups must have an ‘aggression’ value, which increases if the spot they’re trying to move onto is occupied.
3. Enemies can siege through buildings as a result of their aggression.
    a. Buildings will need health
    b. If beyond a certain threshold, enemies will engage in sieges with buildings if they’re in the way.

Implementation Methods

For the audio manager, a ‘Sound’ class was created, which held information on an audio clip and how it would be played. A function within the audio manager singleton was written which played a sound, with a large variety of parameters, seen in figure 18, which could customize how the sound would play.

```
/// <summary>
/// Plays either a 2D or 3D sound on the passed in targetObject
/// </summary>
/// <param name="_name">Name of the clip to play.</param>
/// <param name="_audioType">The type of audio list to choose from.</param>
/// <param name="_mixer">Mixer on which to play it on</param>
/// <param name="_seperateObject">Whether to spawn a seperate object with the source on. Useful for things about to be destroyed.</param>
/// <param name="_destroyAfterPlaying">Whether to destroy the audioSource after the clip has played.</param>
/// <param name="_playin2D">Whether to acknowledge the spatialBlend and min/max distance of the clip. If true, spatial blend will be set to 0.</param>
/// <param name="_targetObject">Target GameObject to attatch the audio source to.</param>
/// <param name="_pitchShiftAmount">An amount of pitch shift to apply to the audio source. If 0, no pitch shift will occur.</param>
36 references
```

Figure 18: Parameters for playing a sound

To implement enemy aggression, the Enemy behaviour script was given a public variable that it could use, called groupAggression. When it came to an enemy starting to move, it checked the node ahead. If it wasn’t clear, the movement was paused, and the enemy’s aggression value increased. Pathfinding was then run again, to see if a new path was available with the new aggression value considered. Based on the round number, an equation is used to determine whether an enemy's aggression is enough for it to navigate through buildings. That equation can be seen in figure 19, and if an enemy’s aggression was above this calculated value then a siege would occur.

$$aggression\ chance\ =\ 1.0\ -\ (0.4 * (RoundCounter/2))$$

Figure 19: Aggression Calculation

If this boundary is passed, the enemy would attach an instance of the ‘SiegeBehaviour’ script, which is a modified version of the ‘BattleBehaviour’. This meant buildings were given a health component, which allows them to take damage, get destroyed, and regenerate health after a period of inaction. The timer on the siege script calls the enemies to do damage to the building and do so until the building is destroyed and hence the node becomes navigable.

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9: Week 6

6.3.2 - Sprint 7: Menu Functionality and Final Mechanics

Overview

In sprint 7, it was a requirement that instead of the enemies instantly destroying the gem, they entered a siege with it. This gives the player time to interject with their spells, the functionality of which also needed to be added at this stage. It was relatively unexplored up until this week, however a main menu interface needed to be added, so that the player could restart the game once it ended. By the end of this sprint, all the final mechanics will be in the game in a prototyped form.

Sprint Objectives

1. Enemies must siege a gem to win
2. A Main Menu
    a. A physical button,
    b. Collider on the player’s hand to press it
3. Spells
    a. Spell should take the form of a lightning bolt coming from the hand and hitting the enemies
    b. Work on a group if the player is large
    c. On individual units if the player is small.

Implementation Methods

In order to complete the first objective, the enemy’s behaviour was slightly altered to consider whether the gem was the next step in their path. If so, they initiated a siege, much like if they were destroying a building. When the gem runs out of health, a custom function turns the game speed to zero, halting all movement of units. This is when the main menu appears, allowing the player to press a button and restart the game.

The physical buttons on the main menu use a box collider and rigidbody. Once pushed by the hand, it tracks its distance away from its original position, and if it has passed a certain threshold, it activates and won’t go any further. Figure 20 shows the code used to control the button.

```
void Update()
{
    m_pressed = false;
    if(isLocked == false)
    {
        // Return button to startPosition
        if (transform.localPosition != m_startPosition)
        {
            transform.localPosition = Vector3.Lerp(transform.localPosition, m_startPosition, Time.deltaTime * m_buttonReturnSpeed);
        }

        if (transform.localPosition.z > m_startPosition.z + m_buttonPressDistance - (m_buttonPressDistance / 4))
        {
            //Button is lower than it should be
            transform.localPosition = new Vector3(m_startPosition.x, m_startPosition.y, m_startPosition.z + m_buttonPressDistance);
            if (!m_pressed && m_released)
            {
                m_released = false;
                m_pressed = true;
                m_buttonPressMethod.Invoke(); //Invoke the method assigned to the button.
            }
        }
        else if (transform.localPosition.z < m_startPosition.z)
        {
            //Button is higher than it should be
            transform.localPosition = new Vector3(m_startPosition.x, m_startPosition.y, m_startPosition.z);
            m_released = true;
        }

        else if (Mathf.Abs(transform.localPosition.z - m_startPosition.z) < m_buttonPressDistance / 5.0f)
        {
            m_released = true;
        }
        //Dectivate unpressed button
        if (transform.localPosition.x != m_startPosition.x || transform.localPosition.y != m_startPosition.y)
        {
            transform.localPosition = new Vector3(m_startPosition.x, m_startPosition.y, transform.localPosition.z);
        }
    }
}
```

Figure 20: Physical Button Code

The spell system uses the point and click functionality and detects whether the hit location of the raycast hits either a group of enemies or a singular enemy. It then deals damage as per the defined values in the ‘InteractionManager’ script. For this sprint, only the damage spell is implemented. The interesting functionality comes from the procedural lightning effect, shown and explained in figure 21.

```
3 references
IEnumerator regularAttackEffect(float _playerScale, GameObject _controller, Vector3 _hitPos)
{
    m_currentlyAttacking = true;
    //AudioManager.Instance.Play2DSound(SoundLists.zapSounds, false, 0, true, false, true);
    AudioManager.Instance.PlaySound("regularLightning", AudioLists.Combat, AudioMixers.Effects, true, true, true, this.gameObject, 0.1f);

    //Create a seperate object to hold the zap effect.
    GameObject _goLine = new GameObject("regularAttackEffect");
    StartCoroutine(destroyEffect(_goLine, m_lightningTiming * m_lightningOccurance));
    _goLine.transform.parent = _controller.transform;

    List<LineRenderer> _lightningLines = new List<LineRenderer>();

    //Create the amount of line renderers required.
    for (int i = 0; i < m_lightningStrikes; i++)
    {
        GameObject _go = new GameObject("regularAttackEffect");
        _go.transform.parent = _goLine.transform;
        Destroy(_go, m_lightningTiming * m_lightningOccurance);
        _lightningLines.Add(new GameObject("regularAttackEffect").AddComponent<LineRenderer>());
        _lightningLines[i].material = m_lightningMaterial;
        _lightningLines[i].startWidth = 0.1f * _playerScale;
        _lightningLines[i].endWidth = 0.1f * _playerScale;
    }

    ///Break the line down into positions, equally sized.
    float _lineDistance = Vector3.Distance(m_pointPosition.position, _hitPos);
    float _breakDistance = _lineDistance / m_lightningStrikeBreakPoints;
    Vector3 _direction = _hitPos - m_pointPosition.position;
    _direction.Normalize();
    List<Vector3> _breakPositions = new List<Vector3>();
    m_breakPoints = new List<Vector3>();

    for (int i = 0; i < m_lightningStrikeBreakPoints; i++)
    {
        _breakPositions.Add(m_pointPosition.position + _direction * (i * _breakDistance));
        m_breakPoints.Add(_breakPositions[i]);
    }


    //Set the positions on the line to a random variation of those points. creates the lightning effect.
    for (int i = 0; i < m_lightningOccurance; i++)
    {
        foreach (LineRenderer _line in _lightningLines)
        {
            _line.positionCount = _breakPositions.Count;
            Vector3[] _linePositions = new Vector3[_breakPositions.Count];
            _linePositions[0] = m_pointPosition.position;
            for (int j = 1; j < _breakPositions.Count; j++)
            {
                _linePositions[j] = _breakPositions[j] + Random.insideUnitSphere * m_lightningWidth * _playerScale;
            }
            _linePositions[_breakPositions.Count - 1] = _hitPos;
            _line.SetPositions(_linePositions);
        }
        yield return new WaitForSeconds(m_lightningTiming);
    }
    //Clear all of the lightning

    foreach (LineRenderer _line in _lightningLines)
    {
        Destroy(_line.gameObject);
    }

    yield return new WaitForSeconds(m_attackCooldown);
    m_currentlyAttacking = false;
}
```

Figure 21: Code-snippet of the lightning effect

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 7

6.3.3 - Sprint 8: Asset Refinement and Smarter Enemies

Overview

The goal of sprint 8 was to add the other spells and improve the user interface using physical buttons. It is also a requirement that more audio effects are added, during placing buildings and battles, and that mines and barracks receive their final updated assets. The

largest addition to this sprint will be a tank enemy. These will navigate slightly differently to other enemies, have much larger health and damage and only spawn on certain rounds.

Sprint Objectives

1. Finish spells
    a. Slow down units spell added
    b. Speed up units spell added
    c. Consider effects of these spells in battle and sieges.
2. Improvements to User Interface
    a. Book uses physical buttons
3. Combat and Building Audio
4. Refined Mine and Barracks
5. Tank Enemy
    a. Unit that's 3 times the size of others
    b. Much higher health and damage range
    c. Starts spawning on round 4, spawns again after 3 rounds, adding one tank each time.
    d. Navigates towards a random building placed by the player, not always straight towards the gem.

Implementation Methods

Since the start of the implementation of the units, they have had a 'timePerception' property, which is used when they move. This has a default value of 1 and is used as a multiplier on their timer. In order to implement spells which had the ability to slow down and speed up units, it was simply a case of changing that property for a short amount of time, and then returning it to 1 again. Using the same implementation of the damage spell, a coroutine which changes their time perception property was written and added into the 2 new spells. As shown in figure 21, considerations for this value are also added to the 'SiegeBehaviour' script. This functionality is also present in the 'BattleBehaviour' script, so the player can interact with enemies through that also.

```
float _totalTimePerception = 0;
//Get the average timePerception of all enemy groups in the siege
for (int i = 0; i < enemyGroups.Count; i++)
{
    _totalTimePerception += enemyGroups[i].timePerception;
}
m_timePerception = _totalTimePerception / enemyGroups.Count;
//Use timePerception as a multiplier when running the timer.
m_currentTimer -= Time.deltaTime * GameManager.Instance.GameSpeed * m_timePerception;
```

Figure 22: timePerception considerations within the SiegeBehaviour script.

The improvement to the user interface was simply a case of reusing the physical buttons found on the main menu. This proved to be a much more natural input method, so the same

interaction was added for selecting spells. Audio enhancements were completed using the audio manager singleton.

Refined assets were produced using MagicaVoxel and taken into Blender for final touch ups before importing into Unity. The assets produced can be seen in Figure 23 and are the final renditions of the buildings in the project.

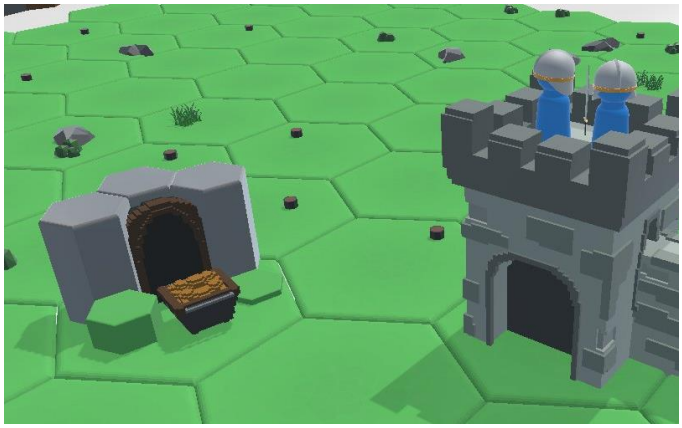


Figure 23: The final building models

Implementation of a larger ‘Tank’ enemy was straight forward. This was due to the implementation of a Unit class, along with customizable spawning of enemies, which were already modifiable enough to produce this enemy type. A larger health and damage range was used, and the spawning was modified so that only one enemy spawned which was three times the size of the regular enemies. An adapted pathfinding function was written for this enemy, that allowed it to locate any placed building and navigate straight towards it.

The enemy spawning function was modified so that it generates a list of all the enemies it should be spawning before it starts doing so. As can be seen in Figure 24, If a tank is required on that round, it will randomly place it within the enemy queue. The enemy spawns then have references to both the normal enemy group and the tank enemy, which it uses accordingly.

```
//Create a list of the enemies to spawn this round.
List<enemyTypes> _roundEnemies = new List<enemyTypes>(enemyAmount);

for (int i = 0; i < enemyAmount; i++)
{
    _roundEnemies.Add(enemyTypes.Regular);
}


//Work out the amount of Tank enemies to spawn from the current Round.
int _tanksToSpawn = 0;
if (RoundCounter == tankRoundStart)
{ _tanksToSpawn = m_tankCount; }
else if (RoundCounter % tankRoundFrequency == 0 && RoundCounter > tankRoundStart)
{
    m_tankCount++;
    _tanksToSpawn = m_tankCount;
}

//Add those tanks to the list of roundEnemies.
for (int i = 0; i < _tanksToSpawn; i++)
{
    int _randomIndex = Random.Range(0, _roundEnemies.Count - 1);
    if (_roundEnemies[_randomIndex] == enemyTypes.Regular)
    {
        _roundEnemies[_randomIndex] = enemyTypes.Tank;
    }
    else
    {
        i--;
    }
}
```

Figure 24: Code-snippet of the enemy queue being created

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 8

6.3.4 - Sprint 9: MVP Finalization

Overview

The progress of the minimum viable product phase had been very rapid up until this point, so there was very little to be done to conclude this stage of development. It was a requirement that the defensive walls were 3D modelled this week, and that any walls placed instantiated a connecting block to any adjacent walls. The user interface also needed some final improvements this sprint, including an actual Book model and tweaks to the user experience of the physical buttons.

Sprint Objectives

1. Final defensive wall improvements
    a. 3D model for both a defensive wall and a connector added

b. Connecting Structure instantiated between walls

2. User experience improvements to the interface
    a. 3D model for a book added
    b. Tweaks to physical buttons

Implementation Methods

A model of a defensive wall was produced in MagicaVoxel and imported into Unity. In order to implement connecting walls, a 'WallConnectionBehaviour' script was added to the wall prefab. This script looks for any adjacent walls to the one it's placed on and instantiates a wall connection between them. It also tracks which connections it has attached to it, so that they're also destroyed when the wall is.

The user interface was greatly improved, through a new model and various other visual changes. The displayed prices of the buildings were linked to the Game Manager, so they're easily modifiable if required. The layout of the book was greatly improved also, so it now shows the timer, the amount of gold the player has, and which round they're on. These changes can be seen in figure 25.

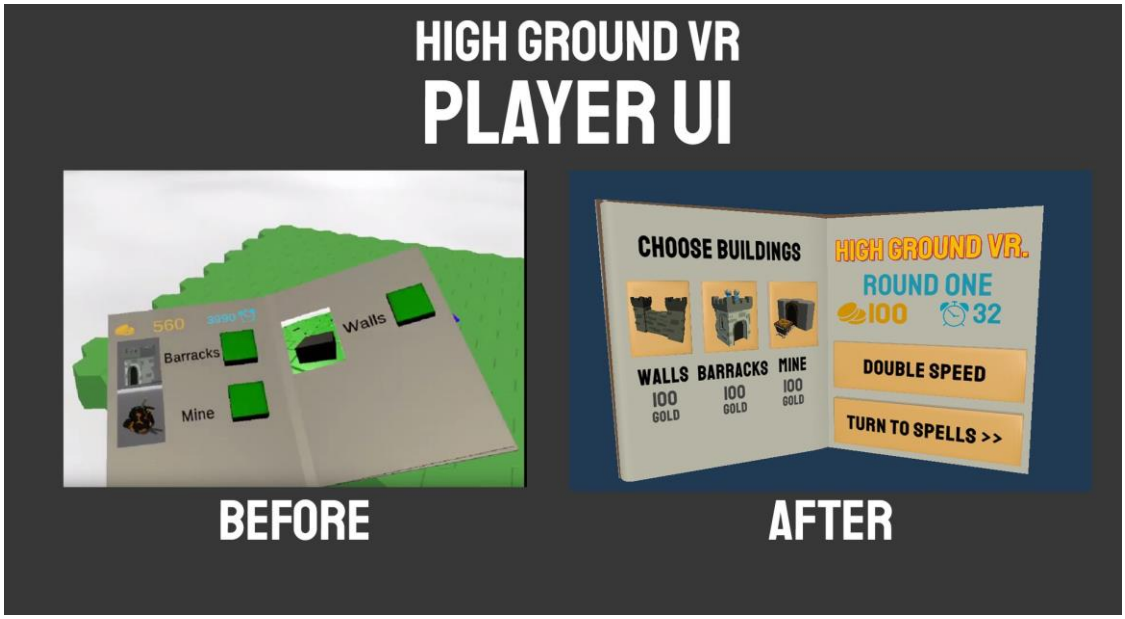


Figure 25: Improvements and additions to the user interface

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 9

6.4 - Usability Preparation

6.4.1 - Sprint 10: Preparing BETA

Overview

This sprint was devoted entirely to producing a playable demo. The additions made during this phase were purely user experience based, ready for usability testing. A new main menu was required during this phase, that used the point and click interaction on a large screen in front of the player. This new menu would display more information and allow players to change various settings about the game.

Sprint Objectives

1. UX tweaks and additions.
    a. Fix the pivot point on the ‘turning the page’ animation on the book UI
    b. Add controller rumble
    c. Dust thrown up when the player shoots the ground with a spell.
2. New Main Menu
    a. Uses the point and click interaction method

Implementation Methods

The user experience tweaks and additions were very straightforward to implement. Changes to animations were made, and particle effects added when the player uses a spell. The rumble uses the vibration class provided in the SteamVR package. This is implemented through a ‘RumbleManager’ singleton, so any script across the project may cause the controllers to vibrate.

The main menu was implemented as a large UI canvas that the player can point at, and then click using the controller’s trigger to select an item. A cursor tracks the intersection point of a raycast from the controller towards the canvas, and the ‘Rect.Overlaps’ function checks whether the cursor is over a button when clicked. Figure 26 shows the final main menu created for the game.

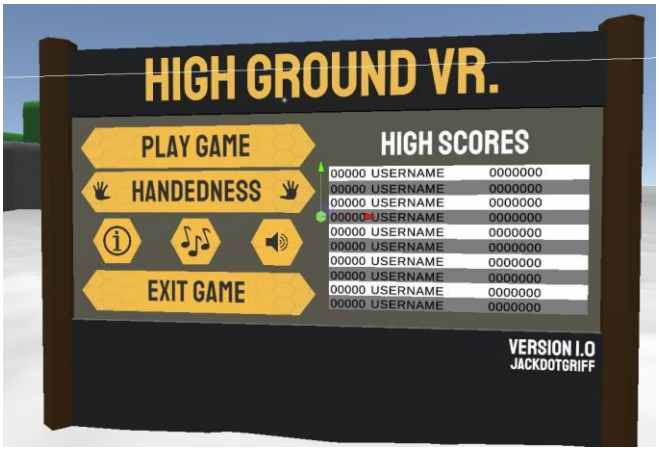


Figure 26: The main menu

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 10

6.5 - Usability Response

6.5.1 - Sprint 11: Usability Response

Overview

Sprint 11 was set aside to act upon any feedback received from usability testing. Due to the coronavirus pandemic, only one play tester was able to play the project in VR. Hence, only a very small amount of feedback was received, and this sprint was relatively clear to work on tasks which may have been postponed until now. This included requirements such as optimization of building placement rules and 3D models for enemy spawns and the gem.

Sprint Objectives

1. Optimise the game rules present in 'ValidateBuildingLocation'
2. Create 3D models for the last two assets that require them.
    a. Gem
    b. Enemy spawn.

Implementation Methods

To start with optimising building placement rules, more exit conditions were added to the loops that check nodes. When a building is placed, it checks a list of rules before it's instantiated. These rules check pathfinding to ensure enemies can still reach the gem, as well as surrounding nodes for certain conditions. Now they're optimised, they will exit loops and reject the placement earlier, hence preventing a possible drop of frame rate.

The models for both the gem and enemy spawn were created in Blender. They were imported into unity and replaced the temporary cube models currently being used. An animation was added to the enemy spawn, which causes the hovering icosphere, seen in figure 27, to spin rapidly when it spawns an enemy. This makes it much easier for a player to see which spawns have been recently active.

Figure 27: The gem and spawn models

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 11

6.6 - Minimum Awesome Product

6.6.1 - Sprint 12: Bug Fixes and Refinement

Overview

This sprint marks the start of the MAP phase, and hence any extra features or refinement resulting in improved user experience can be added. The main requirement of this sprint was that weighted A* pathfinding should be implemented, so that units are able to navigate around potential danger and choose more tactical routes. The second requirement of this phase was that functionality should be added to the audio and handedness settings on the main menu.

Sprint Objectives

1. Weighted A* pathfinding
    a. Enemies should avoid combat and potential danger
2. Settings on main menu
    a. Audio mute toggles, for both effects and music separately
    b. Showing or hiding the info panels.
    c. Handedness toggle (Left or Right-handed)

Implementation Methods

The new implementation of pathfinding valued each node from a start to a goal, and always chose the value of lowest cost.(Swift, 2017) Therefore, modifiers could increase or decrease the cost of a node based upon what surrounded it. Figure 28 shows the costs being added and for what reason a unit may avoid a certain node.

```
/// <summary>
/// Returns a cost for the a node. This is based on it's Type.
/// </summary>
/// <param name="_targetNode"></param>
/// <returns></returns>
2 references
private int nodeCost(Node _targetNode)
{
    switch (_targetNode.navigability)
    {
        case nodeTypes.enemyUnit:
            if (inCombat(_targetNode)) //If a node contains a unit in combat
            {
                return m_enemyInCombatCost;
            }
            else if (inSiege(_targetNode)) //If a node contains a unit in siege
            {
                return m_enemyInSiegeCost;
            }
            else
            {
                return m_enemyUnitCost;
            }
        case nodeTypes.playerUnit:
            return m_playerUnitCost;
        case nodeTypes.mine:
            return m_destructableBuildingCost;
        case nodeTypes.wall:
            return m_destructableBuildingCost;
    }
    //If the node would lead to this unit having to be in combat, avoid it.
    foreach(Node _adjNode in _targetNode.adjecant)
    {
        if(_adjNode.navigability == nodeTypes.playerUnit)
        {
            return m_adjacentToPlayerUnitCost;
        }
    }
    return 0; //0 Cost for empty navigable areas.
}
```

Figure 28: Code-snippet of added node costs

In order to add functional audio settings, audio mixers were created for both music and sound effects. Their volume parameters were exposed, so they can be changed in code, and the appropriate functions for changing the volume added to the buttons on the main menu. In order to show or hide the info panels, the toggle button was attached to an animation of the boards moving up or down out of the surrounding cloud environment. This toggle allows players to get a less obstructed view of their environment if they're familiar with the game.

Handedness considerations were added towards the start of the project but were relatively behind in implementation. In order to make this functional, the book UI was added and removed as the player moved through the main menu, and hence it could be spawned in the player's correct hand when playing. This accessibility consideration allows for both right and left-handed players to play comfortably.

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 12

6.6.2 - Sprint 13: Further Bug Fixes and Refinement

Overview

It was a requirement during this sprint that a spells info panel was added, so that a small description for the available spells is also shown. The lighting and shadows of the game also need to be reviewed. This is because the settings for them have varied slightly after importing the Universal Render Pipeline.

Sprint Objectives

1. Spell info panel
2. Lighting and shadows review

Implementation Methods

An image was made on Photoshop for the spell's help board. The existing help board for buildings was duplicated, and the image replaced. These help boards act as tooltips within the game, prompting a player to what each available action does.

In order to fix the shadows and lighting of the project, a balance was found between distance and cascades on the shadow settings in the render pipeline. This required testing, as getting shadows to appear as good when the camera is large compared to when the camera is small was a balancing act between maintaining both shadow quality and render distance.

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 13

6.6.3 - Sprint 14: Final Additions

Overview

Due to this being the final sprint, it was essential that the final planned feature was added during this week. It’s a requirement of this sprint for Tower upgrades to be added, so that players can upgrade their barracks. For a cost, the player can increase the number of friendly units spawned from a barracks. To match this, the enemy group size needs to be increased as the rounds go on. This is so that the rounds not only become harder through the number of enemy groups, but harder through the size of those groups also. The final requirement of this sprint is for music fading to be implemented, so that the building and attack phase play different music.

Sprint Objectives

1. Tower Upgrades
    a. Player clicks on a barracks and a UI prompt appears.
    b. Purchasing upgrade increases number of units spawned from a barracks
    c. Start with 2 units, with a max of 6 units.
2. Increase enemy group sizes over rounds.
    a. Start with 2 units, with a max of 6 units

3. Music fading between phases
    a. Building phase has a track
    b. Attack phase has a track
    c. Slow fade between them

Implementation Methods

The implementation of tower upgrades was relatively straight forward. It was created as an amalgamation of existing systems and features. Using the existing point and click interaction, the ‘InputManager’ script checked whether the player was pressing the trigger whilst pointing at an upgradeable barracks. If so, a UI prompt appears, like that of the tank health bar, showing the price of upgrading the barracks. As is shown in figure 29, when the player clicks on the same barracks again, the appropriate gold value is taken from them and the maximum number of units the chosen barracks can spawn is increased. This can be done up to 4 times, with a maximum of 6 units spawned.

```
/// <summary>
/// Upgrades the barracks by increasing the amount of units allowed. This will only run if canUpgradeBarracks returns true.
/// </summary>
1 reference
public void runUpgrade()
{

    //Calculate price based on the minimum units and a maximum number of units.
    int _price = Mathf.RoundToInt(Mathf.Pow(m_unitCount, 2) * 50.0f);

    if (!GameManager.Instance.spendGold(_price))
    {
        Debug.Log("Not Enough Money");
        AudioManager.Instance.PlaySound("incorrectSound", AudioLists.UI, AudioMixers.Effects, false, true, false, this.gameObject, 0.1f);
        return;
    }
    AudioManager.Instance.PlaySound("barracksRespawn", AudioLists.Building, AudioMixers.Effects, false, true, false, this.gameObject, 0.1f);
    GameManager.Instance.addScore(_price / 10);
    m_unitCount++; //Upgrade the total amount of units that are associated with this barracks.
}
```

Figure 29: Code-snippet of barracks upgrading

In order to increase the size of enemy groups as time goes on, the equation shown in figure 30 was written. This equation uses the round counter as a multiplier in order to generate a group size, and then a random number added which increases the chance of group size variation as rounds goes on. The example shown below uses 3 as an initial group size, with the blue and green bar showing the upper and lower ranges of group sizes as the round number (x axis) increases.

$$enemyGroupSize = (initialGroupSize - 0.4) + (0.4 * RoundCounter) \pm (0.1 * RoundCounter)$$

Figure 30: Group Size Calculation

In order to fade between music tracks, appropriate coroutines were added to the audio manager that controlled two different audio sources. One played the building music and the other played the attack music. As the phases change, the volume values of each audio source swap over, as to mute one and unmute another. Public music toggle functions were added, which are called from the Game Manager as the phases change.

For the weekly breakdown of this sprint, including success analysis and further refinement, see: Appendix 9 : Week 14

7 - End-Project report

High Ground VR set out to be a project that explored the genre of tower defence, within the medium of virtual reality. The ambitious objective to complete a fully functional VR gameplay loop within a short development time has been successfully completed, with almost all intended features being implemented and refined through usability testing. The following list highlights possible feature improvements;

- **Basic Virtual Reality viewer and controller tracking;** although functional, hand animations are not implemented when a player presses a button.
- **Procedurally Generated Game Board;** although fully implemented, more varied generation of terrain could expand gameplay possibilities.
- **Teleport down onto the game board and around the hexes;** more variations in the first-person experience, such as smaller building details and attacks could thoroughly improve this feature.
- **A* Pathfinding;** Pathfinding is implemented, although is not as perfect as originally intended. Complications with the use of a hexagonal grid have led to paths which aren't always the most optimised.
- **Combat;** More variation in enemy types is always requested of any tower defence project.

Overall, the project was a success. Not only have all objectives of the project been met, but all the base functional requirements were completely implemented, alongside some extra features. The art style of the game is consistent and pleasing to the eye, and the general experience of playing in VR does not cause motion sickness due to a consistent frame rate.

8 - Project post-mortem

8.1 - Reflection and Appraisal

Upon reflection, it is clear that at least one core feature is lacking in intended functionality, the pathfinding. Due to early complications with transferring a square grid algorithm onto a hexagonal grid, the pathfinding fell behind in its complexity, and wasn't fully refined until late during sprint 12. The current state of the feature allows for enemies to navigate to their intended goal, but they behave strangely when more than one path can be taken, or when they're trapped in by players during gameplay. Further refinement would take to revisiting the pathfinding and its implementation, and thanks to it being a rather versatile system, a complete reimplementation wouldn't be hard to build back into the game.

Other than this issue, High Ground VR was a well thought out and methodical project. The early outline of sprints and phases allowed for well-planned weeks of development to roll continuously through with no setbacks. The success of feature implementation has created a well-rounded gameplay loop, which successfully explores the possibilities of tower defence in VR. There are

no clear bugs which would affect this experience, and through usability testing, the gameplay loop has been proven a fun and enjoyable concept.

8.2 - Performance Graphs

Figure 31 shows a graph of the distribution of task types, and the hours spent on each type. The ‘documentation’ type includes time taken for sprint conclusion checklists, and this hour count does not include any time taken for the final report or supporting documentation.

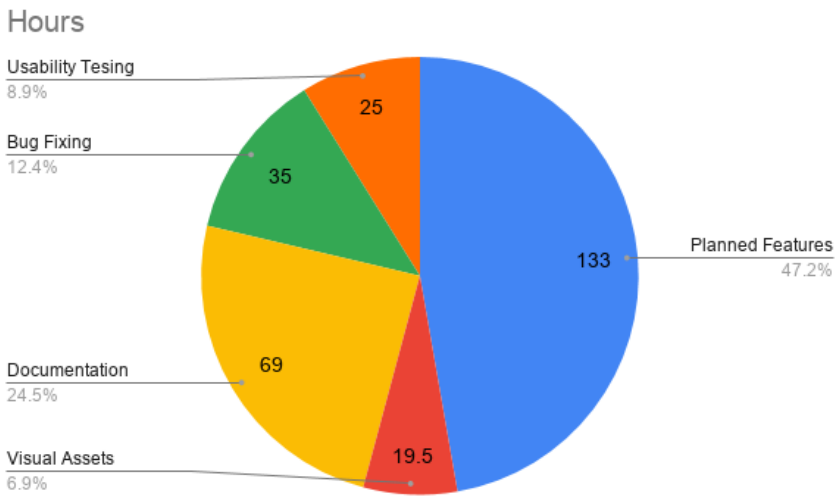


Figure 31: Task Type Distribution

Figure 32 shows the cost estimate for the project. Where tasks took longer than expected, the percentage is highlighted in red, where they took less than assumed is yellow and correct is green. It is evident from the graph that most tasks took less time than originally assumed.

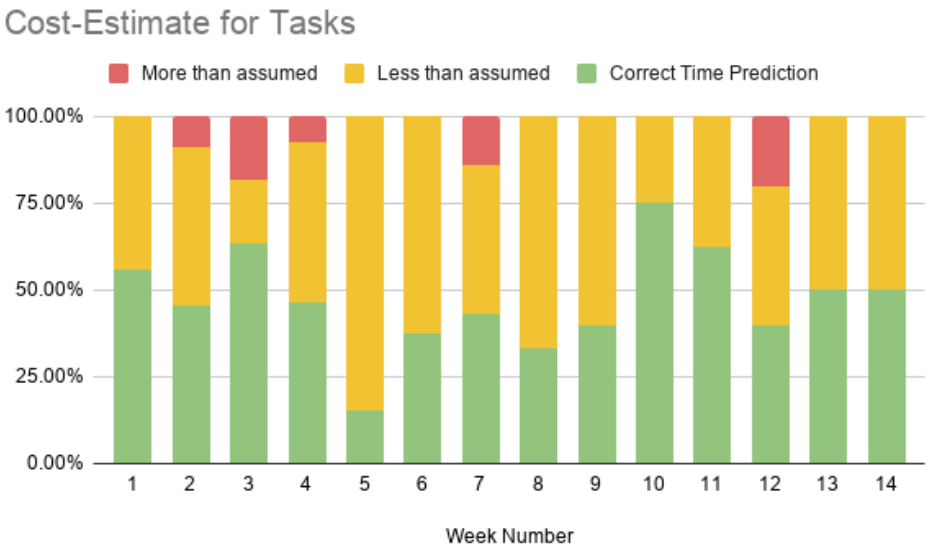


Figure 32: Cost-Estimate Graph

8.3 - Conclusions

Developing a VR game that explores an unsaturated genre was quite a challenging experience. The lack of defined connotations allowed for creative freedom but was restrictive as the game still needed to fit within the confines of the genre. Due to the short development time of this project, there wasn't much allowance for explorations into large prototypes of systems. However, the result is still an enjoyable tower defence experience, that aims to innovate in terms of allowing players to be more creative with how they plan their defences.

9 - References

Avionx (2019) *Skybox Series Free*, *Unity Asset Store*. Available at: https://assetstore.unity.com/packages/2d/textures-materials/sky/skybox-series-free-103633 (Accessed: 23 April 2020).

Bourg, D. M. and Seemann, G. (2004) *AI for Game Developers*. 'O'Reilly Media, Inc.'

Brechner, E. (2015) *Agile Project Management with Kanban*. Microsoft Press.

Drake, S. (2010) *Hexmap Coordinates*, *3DM Design*. Available at: http://3dmdesign.com/development/hexmap-coordinates-the-easy-way (Accessed: 6 May 2020).

French, J. L. (2020) *Ultimate Game Music Collection*, *Unity Asset Store*. Available at: https://assetstore.unity.com/packages/audio/music/orchestral/ultimate-game-music-collection-37351 (Accessed: 13 February 2020).

Imphenzia (2017) *Universal Sound FX*, *Unity Asset Store*. Available at: https://assetstore.unity.com/packages/audio/sound-fx/universal-sound-fx-17256 (Accessed: 13 April 2020).

Red Blob Games: Hexagonal Grids (2013). Available at: https://www.redblobgames.com/grids/hexagons/ (Accessed: 14 April 2020).

Swift, N. (2017) 'Easy A* (star) Pathfinding - Nicholas Swift - Medium'. Medium. Available at: https://medium.com/@nicholas.w.swift/easy-a-star-pathfinding-7e6689c7f7b2 (Accessed: 6 May 2020).

Virtual Reality (VR) - Statistics and Facts (2020) *Statista*. Available at: https://www.statista.com/topics/2532/virtual-reality-vr/ (Accessed: 5 May 2020).

10 - Appendices

Appendix 1: Project Links

Repositories / Code

**Note: Supervisor has been given access to the repo, external markers may require permission to access the repository. If needed, please email jack.griffiths@students.plymouth.ac.uk **

Code Repository [Link] *: The project's main repository*
Asset Repository [Link] *: The repository used to store all asset development files. (e.g. .psd files)*

Build Download [Link] : *Download a build from Itch.io. The password is 'CGDPlymouth'*

Project Management

Trello Board [Link] *: The main Kanban board for the project*
Trello Archive Board [Link] *: Archive of all tasks completed on the project. Organised per phase.*
Weekly Development Reports + Functionality Testing [Link] *: Weekly report spreadsheet.*
Visual Research and Development [Link] : *User Story and Mood Boards*

Social Media

Twitter Moment [Link] *: Collation of all tweets made during development*
Tumblr Devlogs [Link]*: Weekly devlogs on progress made.*
YouTube Development Playlist [Link] *: Archive of video evidence of each implemented feature.*

Appendix 2: User Guide

Installation Instructions

From the repo

1. Download the 'Code Repository' from Appendix 1.
2. Open the 'High Ground VR V1.0 Build' folder
3. Run the 'HighGroundVR.exe' file to play the game. (SteamVR should launch automatically)

From a download link

1. Go to https://jackdotgriff.itch.io/high-ground-vr
2. Enter the password 'CGDPlymouth'
3. Download the 'High Ground VR.zip'
4. Extract the files to the desired location
5. Run the 'HighGroundVR.exe' to play the game. (SteamVR should launch automatically)

System Requirements

Processor: Intel Core i5-4590/AMD FX 8350 equivalent or better

GPU: NVIDIA GeForce GTX 1060, AMD Radeon RX 480 equivalent or better

Memory: 4 GB RAM or more

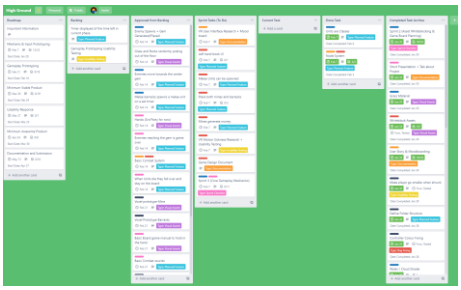
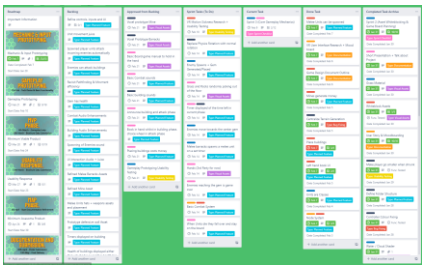
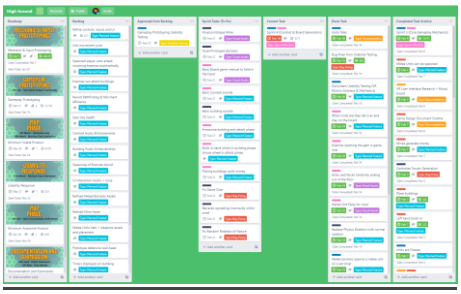
Video output: HDMI 1.4, DisplayPort 1.2 or newer

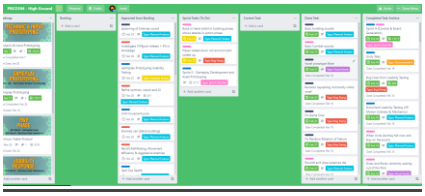
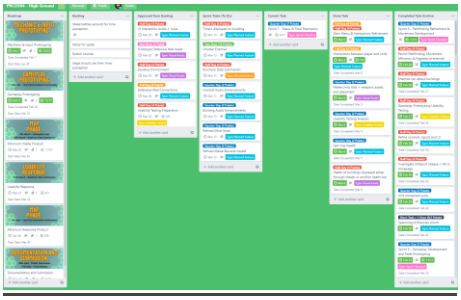
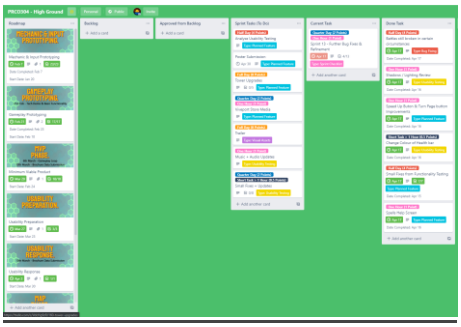
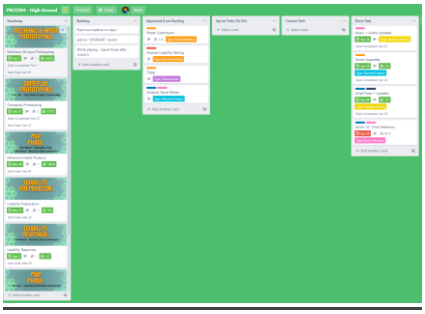
USB port: 1x USB 2.0 or newer

Operating System: Windows 7 SP1, Windows 8.1 or later, Windows 10

Game Controls

Appendix 3: Submitted Trello Screenshots

Trello Board 1 (Deadline 31/01/20 5pm) Assignment		Trello Board 2 (Deadline 07/02/20 5pm) Assignment	
Trello Board 3 (Deadline 14/02/20 5pm) Assignment		Trello Board 4 (Deadline 21/02/20 5pm) Assignment	

Trello Board 5 (Deadline 28/02/20 5pm) Assignment		Trello Board 6 (Deadline 13 March 5pm) Assignment	
Trello Board 7 (Deadline 27/03/20 5pm) Assignment		Trello Board 8 (Deadline 24/04/20 5pm) Assignment	
Trello Board 9 (Deadline 8/05/20 @ 5pm) Assignment			

Appendix 4: Game Design Document

High Ground Game Design Document

This Game Design Document (GDD) has been completed before the start of development of any gameplay elements of the project. It details the basic gameplay loop, core mechanics behind the project, details on the art style and further considerations on accessibility.

Genre

The game is a VR Tower Defence game, with a medieval theme. It features melee ‘turn-based’ style combat, and allows for the player to play strategically, creating effective paths to complete rounds, and become the same size as a unit in game to compete using FPS style magic-abilities. It is played on a terrain built of hexagons, with one building taking up one hexagon.

Breakdown

- Virtual Reality
- Tower Defence
- Medieval Combat
- Strategic Minigame
- FPS Abilities
- Hexagonal Terrain

Innovation

This project is relatively innovative within the genre. There are a very limited number of virtual reality tower defences in the market, meaning there is very little competition. This project innovates by using Hexagonal terrain, relatively unseen in the VR market. Secondly, the enemies in High Ground will have a very basic level of artificial intelligence, as they will make informed decisions on their routes to the centre of the game board based on the buildings placed on the player. For example, if the route built by the player is very complex, there’s a higher chance that enemies will aim to just smash through all the walls instead of abiding by the route.

Breakdown

- Hex Terrain
- Intelligent Enemies that don’t always abide path rules
- Place buildings on any hex
- Fast paced gameplay resulting in enjoyable spectating

Gameplay

Goals

The main goal of the game is to protect the gem in the centre of the game board from incoming attackers. The player does this by building walls and barracks that spawn their own units, in order to create paths from the enemy spawn points into their traps. The round ends either when the player defeats all the enemies, or their gem is destroyed which ends the game. If they’re

successful in completing a round, the number of incoming enemies increases, and the player can continue to build.

Breakdown

- Protect the gem in the centre
- Defend the gem from incoming enemy attackers
- Build walls and buildings to create paths leading enemies from their spawns into traps.
- Defend for an entire wave, at the end of which the number of incoming enemies increases.
- If the gem gets destroyed, the game is over.

User Skills

In order to play this game, the player must have experience with using the basic VR interactions. This includes accurately pointing and clicking with the trigger on a VR controller, as well as using the trackpad button to teleport around the game world. Due to the project being driven for VR, the controls should be more intuitive than that of an alternative input method. In order to succeed, the player must be able to tactically build paths and traps to strategically beat the round, though this only needs to come after a few easy introductory rounds.

Breakdown

- Point and Click with a VR controller
- Teleport with a VR controller
- Tactical building of paths and walkways
- Understanding of game mechanics to aid with strategic gameplay after a few easy rounds.

Game Mechanics

The core building mechanics consist of placing barracks to spawn melee units that automatically get into combat with incoming enemies, Spawn defensive walls which guide enemies around the board, and placing mines which generate gold to be spent on more buildings. The player can then teleport onto the board and use a variety of different attacks to inflict extra damage on incoming enemies. The enemies themselves will spawn from the edge of the map, and their movement will resemble some form of intelligence to improve variety in gameplay. They will make their decisions based on the complexity of the paths created by players, so more complex paths will be more likely to create enemies that ignore it and just attack the buildings to get through them.

Breakdown

- Building Tactically to create paths
- Automatic Intelligent enemy movement
- Teleporting down onto the board
- Zapping and Interacting with Enemies

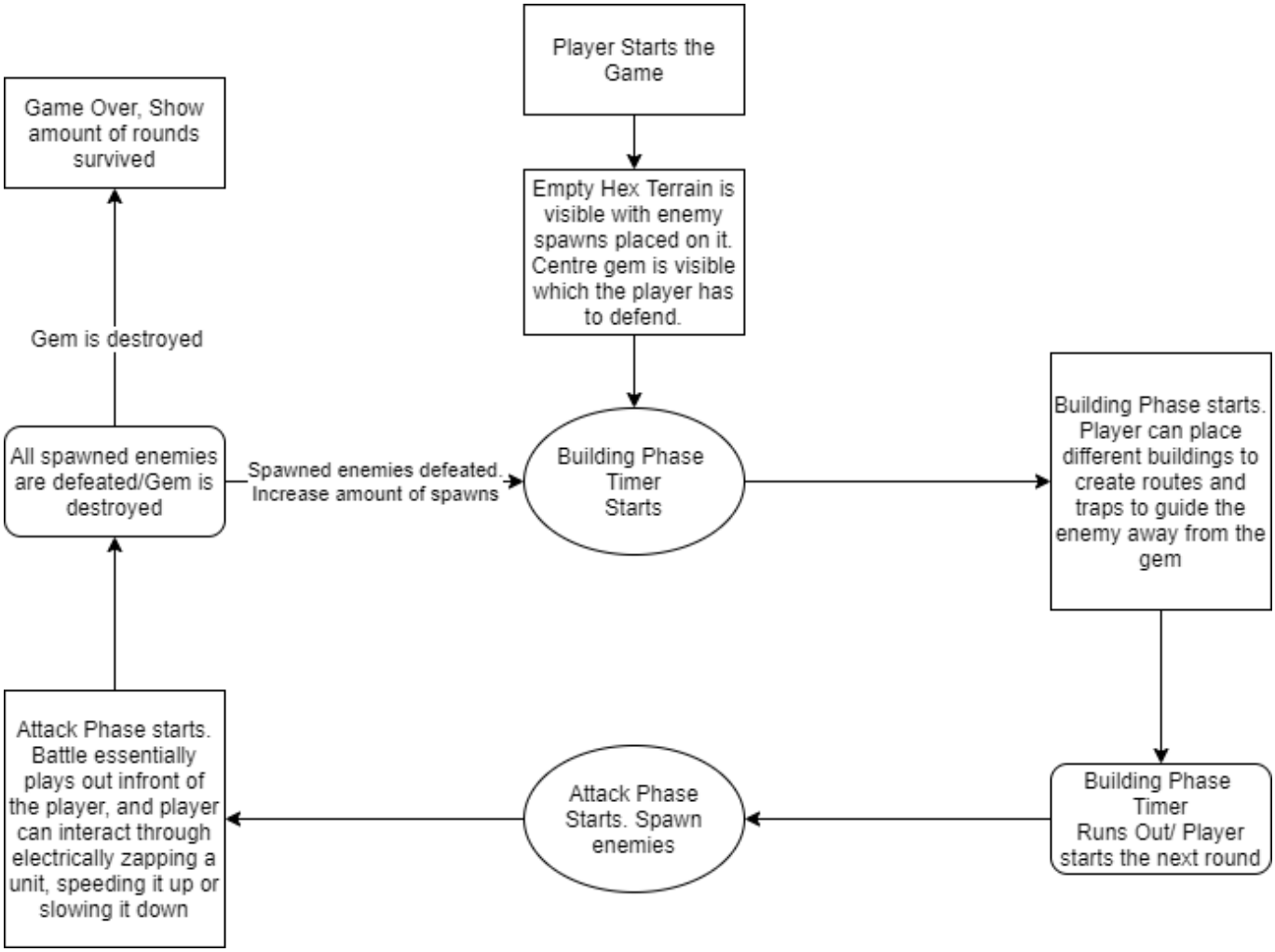
Gameplay Loop

The gameplay loop is broken down into 2 phases, the building phase and attack phase. When the game is started, a hexagonal game board is visible, with some enemy spawns around the

edge of the map. A timer starts, signifying the time left of the building phase. The player will recognise that there is also a gem in the centre of the board which they need to defend, through building walls, barracks and mines to guide and trap enemies. When the timer runs out, the Attack Phase starts, and enemies start to spawn around the visible spawn blocks and make their way towards the gem. The player then has extra attacks they can inflict during this phase, including a speed up, slow down and electric zap which will damage that unit. As the round ends, the amount of enemies spawning will increase, and the player can build again.

Breakdown

- Building Phase
  - Teleport on and off the board using the trackpad
  - Book UI to select building choice
  - Place buildings using the trigger
  - Plan and develop paths from the enemy's spawn point as to guide them into traps you've set
  - Timer limited
  - Incoming enemy spawns visible
- Attack Phase
  - Enemies spawn from spawn points, and move towards the gem
  - Intelligent pathfinding leads them a variety of ways
  - Attack the enemies using a variety of different powers (Zap, Speed Up, Slow Down etc.)
  - Once enemies are all defeated, round is over then you go back to building phase in order to repair + build more structures
  - Number of enemies in next wave increases
  - Maybe a new spawn appears around the edge that the player will have to prepare for.



Player Abilities/Items

The player will have a certain amount of abilities and objects they can place during each phase. During the building phase, they will have an option of 3 different buildings. Firstly, a defensive wall, this will simply act as a barrier to guide enemies around the map. Secondly, a barracks, which spawns friendly units that defend a certain area and its adjacent hexes. Finally, a mine, which generates gold which can be spent on these buildings. The player will have to spend the gold they generate on placing these buildings.

During the action phase, the player will be unable to place any buildings. They will however be able to spend their gold on using their ‘magic’ abilities. This will include zapping a unit to damage them, speeding their movement up, or slowing them down. The goal of these abilities is to allow players to still damage enemies even if they can’t place buildings anymore, just in case an enemy manages to slip through their defences.

Breakdown

- Gold - Used to purchase buildings
- Place a Barracks - Spawn Units in one of the adjacent nodes
- Place a Mine - Generate Gold for the player
- Place Walls - Cheap defensive structure

- Defensive ‘Magic’ abilities - (Damage, Speed Up, Slow Down)
- Expansion Opportunities (Turrets, Healing Units, Ranged Barracks)

Progression and Challenge

Progression in High Ground is wave-based. Players will be given a certain amount of time to place buildings, and once that time is up, they will be unable to place anymore, and a certain number of enemies will spawn. Once all these enemies are defeated, the round is over, and the player is given time to build again. The number of enemies spawning next round will increase based on the amount of times they have completed action rounds. If the player fails to defeat the enemies, and an enemy reaches the central gem, the game is over, and a score is given. The score will be calculated based upon current gold, round number reached and a variety of other factors.

Challenge increases through adding more enemies each round, adding more enemy spawns, and increasing the chance of spawning aggressive enemies that like to destroy buildings.

Breakdown

- Waves of enemies
- Survive the waves
- Enemies reaching the gem is game over
- Score given based on current gold, round reached and other factors.
- Increasing Difficulty through adding more enemies
- Increasing Difficulty through adding more enemy spawns
- Increasing Difficulty through increasing chance of spawning aggressive enemies that like to destroy buildings

Losing

The player has lost when the gem is destroyed. Once the score is given, the player will be given the choice to restart, which will reset the map and restart the game. High scores will also be displayed to the player.

Breakdown

- Gem getting destroyed
- Player will be able to restart
- High scores.

Game Economy

At this stage of the project, it is difficult to determine certain prices of buildings without having working gameplay, however the game economy can be planned on a base level. The player will be given an amount of money at the start of each round, which they can use to spend on buildings. If they place a Mine, the mine will generate money on a set interval, adding to the money a player will have on the next round. A player’s final score will have a certain amount of points added for gold they have at the end. The entire game economy will need to be balanced over quite a long period, so It is essential that gameplay is developed quickly to ensure this can be done properly.

Breakdown

- Player given an amount of money at the start of each round
- Mines generate Money
- Buildings cost a certain amount of money
- Player's score will be increased based on their current money at the end of the game.

Game Rules

The rules of gameplay need to resemble that of a typical 2D board game, as the game itself doesn't operate on the 3rd dimension. The following lists breakdown the rules of each game element, including where they can be placed and rules on their behaviour.

Pathfinding

In order to implement pathfinding, a graph of nodes is required. Each of these nodes will have a 'navigability' property which determines how enemies interact with the node.

- *navigable:* Assigned to nodes that are completely empty, available for enemies to navigate through.
- *destructible:* Assigned to nodes that contain buildings placed by the player. These nodes are usually avoided, but the enemy will be able to destroy them to create a new path.
- *nonplace able*: Assigned to nodes that are navigable, but don't allow for any building on. This includes adjacent nodes to enemy Spawns and the gem.
- *Gem*: Assigned to the node that contains the gem, as interaction with this is different to normal destructible objects.
- *player Unit*: Assigned to nodes that contain a player Unit. If all the units on that node become destroyed, the unit becomes *navigable* until a new unit is spawned again.
- *enemy Unit*: Assigned to nodes that contain enemy Unit groups.
- *enemy Spawn*: Assigned to nodes that contain the enemy Spawns.

Mines

Placement

- *Needs to be placed on an empty Node.*
- *Can't be placed on the outer edges of the map.*
- *Can't be placed on any nodes adjacent to an enemy spawn.*
- *Can't be placed in a location that completely blocks the route to the gem.*

Behaviour

- ❖ Has health, destroyed if health reaches 0.
- ❖ Generates gold on a set timer

Barracks

Placement

- *Needs to be placed on an empty Node.*
- *Can't be placed on the outer edges of the map.*
- *Can't be placed on any nodes adjacent to an enemy spawn.*
- *Can't be placed in a location that completely blocks the route to the gem.*

- *Needs to have an adjacent node clear in the correct direction to spawn units on.*
- *Cannot spawn units on a node adjacent to an enemy spawn.*

Behaviour

- ❖ Has health, destroyed if health reaches 0.
- ❖ Associated units are also destroyed if the building is destroyed.
- ❖ Immediately spawns units on the empty ‘forward’ node to defend adjacent nodes.
- ❖ Starts the battle and manages units within a battle.
- ❖ Respawns units after a set timer if a unit is destroyed in combat.

Defensive Walls

Placement

- *Needs to be placed on an empty Node.*
- *Can’t be placed on the outer edges of the map.*
- *Can’t be placed on any nodes adjacent to an enemy spawn.*
- *Can’t be placed in a location that completely blocks the route to the gem.*

Behaviour

- ❖ Has health, destroyed if health reaches 0.
- ❖ Acts as a barrier to divert enemy units.
- ❖ When placed, draws walls between itself and adjacent defensive walls.

Enemy Spawns

Placement

- *Automatically placed on the outer edges of the map.*
- *Can’t be placed on any nodes adjacent to an enemy spawn.*

Behaviour

- ❖ Spawn enemy units in adjacent nodes on a timer
- ❖ Only spawns up until the limit of the number of enemies per round is reached

The Gem

Placement

- *Needs to be placed on an empty Node.*
- *Can’t be placed on the outer edges of the map.*
- *Can’t be placed on any nodes adjacent to an enemy spawn.*

Behaviour

- ❖ Has health, destroyed if health reaches 0.
- ❖ Game is over once it’s destroyed

Enemy Units

Placement

- *Needs to be spawned by an enemy spawner on an empty Node.*
- *Can only navigate onto empty nodes.*

Behaviour

- ❖ Has health, destroyed if health reaches 0.
- ❖ Has damage, used during battles.
- ❖ Contain multiple units on one node.
- ❖ Uses A* pathfinding to navigate around the map.
- ❖ Aims to reach the gem and attack it.
- ❖ Has an aggression value which determines their choices on whether to navigate through buildings or not
- ❖ Can destroy all buildings placed by the player.
- ❖ Move as a group, with each unit having individual damage and health.

Player Units

Placement

- *Needs to be placed on an empty Node.*

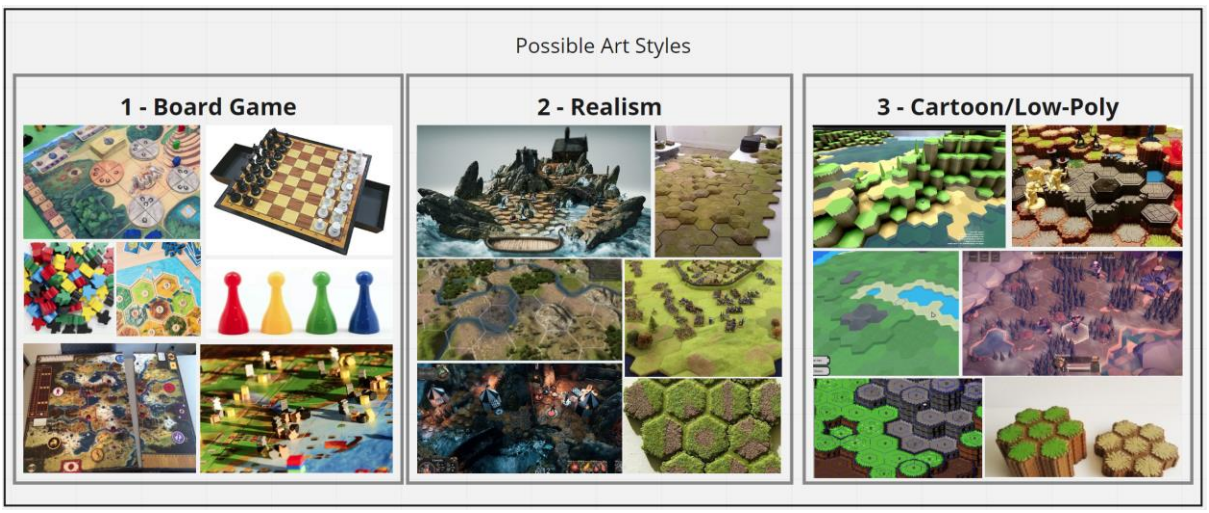
Behaviour

- ❖ Has health, destroyed if health reaches 0.
- ❖ Has damage, used during battles.
- ❖ Contain multiple units on one node.
- ❖ Stays in the node adjacent to the door of the barracks.
- ❖ Get into combat with enemies when they appear in adjacent nodes.

Art Style

Mood Boards

These Mood Boards aim to explore the various Art Styles and models that could be implemented in this project. They are all available in full resolution on the Visual Asset Planning board.



In order to determine the Art Style for this project, 3 styles were explored:

1. **Classic Board Game** - This art style resembles the classic board game style of Wood, Cardboard and hand drawn artwork. This would be a great style for a non-VR board game, however, it wouldn't be ideal for this 3D project due to the flatness of the game board, as well as

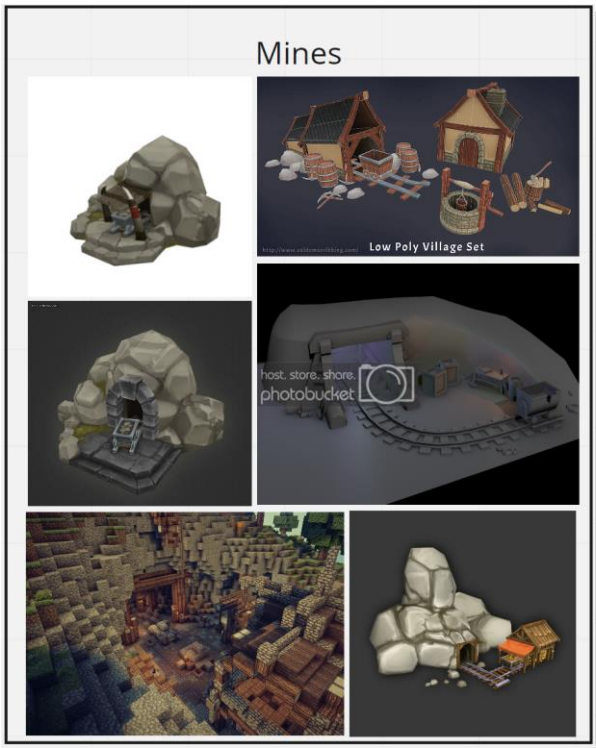
the time taken to draw the artwork for the board itself. However, Reference is going to be taken from the simplicity of the game pieces, people and buildings, as they are quite effective and quickly implemented.

2. Natural Realism - Although still resembling a game board, this art style aims to replicate realistic topology and foliage through realistic shaders and models that are placed together in a game-board shape. Time and GPU restraints wouldn't allow for this style to look effective, and It would be very difficult for it to be accomplished within the time constraints.

3. Cartoon/Low-Poly - This style uses bold colours and low-polygon models to create a really effective mix between Art Style 1 and 2. This style will be quicker to prototype and creates a really nice style with minimal effort, but also allows for a lot of expansion, making it very scalable depending on project scope.

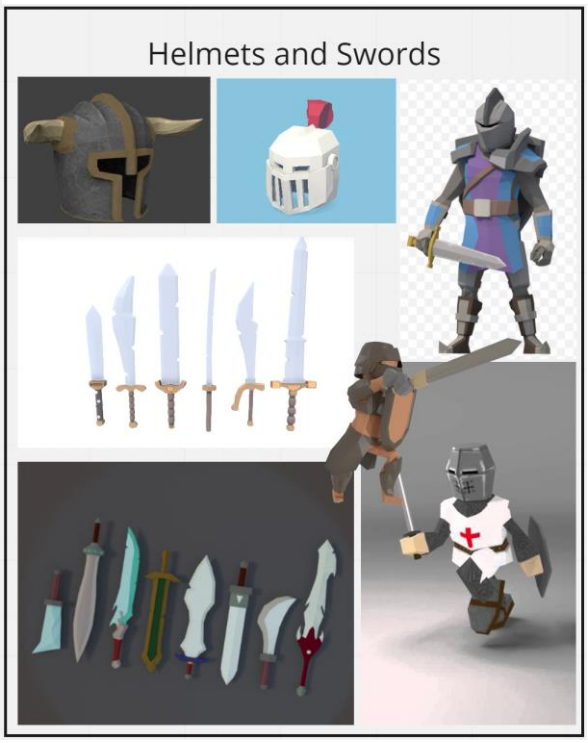
For the reasons above, A mixed approach of Art Style 1 and 3 has been chosen. The environment will be low poly, with bold colours so that any buildings or units on them are easily visible. This will lend itself well to the strategic gameplay that requires the player to easily see all the units on the board. Inspiration will also be taken from the classical board game pieces for use as assets, for example the trees and buildings will resemble cardboard and wood. The following mood boards aim to explore more specific assets, and possibilities regarding their style that fit within the chosen art style.

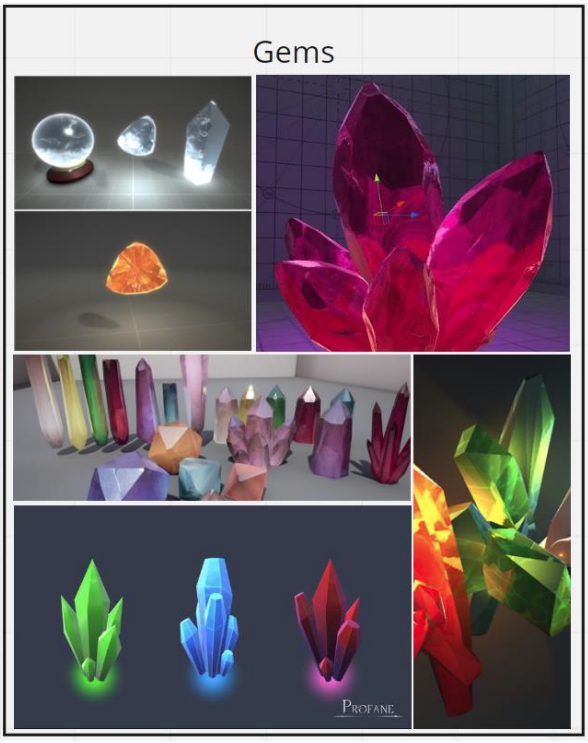
Environmental Features


Mines
Low Poly Village Set
host. store. share.
photobucket


Defensive Walls


Barracks


Helmets and Swords



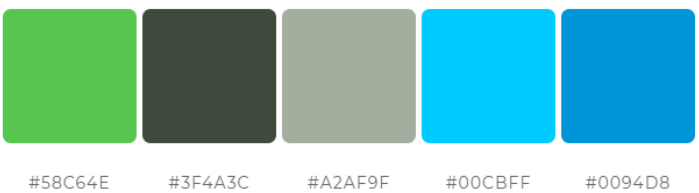
Fonts

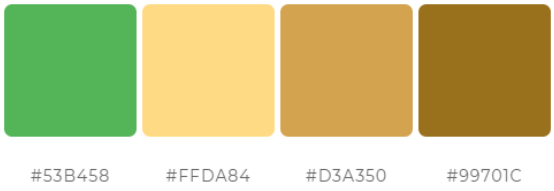
The font chosen is Staatliches. Staatliches is a clean-cut display face with charmingly unconventional proportions. This font is very bold and can easily be styled to meet any colour scheme or art style. The font will be the main font for all titles and presentations developed for the project. Staatliches is a free Google Font, which uses the OFL (Open Font License), allowing for free modification and commercial distribution of the font as long as the license is shown.

High Ground VR.

Colour Schemes

The project itself will have very bright and bold colours, to resemble that of a cartoon-style board game. The colours of logos and branding will contain bold greens and greys, to resemble the medieval landscape this game is aiming to achieve. The tool ColorSpace has been used to generate a few possible palettes to explore.





Music and Sounds

The diegetic audio within High Ground VR will be mostly created through Foley. It will resemble sounds like that of playing an actual board game, such as wooden board game pieces being moved around the map. There will also be small yells and crashing of swords during combat, which will help with the immersion when the player is shrunken down onto the board.

Breakdown

- Wooden Sounds when buildings placed
- The people on the board make small yelling or chirping sounds when attacking
- Metal crashing on metal when swords hit shields
- Music appropriate to the theme (Currently an unexplored area)

Technical Description

Technologies and Tools

High Ground VR is being developed using Unity3D and C#. This is due to its easy accessibility and the fact it's completely free. The SteamVR plugin for Unity will be used to implement all the VR controls and tracking, as SteamVR is the standard currently for VR gaming. Blender and Photoshop will be used for development of 2D and 3D Assets, and Audacity will be used for editing of audio. All this software is readily available within the University. In terms of technology being used, an HTC Vive with a Vive Tracker is going to be used. This is due to its currently low second-hand price, compared with other alternatives. A Valve Index will be used within the university for demonstration purposes, as it has higher specification screens.

Breakdown

- Unity3D
- C#
- SteamVR
- Blender
- Photoshop
- Audacity
- HTCVive with Vive Tracker
- Valve Index

Platforms (Present and Possible)

High Ground VR will be built for SteamVR and released to the VivePort store during the module. The project is being developed with an HTC Vive, so the control scheme will be optimised accordingly, however, it's easy to add more controller mapping for platforms like the Valve Index. The goal of this project is to have very modular code, and a desktop release of the game is not completely out of the question for this reason.

Breakdown

- Steam VR
- HTC Vive Controller Mapping
- Valve Index Controller Mapping
- Possibilities of a desktop release due to the adaptable design of code.

Audience and Marketing

Demographics

A PEGI Rating of 7+ will be the goal of this project, and due to the mild violence between non-human characters it will be quite easy to achieve. The Viveport rating system will need to be explored so that there are no issues with ratings on the store listings. Regarding the target demographic, it's more aimed towards players of ages 13 upwards, due to the complexity of using VR controls. For this reason, a desktop release should be explored in order to fully reach the market.

Social Media Presence

The goal of the development is to have a very transparent development process. This is for the purpose of development tracking and marketing the project. Twitter should be used continuously throughout, with at least 2 or 3 tweets per week using relevant hashtags. Some of the content of these tweets should also be posted on Reddit, as Twitter tends to reach developers more than it does actual customers. A weekly devlog should be posted on Tumblr, which aims to archive sprintly development through videos and short descriptions.

Platforms

Development will eventually lead up to either a Viveport or Itch.io release. A Viveport developer account has been created, however if there are any issues with this release then itch.io will be the alternative. Both are free to use platforms for Indie developers to release, however Viveport is the easiest gateway into VR game sales.

Accessibility

Handedness

The entire project relies on a player using one hand much more than the other, so handedness should be taken into consideration when making all the project's inputs. This will then be a toggle that can change for people while testing and added into the settings screen in-game later in the project.

Colour Blindness

Colour blindness will remain an important consideration whilst designing assets and user interfaces. This works quite well with the chosen art style as all the assets will be bold coloured and easier to distinguish from each other.

Mood Board Image References

Art Styles

2020. [online] Available at: <https://forum.unity.com/threads/honey-hex-framework.265154/> [Accessed 8 May 2020].

aliexpress.com. 2020. US $25.91 19% OFF|150 Pieces Wooden Game Pieces/Pawn/Chess Meeples Carcassonne 16Mm*16Mm For Board Game|Board Games| - Aliexpress. [online] Available at: <https://www.aliexpress.com/item/32898028343.html> [Accessed 8 May 2020].

Amazon.co.uk. 2020. [online] Available at: <https://www.amazon.co.uk/Quickdraw-Traditional-Chess-Plastic-Storage/dp/B01MSTHYWL> [Accessed 8 May 2020].

Discussion, G., 2020. Hexagonal Terrain Mesh. [online] jMonkeyEngine Hub. Available at: <https://hub.jmonkeyengine.org/t/hexagonal-terrain-mesh/38490> [Accessed 8 May 2020].

King, F., 2020. For The King. [online] For The King Wiki. Available at: <https://fortheking.gamepedia.com/For_The_King> [Accessed 8 May 2020].

Pinterest. 2020. Even More Hex Terrain Experiments | Hex Tile, Hex, Tiles Game. [online] Available at: <https://www.pinterest.com/pin/502362533407904283/> [Accessed 8 May 2020].

Printninja.com. 2020. [online] Available at: <https://printninja.com/custom-plastic-board-game-pieces> [Accessed 8 May 2020].

Robertson, A., 2020. 30 Amazing Board Games Not To Miss In 2019. [online] Forbes. Available at: <https://www.forbes.com/sites/andyrobertson/2019/02/21/30-amazing-board-games-not-to-miss-in-2019/#508ddb9467fc> [Accessed 8 May 2020].

Stonemaier Games. 2020. Game Board Extension (Scythe). [online] Available at: <https://stonemaier-games.myshopify.com/products/game-board-extension-scythe-asia> [Accessed 8 May 2020].

Store.steampowered.com. 2020. Save 60% On WARTILE On Steam. [online] Available at: <https://store.steampowered.com/app/404200/WARTILE/> [Accessed 8 May 2020].

Thingiverse.com. 2020. Hex Terrain By Mrhers2. [online] Available at: <https://www.thingiverse.com/thing:1775936> [Accessed 8 May 2020].

Thingiverse.com. 2020. Hex Terrain Tiles By Grumpusbumpus. [online] Available at: <https://www.thingiverse.com/thing:2761952> [Accessed 8 May 2020].

Unsplash.com. 2020. 27+ Board Game Pictures | Download Free Images On Unsplash. [online] Available at: <https://unsplash.com/s/photos/board-game> [Accessed 8 May 2020].

Wargamingtrader.com. 2020. Nice Use Of Hex Terrain. | The Wargaming Trader. [online] Available at: <http://wargamingtrader.com/img/nice-use-hex-terrain> [Accessed 8 May 2020].

Wirecutter: Reviews for the Real World. 2020. Best Board Games For Adults 2020. [online] Available at: <https://thewirecutter.com/reviews/board-games-for-adults/> [Accessed 8 May 2020].

2020. [online] Available at: <https://www.youtube.com/watch?v=yeKRtWDDezM> [Accessed 8 May 2020].

KENOTICKET. 2020. Heroscape 7 Hex Terrain Tiles, 6 Grass, 2 Sand. 8 Pcs Total. [online] Available at: <https://www.kenoticket.net/heroscape-7-hex-terrain-tiles-6-grass-2-sand-8-pcs-total/pd9768724> [Accessed 8 May 2020].

Vimeo. 2020. Unity Hex Terrain Generator. [online] Available at: <https://vimeo.com/58782514> [Accessed 8 May 2020].

VR User Interfaces and Interactions

2020. [online] Available at: <https://virtualrealitycasinos.uk/gonzos-quest-vr-slot-review/gonzos-quest-vr-menus/> [Accessed 8 May 2020].

Apps, G., 2020. Oculus Discussion Forums. [online] Oculus. Available at: <https://forums.oculusvr.com/community/discussion/68667/beat-saber-community-rankings/p6> [Accessed 8 May 2020].

Magee, M., 2020. 'Grav|Lab' Early Access Review – Road To VR. [online] Road to VR. Available at: <https://www.roadtovr.com/gravlab-early-access-oculus-rift-touch-review/> [Accessed 8 May 2020].

Medium. 2020. VR Design Review #2: Menus. [online] Available at: <https://blog.sketchbox3d.com/vr-design-review-2-menus-b0d7ddc3078?gi=9c4234ac3c1b> [Accessed 8 May 2020].

Medium. 2020. VR Design Review #2: Menus. [online] Available at: <https://blog.sketchbox3d.com/vr-design-review-2-menus-b0d7ddc3078> [Accessed 8 May 2020].

Sproll, D., 2020. Virtual Reality Menu Design - Part 1. [online] Re-flekt.com. Available at: <https://www.re-flekt.com/blog/virtual-reality-menu-design-part1> [Accessed 8 May 2020].

Unrealengine.com. 2020. VR Integrator Radial And Dockable Menus By W3 Studios In Blueprints - UE4 Marketplace. [online] Available at: <https://www.unrealengine.com/marketplace/en-US/product/vr-integrator-radial-and-dockable-menus> [Accessed 8 May 2020].

Defensive Walls

References

3DOcean. 2020. Low Poly Dungeon Wall. [online] Available at: <https://3docean.net/item/low-poly-dungeon-wall/8249959> [Accessed 8 May 2020].

2020. [online] Available at: <https://www.cgstudio.com/3d-model/fantasy-stone-castle-325465> [Accessed 8 May 2020].

Critiques, 3., 2020. Low Poly Castle Critique. [online] polycount. Available at: <https://polycount.com/discussion/168520/low-poly-castle-critique> [Accessed 8 May 2020].

Models, V., model, L. and model, L., 2020. LOW POLY CASTLE LEVEL5 Game Asset | 3D Model. [online] CGTrader. Available at: <https://www.cgtrader.com/3d-models/architectural/other/low-poly-castle-level5> [Accessed 8 May 2020].

NetrinoMedia, I., 2020. Low Poly Stone Walls Small 3D Model In Buildings 3Dexport. [online] 3DExport. Available at: <https://3dexport.com/3dmodel-low-poly-stone-walls-small-107258.htm> [Accessed 8 May 2020].

Wall, P., 2020. Download Low Poly Castle Wall Transparent PNG On Yellow Images 360°. [online] Yellowimages.com. Available at: <https://yellowimages.com/images-360/products/low-poly-castle-wall-yi3602393?ca=2_16> [Accessed 8 May 2020].

Barracks

Castle, P., 2020. Download Low Poly Castle Transparent PNG On Yellow Images 360°. [online] Yellowimages.com. Available at: <https://yellowimages.com/images-360/products/low-poly-castle-yi3602121?ca=2_16> [Accessed 8 May 2020].

Drink, F., Design, I., Models, F., Models, 3., Models, M., Models, C., Models, B., Models, L., Models, A., Models, R., Models, O., Models, F. and model, C., 2020. Castle 3D Model - Turbosquid 1176018. [online] Turbosquid.com. Available at: <https://www.turbosquid.com/3d-models/castle-3d-model-1176018> [Accessed 8 May 2020].

Matthews, A., 2020. Santorini Review – Playground Of The Gods. [online] Meeple Mountain. Available at: <https://www.meeplemountain.com/reviews/santorini-review-playground-of-the-gods/> [Accessed 8 May 2020].

Monopoly Wiki. 2020. Houses. [online] Available at: <https://monopoly.fandom.com/wiki/Houses> [Accessed 8 May 2020].

Qublar.com. 2020. Handcrafted Set Of Wood Buildings For Our Medival City Building Board Game. Version 01 | Qublar. [online] Available at: <https://www.qublar.com/prototyping/handcrafted-set-of-wood-buildings-for-our-medival-city-building-board-game-version-01/> [Accessed 8 May 2020].

Store.steampowered.com. 2020. Save 50% On ISLANDERS On Steam. [online] Available at: <https://store.steampowered.com/app/1046030/ISLANDERS/> [Accessed 8 May 2020].

Thingiverse.com. 2020. Talisman Board Game 3D Terrain Pieces By Talismods. [online] Available at: <https://www.thingiverse.com/thing:1919056> [Accessed 8 May 2020].

Mines

3DOcean. 2020. Gold Mine Low Poly. [online] Available at: <https://3docean.net/item/gold-mine-low-poly/3783567> [Accessed 8 May 2020].

3DOcean. 2020. Low Poly RTS Human Mine. [online] Available at: <https://3docean.net/item/low-poly-rts-human-mine/4308794> [Accessed 8 May 2020].

2020. [online] Available at: <https://shop.bitgem3d.com/products/low-poly-rts-orc-mine> [Accessed 8 May 2020].

2020. [online] Available at: <https://www.deviantart.com/skshad/art/Medieval-MineCraft-Coal-Mine-404405411> [Accessed 8 May 2020].

Critiques, 3., 2020. Gem Mine-WIP. [online] polycount. Available at: <https://polycount.com/discussion/99460/gem-mine-wip> [Accessed 8 May 2020].

Valdemarribbing.com. 2020. [online] Available at: <https://www.valdemarribbing.com/?lightbox=dataItem-iv88wqnt> [Accessed 8 May 2020].

Gems

2020. [online] Available at: <https://forum.unity.com/threads/volumetric-crystal-materials-pack.507114/> [Accessed 8 May 2020].

2020. [online] Available at: <https://www.youtube.com/watch?v=SoaOHLLL8A0> [Accessed 8 May 2020].

Assetstore.unity.com. 2020. Glass And Crystals Shader | VFX Shaders | Unity Asset Store. [online] Available at: <https://assetstore.unity.com/packages/vfx/shaders/glass-and-crystals-shader-109815> [Accessed 8 May 2020].

Gfycat. 2020. Profane Gem Shader Study GIF By Beta | Gfycat. [online] Available at: <https://gfycat.com/decisivemagnificentatlanticridleyturtle-overpowered-team-profanegame-beta-dev> [Accessed 8 May 2020].

Twitter.com. 2020. Twitter. [online] Available at: <https://twitter.com/benjaminhale7/status/1109719503800131584> [Accessed 8 May 2020].

Units

Adrenaline Brush. 2020. Pawns - Wooden Pawns For Board Games. [online] Available at: <https://www.adrenalinebrush.co.uk/product/pawns/> [Accessed 8 May 2020].

Blog, C. and Pawns, H., 2020. History And Tips About Chess Pawns. [online] Wholesale Chess. Available at: <https://www.wholesalechess.com/blog/history-and-tips-about-chess-pawns/> [Accessed 8 May 2020].

figures, P., Pack, D. and Pack, D., 2020. Buy Disney Pixar Toy Story 4 Little People 7 Figure Pack | Playsets And Figures | Argos. [online] Argos. Available at: <https://www.argos.co.uk/product/9349016> [Accessed 8 May 2020].

model, C., model, H. and model, B., 2020. Board Game Pawns - Low Poly PBR 3D Model. [online] Sellfy.com. Available at: <https://sellfy.com/p/5l5b2a/> [Accessed 8 May 2020].

Pinterest. 2020. Candy Land Movers/Tokens Replacement Parts ~ Cake Topper Lot Of 2 #Miltonbradley | Cake Toppers, Topper. [online] Available at: <https://www.pinterest.co.uk/pin/661888476460824710/?lp=true> [Accessed 8 May 2020].

The Game Crafter News. 2020. New Board Game Pieces - Tall Thick People. [online] Available at: <https://news.thegamecrafter.com/post/183243865299/new-board-game-pieces-tall-thick-people> [Accessed 8 May 2020].

Helmets and Swords

2020. [online] Available at: <https://pixabay.com/illustrations/knight-lowpoly-3d-armor-sword-4070501/> [Accessed 8 May 2020].

2020. [online] Available at: <https://www.youtube.com/watch?v=CcAXuxA5e3Q> [Accessed 8 May 2020].

Deviantart.com. 2020. Low Poly Helmet By Deuxichthys On Deviantart. [online] Available at: <https://www.deviantart.com/deuxichthys/art/Low-poly-Helmet-566045015> [Accessed 8 May 2020].

Knight, P., 2020. Download Low Poly Knight Transparent PNG On Yellow Images 360°. [online] Yellowimages.com. Available at: <https://yellowimages.com/images-360/products/low-poly-knight-yi3602401?ca=1_16> [Accessed 8 May 2020].

Models, V., model, D. and model, D., 2020. Detailed Low-Poly Helmet With Horns | 3D Model. [online] CGTrader. Available at: <https://www.cgtrader.com/3d-models/military/armor/very-detailed-low-poly-helmet-with-horns> [Accessed 8 May 2020].

OpenGameArt.org. 2020. Low Poly Swords Pack. [online] Available at: <https://opengameart.org/content/low-poly-swords-pack-0> [Accessed 8 May 2020]

.Environmental Features

2020. [online] Available at: <https://www.artstation.com/artwork/xk3eE> [Accessed 8 May 2020].

Behance.net. 2020. Behance. [online] Available at: <https://www.behance.net/gallery/46778871/Low-poly-(stylisedcartoon)-3d-asset-pack> [Accessed 8 May 2020].

Jay Burns. 2020. Stylized Fantasy Environment — Jay Burns. [online] Available at: <https://www.jayburns3d.com/new-project> [Accessed 8 May 2020].

Models, 3., models, V., model, H. and model, H., 2020. Handpainted Stylized Environment - Flora Pack | 3D Model. [online] CGTrader. Available at: <https://www.cgtrader.com/3d-models/various/various-models/handpainted-stylized-environment-flora-pack> [Accessed 8 May 2020].

Robertson, A., 2020. 30 Amazing Board Games Not To Miss In 2019. [online] Forbes. Available at: <https://www.forbes.com/sites/andyrobertson/2019/02/21/30-amazing-board-games-not-to-miss-in-2019/#508ddb9467fc> [Accessed 8 May 2020].

Twitter.com. 2020. Twitter. [online] Available at: <https://twitter.com/MaxNielsenDev/status/1226534821977088000/photo/1> [Accessed 8 May 2020].

Unrealengine.com. 2020. Low Poly Stylized Environment By Emek Ozben In Environments - UE4 Marketplace. [online] Available at: <https://www.unrealengine.com/marketplace/en-US/product/low-poly-stylized-environment> [Accessed 8 May 2020].

Appendix 5: User Story

This is a user story created during the research and planning stage of the project. This was used to inform the larger feature list, however not everything on this user story was implemented due to time restraints.

Appendix 6: Functionality Requirements

The functionality requirements list is an organised list of required features in the project, and when they will be developed upon and refined. This is organised per phase (Group of 2 or 3 week-long sprints) but does not contain definitions as to what feature needs to be done within what week. Due to the very nature of game development, especially within the space of VR, it's important to not have a hugely detailed feature list. Compared to usual software development, game development has a much larger requirement of user experience, which is hard to write a detailed plan for before any prototyping takes place.

Mechanic and Input Prototyping

VR Inputs

- Basic VR Viewing
- Scrolling Controls (Replicating GORN movement)
- Point at a hex and it highlights it.
- Click when pointing at the hex and it returns the hex.
- Point and click at a block in your left hand and it allows for UI. (For later implementation as a board game manual)
- Motion Sickness considerations (VR research into preventing motion sickness)

Terrain

- Basic Hex Terrain
- Hex forms a square
- Gem in the centre of the terrain
- Each hex acts as a node - Can be non-navigable or navigable.
- Water shader (No vertex displacement)
- Plastic-esque green shader, for use as grass.

Units

- Melee units can be spawned
- Units are scriptable objects with health, damage, movement speed, time perception and colour gradient to spawn within.
- Units have knowledge of the nodes and prototype movement has been implemented

Assets

- Hex with rounded edges
- Basic white player that will be recoloured by its colour gradient.
- White block Mine (Fits within a hex)
- White block Melee Barracks (Fits within a hex)
- White block Gem.

Buildings

- Place both buildings with a point and click.
- Mine's generate Money (VR research into displaying UI)

Gameplay Prototyping

Units

- Enemies appear at the edge of the map before the round starts
- Enemies move towards the centre gem (Move every 5 seconds?).
- Basic Combat System [Units move between nodes to reach a goal, encountering an enemy starts a battle, 1 Hit = 1 Health lost on a set timer]
- When Units die, they fall over and stay in position.

Assets

- MagicaVoxel Prototype Mine Building
- MagicaVoxel Prototype Barracks Building
- White block Defensive Walls
- Grass and Rocks randomly placed poking out of the floor.
- Hands - 3rd party asset is fine.

Audio

- Basic Combat Sounds (Swords, yelling)
- Basic Building Sounds

Gameplay

- Announce building and attack phase (2 mins long by default)
- Book in hand while in building phase
- Choice wheel in attack phase

Buildings

- Place Defensive Walls
- Placing buildings costs money
- Melee Barracks spawns a melee unit on a set timer.

Player Interaction and User Interface

- See a timer of the time left in the current phase

MVP Phase

Input

- Any refinement to control systems that wasn't done during original development.

Lighting

- Some consideration of lighting should be made during this task
- Revisit resolution of shadows

Units

- Enemies reaching the gem is game over
- Check various combinations of buildings, if navigation is not possible,
- Add Juice to Unit movement. (Hopping around as if moved by hands)

Combat Refinement

- Spawned player units attack incoming enemies automatically
- Enemies can also attack buildings
- Ensure efficient movement of enemies towards the correct goal. (Node based Pathfinding research opportunity)
- Enemies must destroy the gem (It has health)

Audio

- Enhance combat audio
- Enhance Building audio
- Spawning of enemies
- UI Interactions

Assets

- Refined Melee Units (Hats, weapons etc.)

Buildings

- Timers of mines and barracks is displayed around the edge of the hex (Subtle green highlight that fills around the edge)
- Health of buildings is displayed with the same outer highlight of the hex OR investigate shaders for deconstructing buildings

Player Interaction and User Interface

- Game Menu - Board game themed that you must open and take options out?
- Scroll Wheel implemented during attack phase allows for zapping enemies, slowing down and speeding up units (Change timePerception value of that unit for a set amount of time)

Usability Preparation

- Ensure full gameplay loop is functional
- Improve viveport listing
- Embed automatic data tracking for gameplay stats

Usability Response

Balancing

- Using feedback given, adjust appropriate values to ensure some basic form of balancing has happened.

MAP Phase

Terrain

- Refine hex shading + overall style. [Message reference image artists to see if they have any tips.

Player Interaction and User Interface

- UI Refinement (Working settings, Improved button functionality etc.)
- Building Upgrades

Assets

- Refined Melee Barracks
- Refined Mines

Units

- Unit movement refinement

Poster

- Make Project Poster
- Higher priority than other MAP tasks, so moving them into DocandSub phase may be essential

Bug Fixing and User Experience Refinement

- This task is open to additions throughout development and usability testing.

Appendix 7: Sprint Conclusion Checklist

A checklist of tasks that were completed at the end of each week of development.

Functionality Tested
All added code + functionality should be functionality tested and the document filled out.

Piazza Post
After the coronavirus pandemic, a weekly piazza post became a requirement to keep by supervisor up to date with my work

Devlog
A weekly devlog should be written. This included recording videos and uploading them to YouTube for use in this

Coding Standards
Time specifically set aside to check that the code written during that week was up to standard and commented adequately.

Upload Trello Screenshot to DLE
A module requirement

Automated Tweets setup
Setup tweets to run throughout the week, so that I do not have to think about them whilst I'm developing

Next Sprint Tasks Created
Create the tasks in trello ready for the next sprint of work

Complete Weekly Sprint Reports
Complete the weekly Sprint reports, including adding any task times that may have been missed

Update Roadmap Checklists
Update the checklists on the trello roadmap to ensure they're up to date with the progress

Update Final Report Weekly Breakdown
Update the weekly breakdown appendices on my final report with all the reports generated during the week.

Merge Staging into master
Merge the staging branch into the master branch when everything was tested and committed

Sprint Archived into High Ground Archive Board
Archive the sprint into the "High Ground Archive Board" on trello, under the correct columns.

Appendix 8: Blog Post Links

Week 1 - Input and Project Preparation (20th Jan 2020 - 24th Jan 2020)
https://jackgriffithsdev.tumblr.com/post/190441513833/high-ground-devlog-1-vr-inputs

Week 2 - Input Refinement and Basic Gameplay Elements (27th Jan 2020 - 31st Jan 2020)
https://jackgriffithsdev.tumblr.com/post/190557037363/high-ground-devlog-2-game-board-input

Week 3 - Core Gameplay Mechanics (3rd Feb 2020 - 7th Feb 2020)
https://jackgriffithsdev.tumblr.com/post/190717439698/high-ground-devlog-3-core-gameplay-mechanics

Week 4 - Combat and Board Generation (10th Feb 2020 - 14th Feb 2020)
https://jackgriffithsdev.tumblr.com/post/190867144953/high-ground-devlog-4-gameplay-development

Week 5 - Gameplay Development and Asset Prototyping (17th Feb 2020 - 21st Feb 2020)
https://jackgriffithsdev.tumblr.com/post/190990792378/high-ground-devlog-5-prototype-assets

Week 6 - Pathfinding Refinement and Movement Development (23rd Feb 2020 - 28th Feb 2020)
https://jackgriffithsdev.tumblr.com/post/611493456956768256/high-ground-devlog-6-aggressive-enemies

Week 7 - Menu and Final Mechanics (2nd Mar 2020 - 6th Mar 2020)
https://jackgriffithsdev.tumblr.com/post/612075685077516288/high-ground-devlog-7-spell-attacks

Week 8 - Asset Refinement + Smarter Enemies (9th Mar 2020 - 16th Mar 2020)
https://jackgriffithsdev.tumblr.com/post/612760266698031104/high-ground-devlog-8-game-refinement

Week 9 - MVP finalization (17th Mar 2020 - 23rd Mar 2020)
https://jackgriffithsdev.tumblr.com/post/613376456067776512/high-ground-devlog-9-ui-improvements

Week 10 - Preparing BETA (23rd Mar 2020 - 27th Mar 2020)
https://jackgriffithsdev.tumblr.com/post/614108222167695360/high-ground-devlog-10-usability-preparation

Week 11 - Usability Response (30th Mar 2020 - 3rd Apr 2020)
https://jackgriffithsdev.tumblr.com/post/614649619486588928/high-ground-devlog-11-overall-stylistic-updates

Week 12 - Bug Fixes and Refinement (6th Apr 2020 - 10th Apr 2020)
https://jackgriffithsdev.tumblr.com/post/615385683834634240/high-ground-devlog-12-pathfinding-overhaul

Week 13 - Further Bug Fixes and Refinement (13th Apr 2020 - 17th Apr 2020)
https://jackgriffithsdev.tumblr.com/post/615932686061780992/high-ground-devlog-13-wrapping-things-up

Week 14 - Final Additions (20th Apr 2020 - 24th Apr 2020)
https://jackgriffithsdev.tumblr.com/post/616402720618954752/high-ground-devlog-14-final-additions

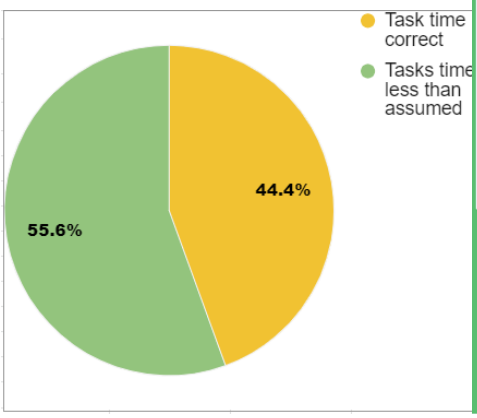
Appendix 9: Weekly Breakdowns + Sprint Analysis

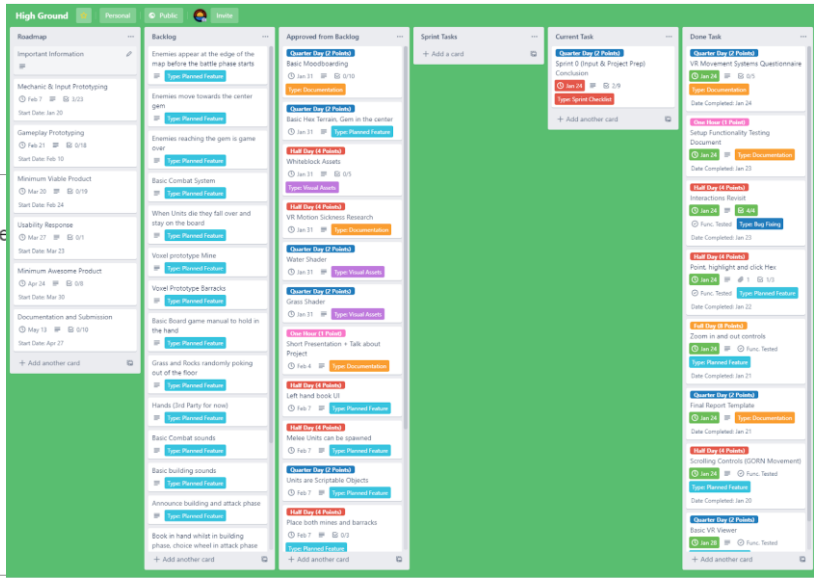
Week 1 - Input and Project Preparation (20th Jan 2020 - 24th Jan 2020)

	Mechanic & Input Prototyping		20th Jan 2020 - 24th Jan 2020	Supervisor Meeting Date : N/A		
	Sprint 1 - Input and Project Preperation		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
1	Functionality Testing Success Rate	80.00%	Basic VR Viewing	Planned Feature	2	1	1
	Devlog Link	https://ja	Scrolling Controls (Replicating GORN movement)	Planned Feature	4	2.5	1.5
	Coding Standards Complete	☑	Zoom in and out through pinch and grab	Planned Feature	8	5	3
	Trello Screenshot Upload	☑	Point at a hex and click when pointing at the hex and it returns t	Planned Feature	4	3.5	0.5
	Next Sprint Tasks	☑	Final Report Template	Documentation	2	2	0
	Roadmap Updated	☑	VR Movement Systems Questionaire	Documentation	2	2	0
	Branches Merged	☑	Movement Systems Revisit	Bug Fixing	4	4	0
			Setup Functionality Testing	Documentation	1	0.5	0.5
	Task time more than assumed	0					
	Task time correct	4					
	Tasks time less than assumed	5					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	2	2	0

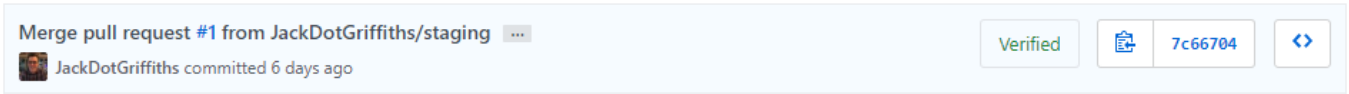
	Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successfull	Notes
1	Basic VR Viewer	Feature Implementation	Open and run the project in VR	Headset tracks accordingly and player can see ever	As expected	☑	
	Zoom in and out	Feature Implementation	Hold triggers while moving hands apart and together	Game board scales up and down based on player in	As expected	☑	
	Point and Click at Hex	Feature Implementation	Point and trigger hex	Pointing highlights the hex and clicking returns the n	Functionality works, however there are issues where	☐	Task added to next sprint to fix.
	Interactions Revist (Teleport onto board)	Feature Implementation	Point onto board and press thumbwheel in	Player can teleport on and off the board using the tra	As expected	☑	
	Player can spin the board around.	Feature Implementation	Hold one trigger and move it to turn the boad.	Board rotates appropriately.	As expected	☑	Not as responsive as desired, add

Sprint Complete	24/01/2020
Prototyping	0
Feature Implementations	5
Design Changes	0
Bug Fixes	0
Success Count	4
Total Tasks	5
Percentage Success	80.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Commits on Jan 24, 2020



Notes

- 10 commits during this sprint
- The intended control method didn't work out as expected, so the task "Movement Systems Revisit" changed the functionality of the movement system within the game.
- All tasks completed on time
- Sorted out most of my documentation
- Filled out my roadmap on trello
- Prototyped most VR interactions

- Changed the VR interactions to be instant teleport rather than zoom in. Much more intuitive.

Success Analysis

During this sprint, two different movement systems were prototyped. The test-driven development implemented during the entirety of this project, allowed both to be refined and tested to determine which was most appropriate. Out of the two, pointing and teleporting was the most appropriate. This is due to it already being a well-established virtual reality movement system, hence it is the most accessible to existing players. The game board generation successfully places hexagons in the desired location and works to any desired length and width. These are currently stored as a list of Game Objects within the Gameboard Generation component.

All the implemented features were functionality tested at the end of this week, to an 80% success rate. Issues were spotted with removing the selection highlight once a player had pointed away from the hexagon, which has been added as a task to fix in the coming sprint. All the SMART objectives were met; hence the project has a great basis to move onto the next sprint in which the basic gameplay elements are to be prototyped.

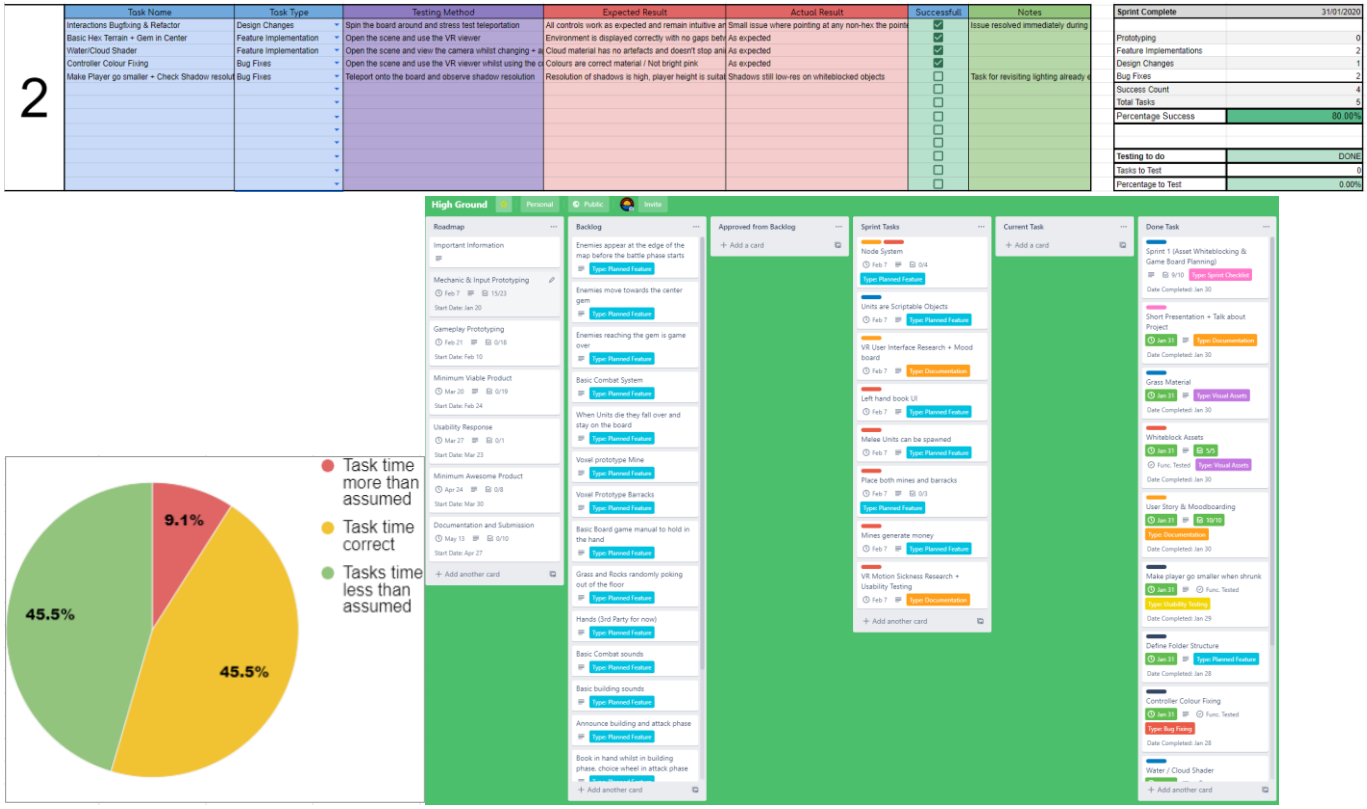
Further Refinement

The input system will continue to be tweaked and improved throughout development, as a variety of User Interfaces are prototyped, and new interactions are added. After the usability testing session at the end of this phase, the input and interactions will be reviewed as to their success, and necessary changes made based on feedback.

The game board generation is currently just an algorithm that instantiates game objects in a position. In sprint 3, the node system will be fully implemented. This stores the navigational data, such as coordinates and contents of a node, ready for pathfinding.

Week 2 - Input Refinement and Basic Gameplay Elements (27th Jan 2020 - 31st Jan 2020)

2	**Mechanic & Input Prototyping**		27th Jan 2020 - 31st Jan 2020	Supervisor Meeting Date : N/A			
	Sprint 2 - Input Refinement and Basic Gameplay Elem		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	80.00%	Interactions Bugfixing & Refactor	Bug Fixing	4	3.5	0.5
	Devlog Link	https://ja	User Story & Moodboarding	Documentation	8	4	4
	Coding Standards Complete	☑	Basic Hex Terrain + Gem in Center	Planned Feature	2	2	0
	Trello Screenshot Upload	☑	Whiteblock Assets	Visual Assets	4	2	2
	Next Sprint Tasks	☑	Water/Cloud Shader	Visual Assets	2	3	-1
	Roadmap Updated	☑	Grass Material	Visual Assets	2	1	1
	Branches Merged	☑	Define Folder Structure	Documentation	0.5	0.5	0
			Controller Colour Fixing	Bug Fixing	0.5	0.5	0
	Task time more than assumed	1	Make Player go smaller + fix low res shadow rendering	Bug Fixing	0.5	0.5	0
	Task time correct	5	Presentation	Documentation	1	1	0
	Tasks time less than assumed	5					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	2	1.5	0.5



Commits on Jan 30, 2020

Merge pull request #2 from JackDotGriffiths/staging ...

JackDotGriffiths committed 10 minutes ago

Verified 0d0b33c

Notes

- 15 commits during this sprint
- Provide a screenshot of the cloud shader graph
- Lots of refinement of interactions made during this sprint
- Skybox added a cloud shader that works using the lightweight render pipeline and shader graph to add varying noise and vertex displacement to a flat plane with 25k equally triangulated vertices. 100k vertices was tried but lagged in vr
- Quite a lot of purely cosmetic tasks that took a short amount of time but contributed to the overall social media presence during the phase.
- Noticed lots of low-res shadows during this phase so changed it from the player shrinking to the environment growing. This problem still exists but is under review
- Hex shape recreated with a slightly bevelled edge. This is a much more optimised version of Hex#2, which had 20 bevel layers. In hex#3 there are only 6 bevel layers.

Success Analysis

Everything added during this sprint was functionality tested, to an 80% success rate. There was an issue where the new assets would not cast shadows on the game environment, which is due to issues in the settings of the newly imported Universal Render Pipeline. All the

smart objectives and features planned for this week were fully implemented, and the white blocked assets are appropriate representations for the upcoming sprints and usability testing.

Further Refinement

Further refinement of the assets created within sprint comes in Sprint 5, when MagicaVoxel will be used to create more accurate representations of each building. The final assets will be made in Sprint 8. The reason for staggering the development of these assets is to spread out their creation, so not too long is spent on visual assets in any one week. The visual research will continue to be referenced throughout the project, however, does not require any further work.

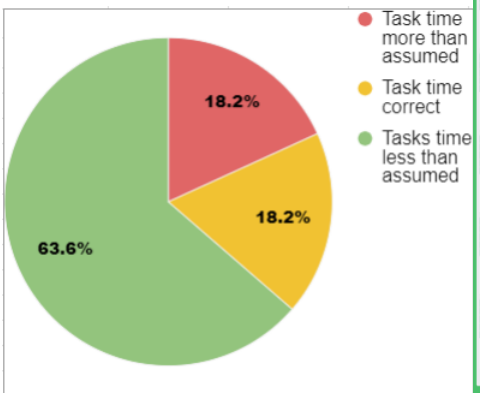
Week 3 - Core Gameplay Mechanics (3rd Feb 2020 - 7th Feb 2020)

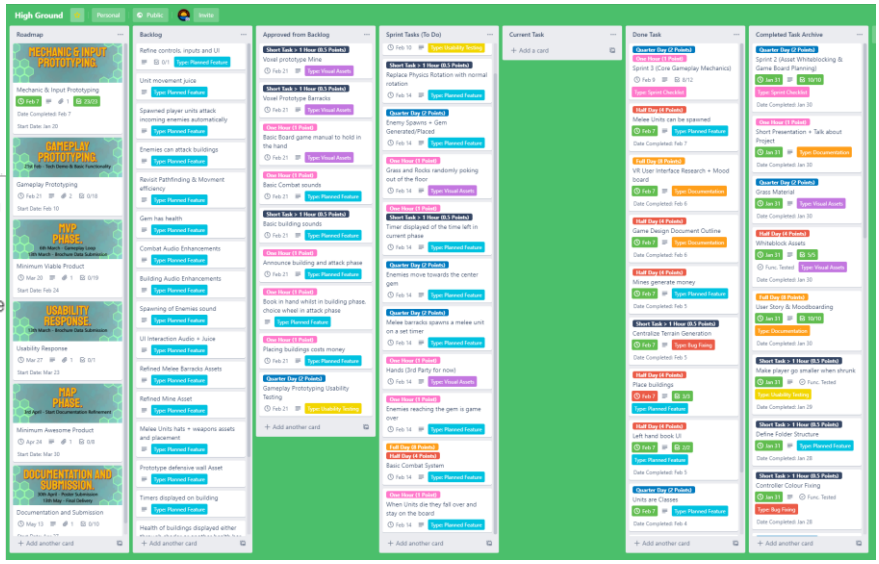
3	Mechanic & Input Prototyping		3rd Feb 2020 - 7th Feb 2020	Supervisor Meeting Date : 04/02/2020 11:30		
	Sprint 3 - Core Gameplay Mechanics		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	100.00%	Node System	Planned Feature	12	13	-1
	Devlog Link	https://ja	Units are Classes	Planned Feature	2	0.5	1.5
	Coding Standards Complete	✓	VR User Interface Research + Mood board	Documentation	8	2	6
	Trello Screenshot Upload	✓	Left hand book UI	Planned Feature	4	3.5	0.5
	Next Sprint Tasks	✓	Melee Units can be spawned	Planned Feature	4	4	0
	Roadmap Updated	✓	Place buildings	Planned Feature	4	2	2
	Branches Merged	✓	Mines generate money	Planned Feature	4	1	3
			VR Motion Sickness reasearch	Documentation	2	1	1
	Task time more than assumed	2	Centralize Terrain Generation	Bug Fixing	0.5	0.5	0
	Task time correct	2	Game Design Document Outlined	Documentation	4	3	1
	Tasks time less than assumed	7					
	Sprint Complete	✓	Sprint Conclusion Checklist	Sprint Checklist	3	4	-1

3

Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successful	Notes
Node System + Pathfinding	Feature Implementation	Use the Pathfinding Development scene and run the auto	Viewing paths generated that are very efficient, with	As expected	✓	
Units are Classes	Feature Implementation	Breakpoint after the Unit constructure and check values	For each unit spawned by the barracks, the property	As expected	✓	Small issue immediately resolved
Left Hand Book UI	Feature Implementation	Check Left & Right hand placement and selection of all bu	Buttons work as expected for both left and right hand	As expected	✓	
Building Placement	Feature Implementation	Try placing all building options in various combinations on	Correct buildings are spawned in the correct node. 2	As expected	✓	
Centralized Terrain Generation	Feature Implementation	Run the terrain generation and check positions during test	Game board can be generated and is centralised in t	As expected	✓	
Mines Generate Money	Feature Implementation	Place a mine on the board and check the DebugWall to se	One mine should produce a certain amount, two min	As expected	✓	
Melee Units can be spawned	Feature Implementation	Place a barracks on the board, and use a breakpoint to in	Barracks are placed and units are placed in the corr	As expected	✓	
					☐	
					☐	
					☐	
					☐	
					☐	

Sprint Complete	07/02/2020
Prototyping	0
Feature Implementations	7
Design Changes	0
Bug Fixes	0
Success Count	7
Total Tasks	7
Percentage Success	100.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- A lot of time spent on the Node system and pathfinding this week, so much so that I spent the weekend before the sprint planning and researching the process.
- Scriptable objects were going to be used for a unit, however using classes seemed much more appropriate.
- A lot of frameworks are built behind placing buildings, leaving room for verifying a buildings location before placing it.
- GDD planned out during this phase, as well as a lot of art styles and other documentation.
- Whilst testing, I noticed lagging issues when the board is rotating with all the spaces full. This could potentially become a bigger issue when geometry is not this simplistic, so the physics board rotation will be replaced with transform rotation which is much less intensive.

Success Analysis

All the SMART objectives were completed during this week, and the functionality testing came to a 100% success rate. The node class holds data on each hexagon as expected, and the graph is stored correctly in the ‘Gameboard Generation’ singleton. The pathfinding functions as required for this level of the project, and all other planned interactions are prototyped successfully.

Further Refinement

The Node System is completely implemented as intended and doesn’t require any further refinement. The pathfinding will be revisited as in Sprint 6, where aggressive enemies will be implemented and weighted node A* pathfinding implemented. The User Interface will continuously be tweaked and updated, with the major updates happening at the start of ‘MVP Phase’ in sprint 6 and regular updates as time goes on. Changes to all these mechanics will be introduced as usability testing goes on also.

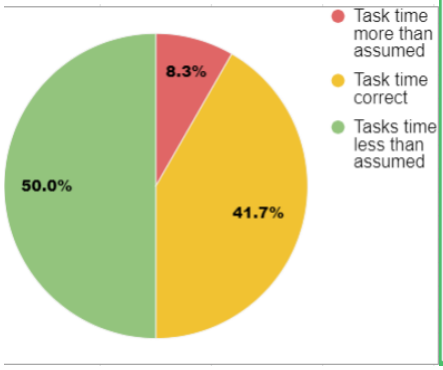
Week 4 - Combat and Board Generation (10th Feb 2020 - 14th Feb 2020)

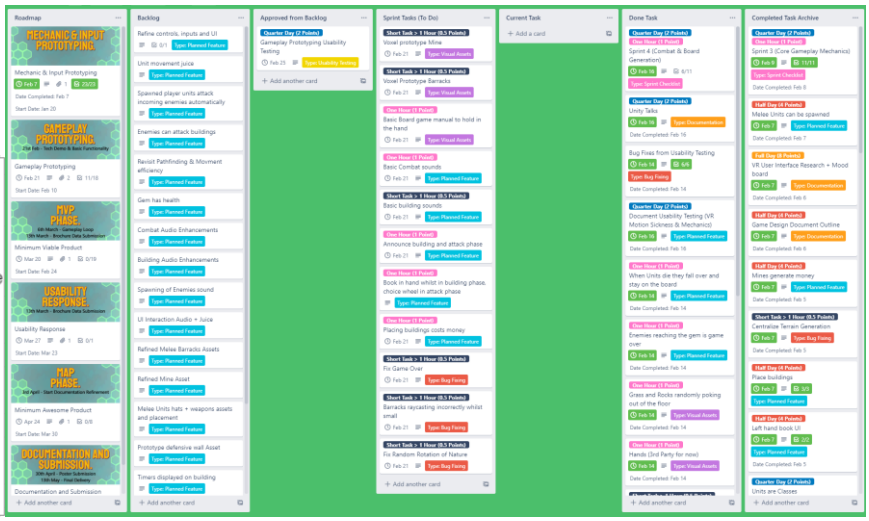
4	Gameplay Prototyping		10th Feb 2020 - 14th Feb 2020	Supervisor Meeting Date : 11/02/2020 11:30			
	Sprint 4 - Combat & Board Generation		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	85.71%	VR Motion Sickness Research + Usability Testing Documentatio	Usability Testing	4	3	1
	Devlog Link	https://ja	Replace Physics Rotation with Normal Rotation	Planned Feature	0.5	0.5	0
	Coding Standards Complete	☑	Enemy Spawns + Gem Generated/Placed	Planned Feature	2	1	1
	Trello Screenshot Upload	☑	Grass and Rocks randomly placed on the board	Visual Assets	1	1	0
	Next Sprint Tasks	☑	Timer displayed of the time left in the phase	Planned Feature	1.5	0.5	1
	Roadmap Updated	☑	Enemies move towards center gem	Planned Feature	2	4	-2
	Branches Merged	☑	Melee barracks spawn a melee unit on a timer	Planned Feature	2	1	1
			Hands (3rd party)	Visual Assets	1	0.5	0.5
	Task time more than assumed	1	Enemies reaching the gem is game over	Planned Feature	1	1	0
	Task time correct	6	Basic Combat System	Planned Feature	12	6	6
	Tasks time less than assumed	6	When units die they fall over and stay on the board	Planned Feature	1	1	0
			Watch Unity Talks on Scriptable Objects	Documentation	2	2	0
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3	3	0

4

Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successfull	Notes
Enemy Spawns + Gem Placed Automatically	Feature Implementation	Generate varying sizes of the environment and start the g	Gem is always centralized and enemy spawns are a	As expected	☑	
Timer on current time left implemented	Feature Implementation	Watch the timer go down. Change the time and make sure	Timer goes down and updates correctly	As expected	☑	
Enemies spawn from spawn locations	Feature Implementation	Start the project and watch the spawns after the setup tim	Enemies will spawn, the amount of enemies spawn a	As expected	☑	
Enemies move towards the gem	Feature Implementation	Watch the enemies. Debug Drawline their path	Enemies will spawn and then move towards the gem	As expected	☑	
Basic combat	Feature Implementation	Create specific examples where combat is forced	Enemies that encounter and adjacent friendly units a	As expected	☑	
Melee barracks spawn a melee unit on a set time	Feature Implementation	Once combat has occured, check the friendly unit respaw	Unit spawns according to the set time. Cannot spaw	As expected	☑	
Grass and Rocks randomly poking out of the floo	Feature Implementation	Turn on the ambient nature in the editor and press play or	Random grass and rocks will be placed as per the de	As expected	☑	Rotation isn't random, task added t
Enemies reaching the gem is game over	Feature Implementation	Allow for the game to play out, ensure game speed is set	Game over displayed on the screen and everything s	Only checks if the game is over when the round is ov	☐	Very quick fix, task added to trello.
When units die they fall over and stay on the boa	Feature Implementation	Create specific examples where combat is forced, observ	During combat, enemies and players will fall over on	As expected	☑	
Player can build on enemy spawn adjacent node	Bug Fixes	Play the game and attempt to build on nodes adjacent to	Player is unable to place buildings	As expected	☑	
Fix building placement whilst small	Bug Fixes	Teleport onto the board and attempt to build all types of bu	Buildings spawn, scaled and positioned correctly.	Buildings place correctly, barracks raycasts onto the	☐	Task added to trello
Player can teleport onto occupied nodes	Bug Fixes	Attempt to teleport onto nodes with buildings on	Player is unable to teleport onto targeted node	As expected	☑	
Enemies can stack on top of eachother whilst mo	Bug Fixes	Allow the simulation to play out for a while, watching enem	Enemies do not stack on top of eachother, and they	As expected	☑	
Enemies teleport in from 0,0,0, not the correct lo	Bug Fixes	Play the game and watch enemies spawn, ensure it's fron	Enemies spawn on nodes adjacent to enemy spawn	As expected	☑	

Sprint Complete	14/02/2020
Prototyping	0
Feature Implementations	9
Design Changes	0
Bug Fixes	5
Success Count	12
Total Tasks	14
Percentage Success	85.71%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- Spent a lot of time this week implementing movement of enemies, as the pathfinding needed to be adapted to work for an actual unit.
- Physics based rotation was too intensive for the possibility of lots of game pieces on the board, so an alternative was tried not using physics, but it didn't feel intuitive enough to warrant having it as a feature. If the player wishes to rotate the board, they should just walk around it.
- Hand rotated so it looks like you're holding the book.
- Nature assets I've created for another project used temporarily for this sprint, just as placeholders.

Success Analysis

This was a sprint of significant additions, which were not expected to be perfectly implemented during this week. Despite this, all the major updates made during this week were fully functioning when tested, and any issues were with smaller additions. The functionality

testing came to a success rate of 87.71%, with issues occurring when an enemy reached a gem, and the grass and rocks placed on the board not having random rotation when instantiated.

Further Refinement

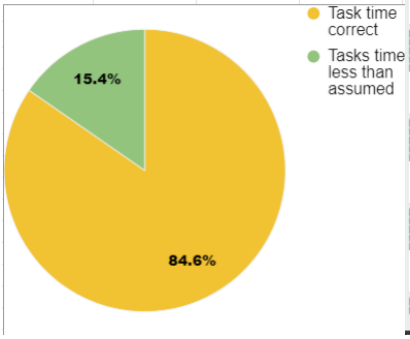
The core features added during this sprint will be refined incrementally throughout the development of this project. In the next sprint, a minor form of balancing will take place so that the enemy and friendly units are adequately matched in combat. Units will be given more animations, so they look like they’re attacking each other during a battle. This will occur in sprint 6. The next large addition to the combat system will occur in sprint 7, when player to unit interactions are implemented. This will include spells that allow the player to slow down or speed up units, hence that needs to be reflected in battle.

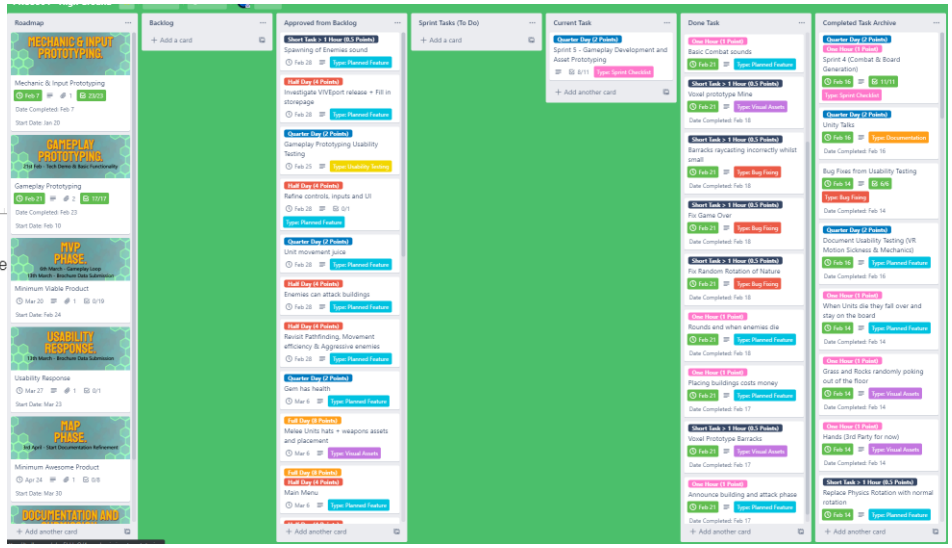
Week 5 - Gameplay Development and Asset Prototyping (17th Feb 2020 - 21st Feb 2020)

5	Gameplay Prototyping		17th Feb 2020 - 21st Feb 2020	Supervisor Meeting Date : 18/02/2020 11:30			
	Sprint 5 - Gameplay Development and Asset Prototypi		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	100.00%	Voxel Prototype Mine	Visual Assets	0.5	0.5	0
	Devlog Link	https://ja	Voxel Prototype Barracks	Visual Assets	0.5	0.5	0
	Coding Standards Complete	☑	Player scaled down not environment scaled up.	Bug Fixing	1	0.5	0.5
	Trello Screenshot Upload	☑	Basic Combat sounds	Planned Feature	1	1	0
	Next Sprint Tasks	☑	Basic Buildings sounds	Planned Feature	0.5	0.5	0
	Roadmap Updated	☑	Announce building & Attack phase	Planned Feature	1	1	0
	Branches Merged	☑	Book in hand whilst in build phase, shows interactions in attack	Planned Feature	1	1	0
			Placing buildings costs money	Planned Feature	1	0.5	0.5
	Task time more than assumed	0	Fix Game Over	Bug Fixing	0.5	0.5	0
	Task time correct	11	Barracks raycasting incorrectly whilst small	Bug Fixing	0.5	0.5	0
	Tasks time less than assumed	2	Fix random rotation of nature	Bug Fixing	0.5	0.5	0
			Round ends when all enemies die	Planned Feature	1	1	0
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3	3	0

5	Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successfull	Notes
	Announce building and attack phase	Feature Implementation	Run through a gameplay session and ensure rounds are	Phases are shown on the debug board correctly, and	As expected	☑	
	Placing Buildings costs money	Feature Implementation	Place buildings and ensure correct cost of building is take	Correct building cost is taken away from the players	As expected	☑	
	Rounds end when all enemies die	Feature Implementation	Play through a round and check that it ends and building f	Round ends once all enemies have been defeated	As expected	☑	
	Fixed random rotation of nature	Bug Fixes	Run the environment generation on a variety of map sizes	All ambient nature elements have random rotation	As expected	☑	
	Barracks raycasting incorrectly whilst small	Bug Fixes	Teleport onto the game board and place a barracks	The enemies that are spawned by a barracks are pla	As expected	☑	
	Basic Combat Sounds	Feature Implementation	Play through a level and listen to the audio when an enem	Random combat sound plays as expected	As expected	☑	
	Basic Building Sounds	Feature Implementation	Play through a level and place a few buildings to check th	Random building sound plays as expected	As expected	☑	
	Book in hand whilst in building phase, shows atta	Feature Implementation	Play through a few rounds and check the book changes a	Player is shown attacks whilst in the action phase	As expected	☑	
	Player scaled down not environment scaled up	Bug Fixes	Teleport onto the game board and check that the enemies	In both small and large sizes, enemy movement rem	As expected	☑	
						☐	
						☐	

Sprint Complete	21/02/2020
Prototyping	0
Feature Implementations	6
Design Changes	0
Bug Fixes	3
Success Count	9
Total Tasks	9
Percentage Success	100.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- I'm going to a wedding this week, so I only have Monday and Tuesday available for development. That means tasks on this list are very lacking, and extra time will be spent doing documentation whilst travelling.
- Building validation builds upon a bit during this, fixing a few issues and making gameplay more complex through restrictions on building placement.
- The scroll wheel was removed as an idea as it would make much more sense to keep with one input method, making the project more accessible.
- Voxel models made are more developed, giving a better idea of the final product.
- Issue fixed where barracks wouldn't regenerate more than one unit.
- Audio Manager implemented, and appropriate methods written for playing sounds from specific locations.
- The wedding this week has resulted in me completing the phase at the end of the week on Sunday, instead of on the Friday. I knew I would have to pick up work over the weekend, so this was not an unplanned issue.

Success Analysis

The additions made during this sprint were implemented to a 100% success rate, as everything added was fully functional when tested. The game loop works as expected; however, usability testing may reveal unseen issues. This sprint marks the end of the prototyping of this project, and it’s currently at a very good place feature wise to develop the prototypes of features even further.

Further Refinement

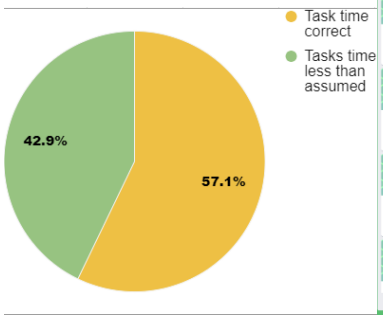
The difficulty of enemies will be tweaked after the usability testing at the end of this sprint, and more advanced difficulty related additions will occur in the MAP phase, in sprint 14. This includes incrementing the size of an enemy group, so it spawns as a group of 2 initially and then the groups get larger as the rounds go on. The building placement rules are fully implemented; however, issues may arise when usability testing that require fixing later in the project. For this reason, a task has been added to sprint 11 to review and optimise the script. The only additions to the phase announcement in the future will be a form of audio announcement, and the ability to skip the build phase, occurring in sprint 12 and 13 respectively.
The final building asset refinement will happen in sprint 8, where their representations will be reviewed, and any changes made in Blender.

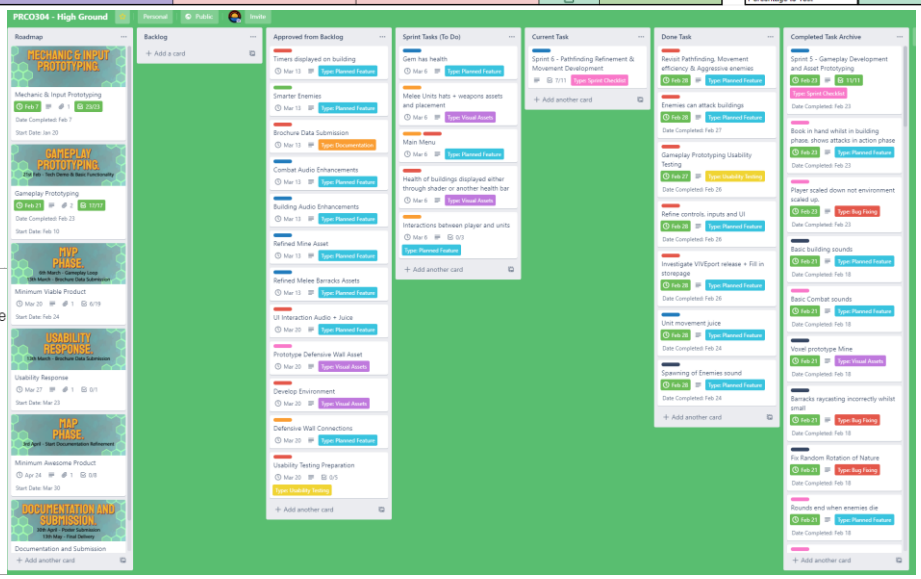
Week 6 - Pathfinding Refinement and Movement Development (23rd Feb 2020 - 28th Feb 2020)

	Minimum Viable Product		24th Feb 2020 - 28th Feb 2020	Supervisor Meeting Date : 25/02/2020 11:00		
	Sprint 6 - Pathfinding Refinement & Movement Develo		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
6	Functionality Testing Success Rate	100.00%	Spawning of Enemies sound	Planned Feature	0.5	0.5	0
	Devlog Link	https://ja	Investigate VIVEport release + Fill in parts	Planned Feature	4	3	1
	Coding Standards Complete	☑	Gameplay Prototyping usability testing	Usability Testing	4	4	0
	Trello Screenshot Upload	☑	Refine controls, inputs and UI	Planned Feature	4	2	2
	Next Sprint Tasks	☑	Unit movement juice	Planned Feature	2	2	0
	Roadmap Updated	☑	Enemies can attack buildings	Planned Feature	4	4	0
	Branches Merged	☑	Revisit pathfinding, movement efficiency & Aggressive enemies	Planned Feature	4	3	1
	Task time more than assumed	0					
	Task time correct	5					
	Tasks time less than assumed	3					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3	3	0

	Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successful	Notes
6	Spawning of enemies sound	Feature Implementation	Listen to the enemies as they spawn	Sound plays when an enemy spawns	As expected	☑	Max distance needed to be increas
	Refine controls, inputs and UI	Feature Implementation	Play a level and examine all UI interactions	All UI interactions act as expected and are intuitive	As expected	☑	
	Unit movement juice	Feature Implementation	Play a level and examine the Units movement	Units jump higher and can rotate towards the mover	As expected	☑	
	Enemies can attack buildings	Feature Implementation	Block in an enemy spawn with walls and check their react	Enemy destroys the wall, or navigates around it if the	As expected	☑	
	Revisit pathfinding, movement efficiency & Aggr	Feature Implementation	Attempt to block in enemies in a variety of circumstances	Trapped enemies become more aggressive, and agg	As expected	☑	
						☐	
						☐	
						☐	
						☐	
						☐	
						☐	

Sprint Complete	28/02/2020
Prototyping	0
Feature Implementations	5
Design Changes	0
Bug Fixes	0
Success Count	5
Total Tasks	5
Percentage Success	100.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- Lots of tweaks/changes made to the balancing and timing of certain events within the project ready for usability testing. These were tested within a small group before showing to other course mates. I had a set amount of time to refine any UI elements that weren't updated and ensure that interactions were just as intuitive as they could be for this point.
- There weren't any notes on this after the usability testing so there were only a few things that could be done, namely updating the book UI to have new images.

Success Analysis

This sprint was tested to a 100% success rate, and playtesting during this sprint really showed that the addition of aggression really changed the gameplay in a positive way.

Further Refinement

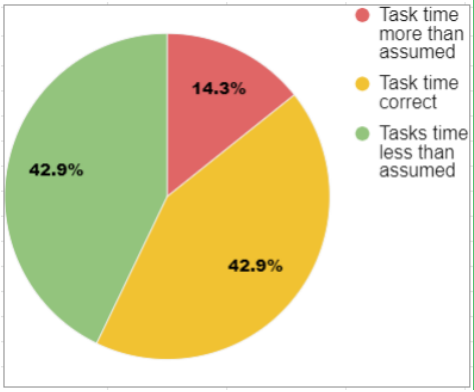
Sound effects will be continuously added and tweaked through all coming sprints, but with the main enhancements and additions occurring in sprint 8.The next time aggression will be reviewed will be when the larger tank enemy is added in sprint 8; although it’s unlikely any major additions will occur until usability testing has taken place at the end of this phase.

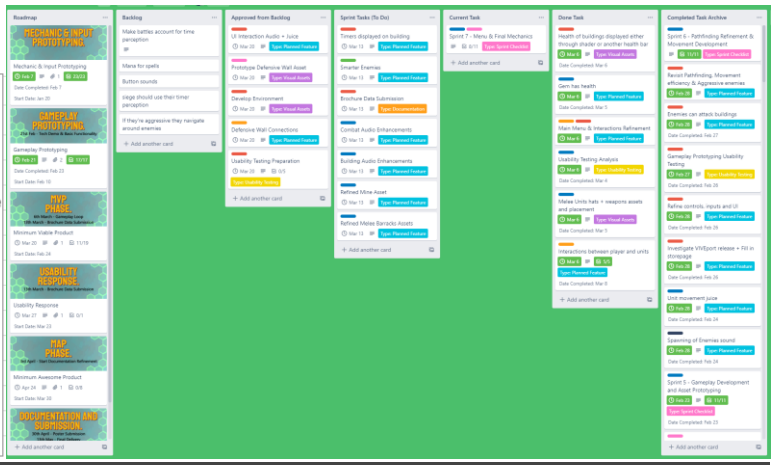
Week 7 - Menu and Final Mechanics (2nd Mar 2020 - 6th Mar 2020)

7	Minimum Viable Product		2nd Mar 2020 - 6th Mar 2020	Supervisor Meeting Date : N/A			
	Sprint 7 - Menu & Final Mechanics		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	100.00%	Gem has health	Planned Feature	2	1	1
	Devlog Link	https://a	Melee Units hats + weapons assets and placement	Visual Assets	2	2	0
	Coding Standards Complete	☑	Main Menu & Interactions Refinement	Planned Feature	12	10	2
	Trello Screenshot Upload	☑	Health of buildings displayed	Visual Assets	4	2	2
	Next Sprint Tasks	☑	Player to Unit interactions	Planned Feature	8	10	-2
	Roadmap Updated	☑	Usability Testing Analysis	Usability Testing	2	2	0
	Branches Merged	☑					
	Task time more than assumed	1					
	Task time correct	3					
	Tasks time less than assumed	3					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3	3	0

7	Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successfull	Notes
	Gem has health	Feature Implementation	Play through a game and check the gem being attacked a	The gem has health that regenerates after a set time	As expected	☑	
	Melee Units hats + weapons assets and placeme	Feature Implementation	Watch enemies spawning and check their animations	Enemies have helmet and sword in the correct positi	As expected	☑	
	Main Menu	Feature Implementation	Play a game through till the end, press restart and exit	Menu appears when the round is over, restart game	As expected	☑	
	Physical Buttons	Feature Implementation	Attempt to press the buttons from a variety of angles and	Buttons press with the hand and nothing else, functi	As expected	☑	Had a few ideas on how to improve
	Health of buildings displayed	Feature Implementation	Play through a level and observe the building health on w	Building health displayed on the hex around the build	As expected	☑	
	Player to Unit interactions	Feature Implementation	Use all 3 of the implemented interactions on enemies	All of the interactions work as expected. Regular deal	As expected	☑	
						☐	
						☐	
						☐	
						☐	

Sprint Complete	06/03/2020
Prototyping	0
Feature Implementations	6
Design Changes	0
Bug Fixes	0
Success Count	6
Total Tasks	6
Percentage Success	100.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- Vive tracker functionality implemented during this sprint. Unforeseen issue where the tracker doesn't work whilst on the players hand, so now the player must have the tracker on the back of their hand.
- Player to Unit Interactions took longer than expected as a complete redesign of the BookUI was required.
- Physical buttons were implemented this phase, and they took a while to get fully implemented. Luckily because these buttons were previously implemented, making a main menu was a bit quicker than previously planned. It only required Restart and Quit, and due to consistent encapsulation, making restart was quick.

Success Analysis

All the features implemented during this sprint were tested to a 100% success rate. All the smart objectives laid out were functioning as expected.

Further Refinement

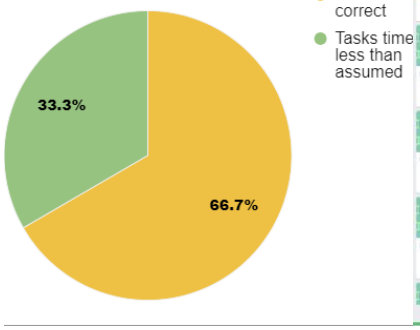
The gem being destroyed will be further refined with audio and visual updates coming in sprint 10 and 11 respectively. This will trigger a clearer game over sequence, which will come with the various UI updates happening in sprint 9. The spell system will also receive two new spells in sprint 9, both one that allows players to slow down units but also to speed them up.

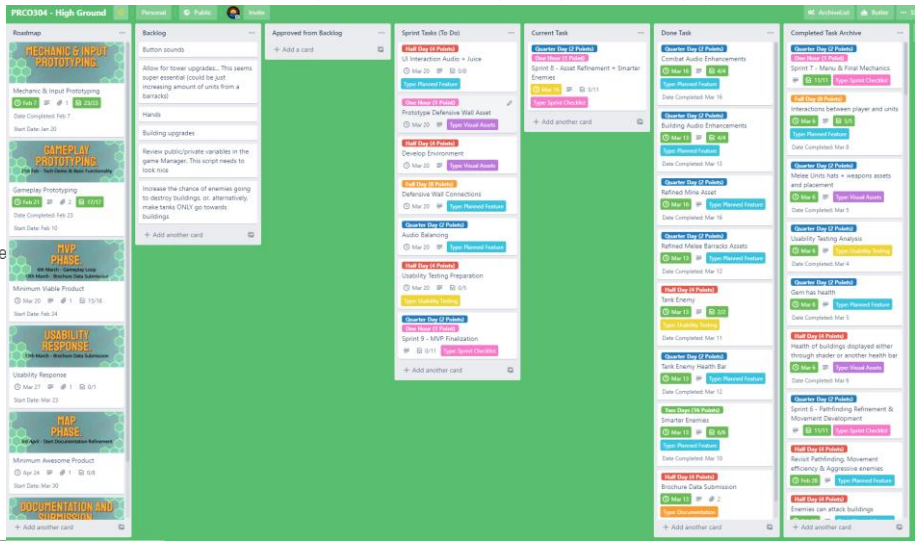
Week 8 - Asset Refinement + Smarter Enemies (9th Mar 2020 - 16th Mar 2020)

8	Minimum Viable Product		9th Mar 2020 - 16th Mar 2020	Supervisor Meeting Date : 10/03/2020 11:00			
	Sprint 8 - Asset Refinement + Smarter Enemies		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate		Smarter Enemies	Planned Feature	16	8	8
	Devlog Link		Brochure Data Submission	Documentation	4	3	1
	Coding Standards Complete	☐	Combat Audio Enhancements	Planned Feature	2	2	0
	Trello Screenshot Upload	☐	Building Audio Enhancements	Planned Feature	2	2	0
	Next Sprint Tasks	☐	Refined Mine Asset	Planned Feature	2	1	1
	Roadmap Updated	☐	Refined Melee Barracks Asset	Planned Feature	2	2	0
	Branches Merged	☐	Tank Enemy	Usability Testing	4	4	0
			Optional Enemy Health Bars	Planned Feature	2	2	0
	Task time more than assumed	0					
	Task time correct	6					
	Tasks time less than assumed	3					
	Sprint Complete	☐	Sprint Conclusion Checklist	Sprint Checklist	3	3	0

8	Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successful	Notes
	Smarter Enemies	Feature Implementation	Observe enemy movements after a variety of buildings be	Enemies tend to follow paths more efficiently, they're	As expected	☑	Enemy refinement continues next s
	Tank Enemy	Feature Implementation	Force a tank enemy to spawn on round 1, then observe it	Tank enemy is fully aggressive, it's difficult to kill, ma	As expected	☑	
	Tank Enemy Health Bar	Feature Implementation	Observe the tank enemy as it takes damage	The health bar points at the camera and reacts to da	As expected	☑	
	Audio Enhancements	Feature Implementation	Play a level and listen to the variety of new audio effects	Audio plays correctly in both 2D and 3D circumstanc	As expected	☑	Audio balancing is a task ready for

Sprint Complete	13/03/2020
Prototyping	0
Feature Implementations	4
Design Changes	0
Bug Fixes	0
Success Count	4
Total Tasks	4
Percentage Success	100.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- Started testing builds this phase, found an issue with building the universal render pipeline but I managed to fix it.
- A few small tasks were delayed into the next sprint, due to illness on the Friday. This included refined mine assets and combat audio enhancements. I also had to delay the end of the sprint by one day due to the illness. I was still able to complete this sprint on the Monday after, which didn't affect the next sprint as it was already lacking.

Success Analysis

All the features implemented during this sprint were tested to a 100% success rate. All the smart objectives laid out were functioning as expected.

Further Refinement

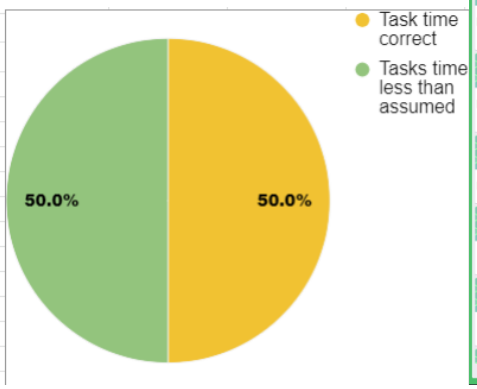
Further refinement of these features will only occur after usability testing at the end of this phase. Hopefully, this will inform any changes to balancing with the Tank Enemy, and any fixes that could improve the user interface. Therefore, sprint 11 will be designated to any potential changes found after this playtesting.

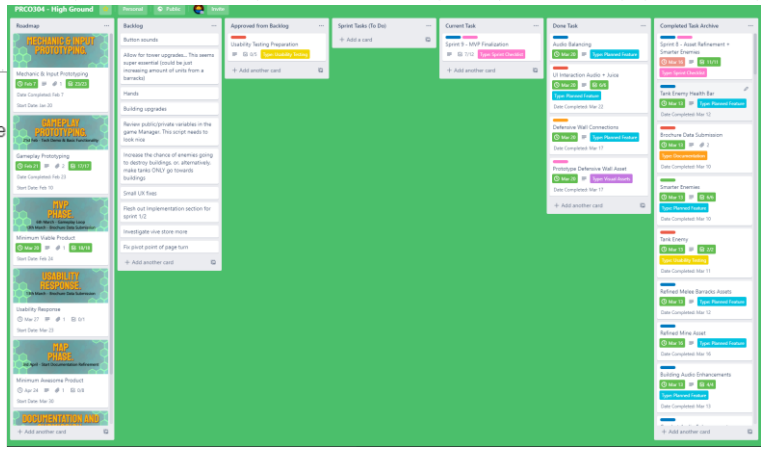
Week 9 - MVP finalization (17th Mar 2020 - 23rd Mar 2020)

9	Minimum Viable Product		17th Mar 2020 - 23rd Mar 2020	Supervisor Meeting Date : N/A		
	Sprint 9 - MVP finalization		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	66.67%	UI Interaction Audio + Juice	Planned Feature	4	4	0
	Devlog Link	https://ja	Prototype Defensive Wall Asset	Visual Assets	1	1	0
	Coding Standards Complete	☑	Defensive Wall Connections	Planned Feature	8	3	5
	Trello Screenshot Upload	☑	Audio Balancing	Planned Feature	2	1	1
	Next Sprint Tasks	☑					
	Roadmap Updated	☑					
	Branches Merged	☑					
	Task time more than assumed	0					
	Task time correct	3					
	Tasks time less than assumed	2					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3	3	0

9	Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successful	Notes
	Defensive Wall Connections	Feature Implementation	Try various combinations of placing walls in different positi	Walls connect as expected. A wall being destroyed r	As expected	☑	
	Audio Balancing	Feature Implementation	Listen to all added audio effects whilst big and small	All sound effects play at an appropriate volume	As expected	☑	
	UI Interactions	Feature Implementation	Play a level of the game, examining the UI for any breaks	Buttons are responsive, no breaks in UI and buttons	A small number of buttons weren't as responsive/clic	☐	Task for this added to next sprint
						☐	
						☐	
						☐	
						☐	
						☐	
						☐	
						☐	

Sprint Complete	20/03/2020
Prototyping	0
Feature Implementations	3
Design Changes	0
Bug Fixes	0
Success Count	2
Total Tasks	3
Percentage Success	66.67%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- I managed to get defensive wall functionality done much quicker than expected. This also included testing during a level to ensure they were the correct size for gameplay.
- Lots of disruption this week as I've had to move home and set up a workstation due to the coronavirus pandemic.
- The week ran one working day over schedule, but luckily my next sprint is a bit less intense so That's not an issue

Success Analysis

All the features implemented during this sprint were tested to a 66.67% success rate. This is due to a few issues with physical buttons on the user interface, all of which were fixed during the functionality testing.

Further Refinement

The only additions made during this sprint that will require further refinement are the changes made to user interface. Any changes will be made after analysing the results from usability testing, hence acted upon during sprint 11.

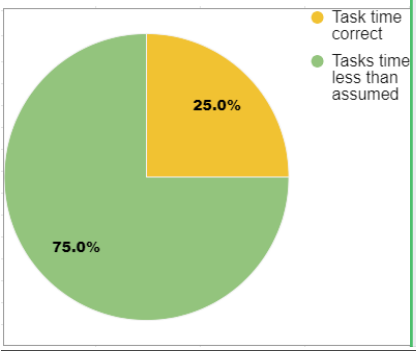
Week 10 - Preparing BETA (23rd Mar 2020 - 27th Mar 2020)

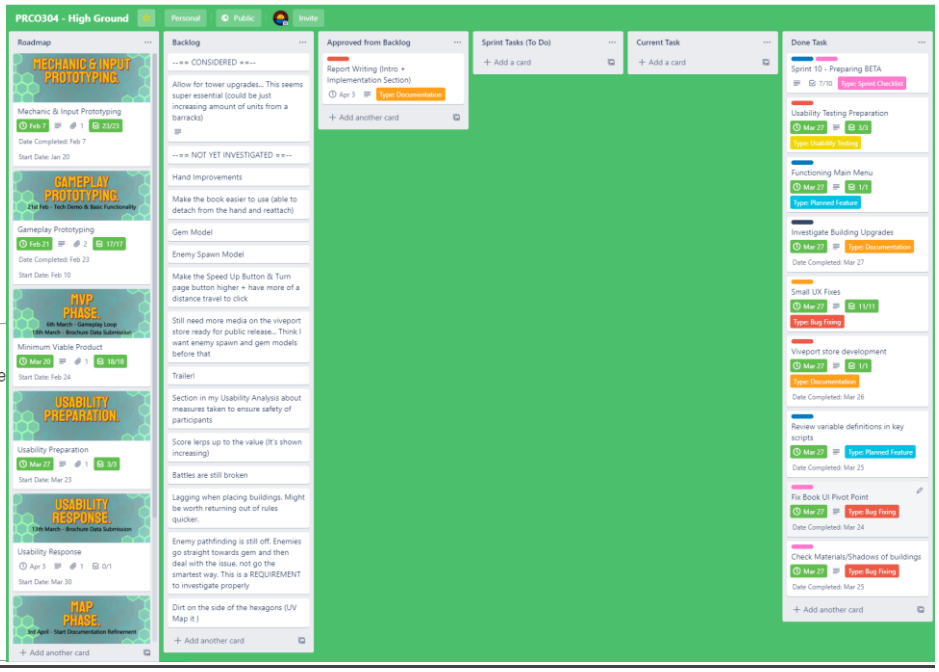
10	Usability Preparation		24th Mar 2020 - 27th Mar 2020	Supervisor Meeting Date : N/A			
	Sprint 10 - Preparing BETA		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	100.00%	Viveport store development	Documentation	4	3	1
	Devlog Link	https://je	Review variable definitions in key scripts	Planned Feature	2	1	1
	Coding Standards Complete	☑	Fix Book UI Pivot Point	Bug Fixing	1	0.5	0.5
	Trello Screenshot Upload	☑	Check Materials/Shadows of buildings	Bug Fixing	1	0.5	0.5
	Next Sprint Tasks	☑	Small UX Fixes	Bug Fixing	8	6	2
	Roadmap Updated	☑	Usability Testing Preperation	Usability Testing	4	3	1
	Branches Merged	☑	Investigate building upgrades	Documentation	0.5	0.5	0
			Functioning Main Menu	Planned Feature	2	2	0
	Task time more than assumed	0					
	Task time correct	2					
	Tasks time less than assumed	6					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3		N/A

10

Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successfull	Notes
Check Materials/Shadows of buildings	Design Changes	Observe the game board with buildings placed on it	Shadows are non-existant, as per the style, material	As expected	☑	
Rumble	Feature Implementation	Cast spells, place buildings, do things that cause the rumt	Rumble occurs in the players main controller when it	As expected	☑	
Functioning Main Menu	Feature Implementation	Test the functionality of the main menu, including play/rest	All current functionality of the main menu works	As expected	☑	
Fix issues with building placement	Bug Fixes	Test all game rules of buildings by placing them in a variet	Building placement calculated correctly	As expected	☑	Bit of lag when running certain requ

Sprint Complete	27/03/2020
Prototyping	0
Feature Implementations	2
Design Changes	1
Bug Fixes	1
Success Count	4
Total Tasks	4
Percentage Success	100.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- As a result of now having to work from home, a bit more preparation is required in order to make the game ready to playtest. Therefore, I have added the Usability Preparation phase which comprises just one sprint. This phase involves preparing the project and Vivestore for the release of the project to BETA testers
- Improvements made to Pathfinding during this sprint, the enemies now check if the node they're navigating onto contains a unit in combat. This means they tend to go around blockages much easier.

Success Analysis

All the features implemented during this sprint were tested to a 100% success rate. All the smart objectives laid out were functioning as expected.

Further Refinement

It is unlikely and of the user experience additions will receive any refinement as it is coming towards the end of development. The main menu however will be updated with functional audio and handedness buttons, which will come in sprint 12. The automatic data tracking is being removed from the project after usability testing takes place.

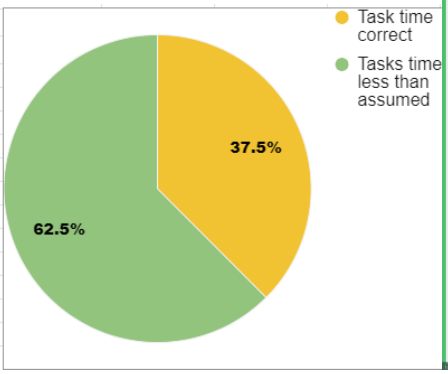
Week 11 - Usability Response (30th Mar 2020 - 3rd Apr 2020)

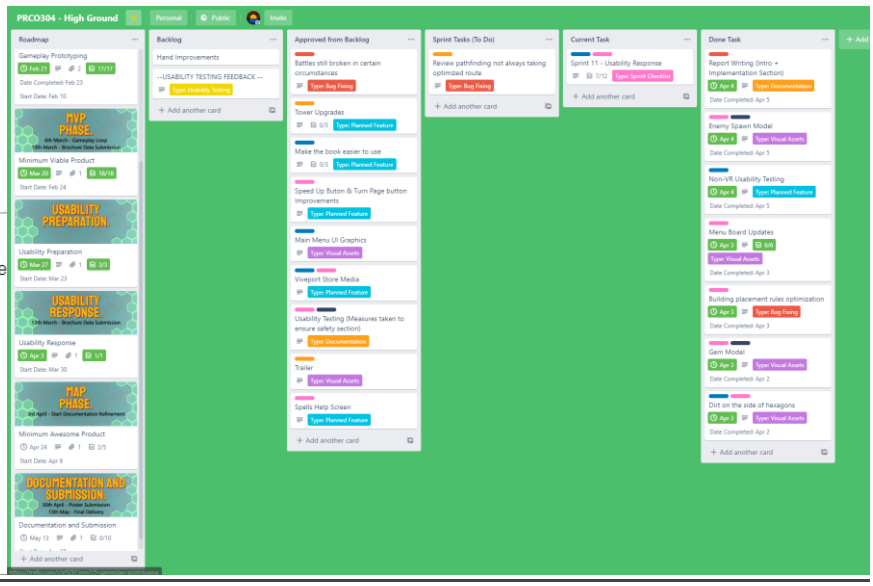
11	Usability Response		30th Mar 2020 - 3rd Apr 2020	Supervisor Meeting Date : N/A		
	Sprint 11 - Usability Response Fixes		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	100.00%	Dirt on the side of hexagons	Visual Assets	3	2	1
	Devlog Link	https://je	Gem Model	Visual Assets	1.5	1	0.5
	Coding Standards Complete	☑	Building placement rules optimization	Bug Fixing	1	1	0
	Trello Screenshot Upload	☑	Menu Board Updates	Visual Assets	1	1	0
	Next Sprint Tasks	☑	Enemy Spawn Model	Visual Assets	1.5	1	0.5
	Roadmap Updated	☑	Report Writing (Intro + Implementation Sections)	Documentation	4	2	2
	Branches Merged	☑	non-VR Usability Testing	Usability Testing	2	2	0
	Task time more than assumed	0					
	Task time correct	3					
	Tasks time less than assumed	5					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3	2	1

11

Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successful	Notes
Building placement rules optimization	Bug Fixes	Place buildings and check all combinations of building pla	Placing buildings creates less lag than before, and t	As expected	☑	
Info Panels on/off button	Feature Implementation	Press the button on/off and check the status of the info pa	Info panels slide away when off, come back when on	As expected	☑	
Music on/off button	Feature Implementation	Press the button on/off and check the status of the music	Music is on when clicked on, and off when off. Optio	As expected	☑	
Effects on/off button	Feature Implementation	Press the button on/off and check the status of the effect	Effects are on when click on, and off when off. Optio	As expected	☑	

Sprint Complete	03/04/2020
Prototyping	0
Feature Implementations	3
Design Changes	0
Bug Fixes	0
Success Count	4
Total Tasks	4
Percentage Success	100.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- Due to the nature of trying to get people testing my project, Usability Response was continuous from this sprint onwards.
- Not really any feedback gained from this week, even though the VR testing method was distributed, and non-VR testing method is to be analysed next week

Success Analysis

All the features implemented during this sprint were tested to a 100% success rate. All the smart objectives laid out were functioning as expected.

Further Refinement

None of the features added during this phase will be refined any further. This is due to both time restrictions and features being already fully implemented at this stage.

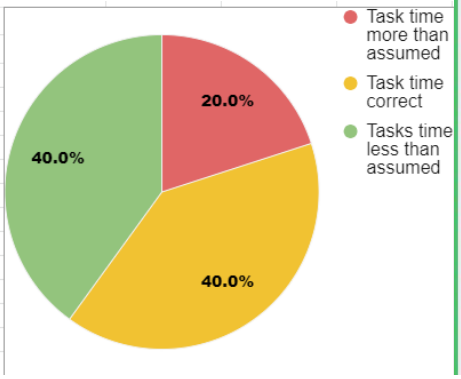
Week 12 - Bug Fixes and Refinement (6th Apr 2020 - 10th Apr 2020)

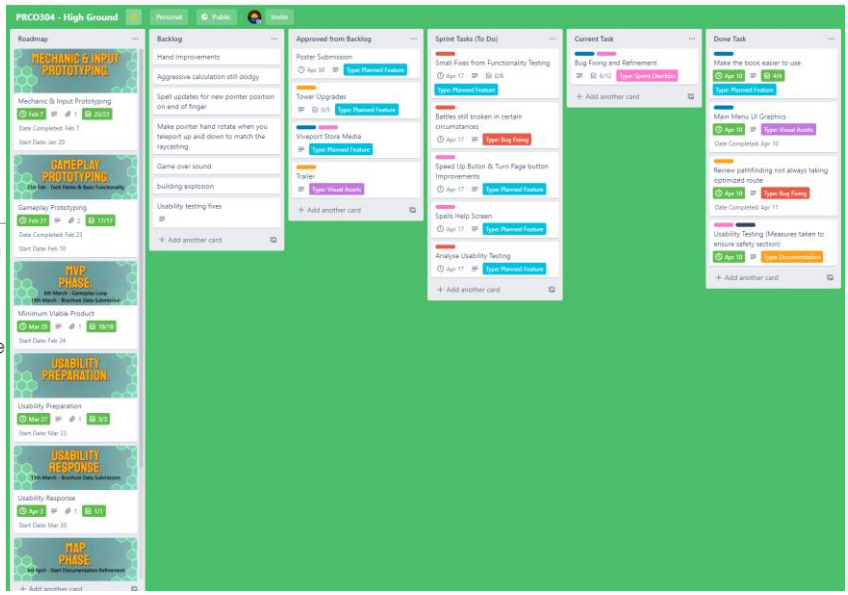
12	Minumum Awesome Product		6th Apr 2020 - 10th Apr 2020	Supervisor Meeting Date : N/A			
	Sprint 12 - Bug Fixes & Refinement		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	50.00%	Review pathfinding	Bug Fixing	8	10	-2
	Devlog Link	https://je	Make the book easier to use	Planned Feature	2	2	0
	Coding Standards Complete	☑	Main Menu UI Graphics	Visual Assets	2	1	1
	Trello Screenshot Upload	☑	Usability testing (Measures taken to ensure safety section)	Documentation	1.5	0.5	1
	Next Sprint Tasks	☑					
	Roadmap Updated	☑					
	Branches Merged	☑					
	Task time more than assumed	1					
	Task time correct	2					
	Tasks time less than assumed	2					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3	3	0

12

Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successful	Notes
Pathfinding Review	Bug Fixes	Create many different environments on which enemies sl	Enemies pathfind appropriately, reacting to changes	Pathfinding drove an enemy into an invalid space.	☐	Very quick fix added to a trello card
Handedness Considerations	Feature Implementation	Choose left and right handed and try all interactions as bo	Both hands work as expected	Left handedness book is misrotated	☐	Very quick fix added to a trello card
New Main Menu Button	Feature Implementation	Run the game and press the main menu button both mid-	Main menu button takes the player back to the main	As expected	☑	
Detatchable Book	Feature Implementation	Play the game as both left and right handed and press gri	Book detatches and functions as expected. Book car	As expected	☑	
					☐	
					☐	
					☐	
					☐	
					☐	
					☐	
					☐	

Sprint Complete	10/04/2020
Prototyping	0
Feature Implementations	3
Design Changes	0
Bug Fixes	0
Success Count	2
Total Tasks	4
Percentage Success	50.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- Pathfinding system fully rebuilt during this phase
- All the considerations necessary for right/left-handed players

Success Analysis

All the features implemented during this sprint were tested to a 50% success rate. The new pathfinding was relatively unrefined at this point, so enemies were still able to navigate onto invalid spaces. Also, the book UI spawned with a strange rotation when the player started the game as left-handed. This meant the player couldn't see the buttons or press them as the book was facing the wrong way. Both issues were fixed quickly after functionality testing, so that issues do not carry over into the next sprint.

Further Refinement

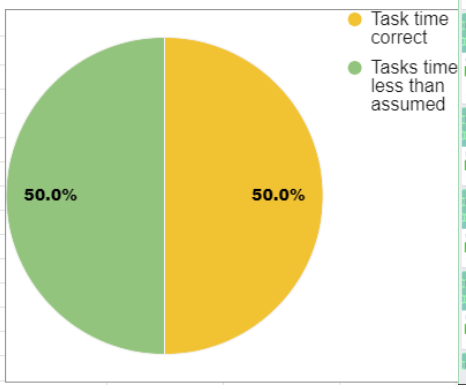
None of the features added during this phase will be refined any further. This is due to both time restrictions and features being already fully implemented at this stage.

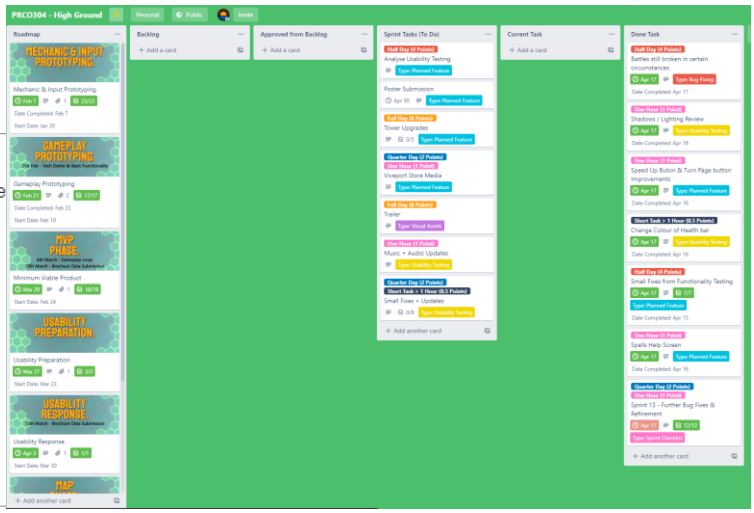
Week 13 - Further Bug Fixes and Refinement (13th Apr 2020 - 17th Apr 2020)

13	Minumum Awesome Product		13th Apr 2020 - 17th Apr 2020	Supervisor Meeting Date : N/A			
	Sprint 13 - Further Bug Fixes & Refinement		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	100.00%	Small fixes from functionality testing	Planned Feature	4	2	2
	Devlog Link	https://e	Battles still borken	Bug Fixing	4	2	2
	Coding Standards Complete	☑	Speed up button & turn page button improvements	Planned Feature	1	0.5	0.5
	Trello Screenshot Upload	☑	Spells help screen	Planned Feature	1	1	0
	Next Sprint Tasks	☑	Change Colour of Health Bar	Usability Testing	0.5	0.5	0
	Roadmap Updated	☑	Shadows/Lighting Review	Usability Testing	1	1	0
	Branches Merged	☑					
	Task time more than assumed	0					
	Task time correct	3					
	Tasks time less than assumed	3					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3		N/A

13	Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successful	Notes
	Aggression Review Fixes	Bug Fixes	Observe changes in aggression as enemies are trapped	Enemies get aggressive at an appropriate speed in r	As expected	☑	
	Enemies navigating through buildings	Bug Fixes	Leave game running with a variety of buildings placed and	Enemies navigate around buildings as expected, and	As expected	☑	
	Left handedness book mis-rotated	Bug Fixes	Start and play game both as left handed and right handed	Book in the correct hand and returns to the correct h	As expected	☑	
	Casting spells broken (out of wrong pos and ang	Bug Fixes	Cast spells as both left and right handed whilst large and s	Spells emit from the correct point on the hand, and h	As expected	☑	
	Colour of health bar	Design Changes	Setup a building to be destroyed and observe it being slap	Health bar colours as it gets damaged	As expected	☑	
	Shadows and Lighting	Design Changes	Play the game and place some buildings. Observe lighting	Lighting acts properly, shadows are visible	As expected	☑	
	Battles still broken	Bug Fixes	Run the game for a while, at double speed, with a large v	Battles do not break after they end.	As expected	☑	
						☐	
						☐	
						☐	
						☐	

Sprint Complete	17/04/2020
Prototyping	0
Feature Implementations	0
Design Changes	2
Bug Fixes	4
Success Count	7
Total Tasks	7
Percentage Success	100.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- Bug notices where the player could leave the book on the ground whilst small.
- Bug noticed where tank wouldn't use the aggressive pathfinding. It now does

Success Analysis

All the features implemented during this sprint were tested to a 100% success rate. All the smart objectives laid out were functioning as expected.

Further Refinement

None of the features added during this phase will be refined any further. This is due to both time restrictions and features being already fully implemented at this stage.

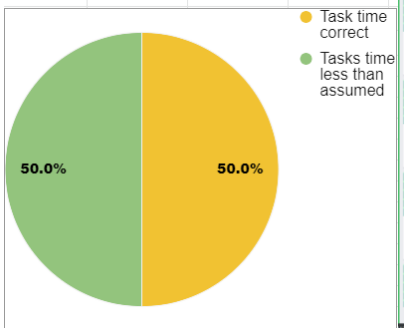
Week 14 - Final Additions (20th Apr 2020 - 24th Apr 2020)

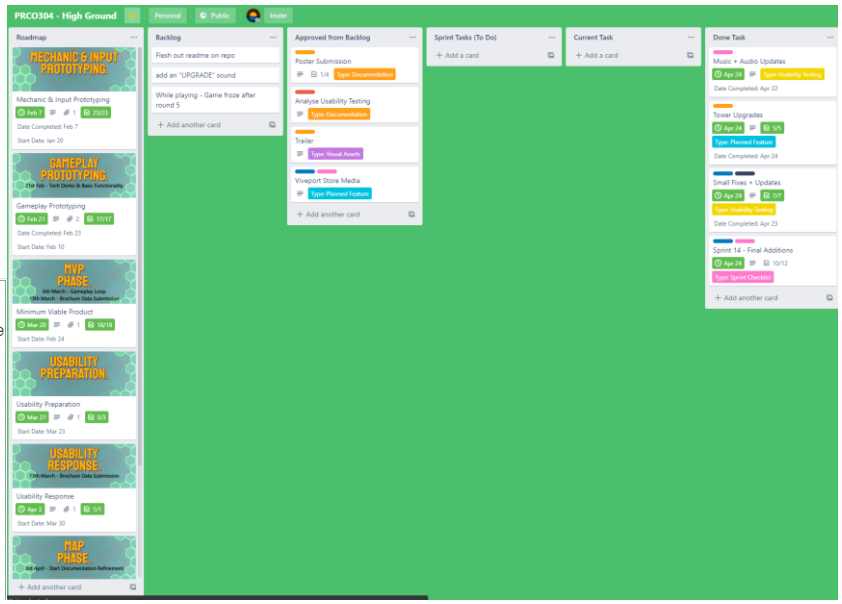
14	Minumum Awesome Product		20th Apr 2020 - 24th Apr 2020	Supervisor Meeting Date : N/A			
	Sprint 14 - Final Additions		Tasks	Task Type	Planned Hours	Actual Hours	Discrepancy
	Functionality Testing Success Rate	100.00%	Tower Upgrades	Planned Feature	8	6	2
	Devlog Link	https://je	Music + Audio Updates	Usability Testing	1	1	0
	Coding Standards Complete	☑	Small Fixes + Updates	Usability Testing	2.5	2	0.5
	Trello Screenshot Upload	☑					N/A
	Next Sprint Tasks	☑					
	Roadmap Updated	☑					
	Branches Merged	☑					
	Task time more than assumed	0					
	Task time correct	2					
	Tasks time less than assumed	2					
	Sprint Complete	☑	Sprint Conclusion Checklist	Sprint Checklist	3	3	0

14

Task Name	Task Type	Testing Method	Expected Result	Actual Result	Successful	Notes
Building Explosion Particle Effect	Feature Implementation	Create a situation where an enemy has to destroy a build	Particle effect appears and destroys after running.	Particle effect didn't destroy after	☑	One line of code fixed this during te
Everything dies when gem does	Feature Implementation	Place some barracks and allow the enemy to win	All units on the board die at the same time, making it	As expected	☑	
Tick sound happening at wrong time	Bug Fixes	Start a game and listen to the ticking sound	Tick sound happens when the text on the clock chan	As expected	☑	
Barracks placement errors	Bug Fixes	Give the player a huge amount of money and test placing	Barracks placement meets all rules outlined in the G	As expected	☑	
Enemy group size increase over rounds	Feature Implementation	Create a situation where the enemies cannot get through	As rounds go on,the size of groups increases to reac	As expected	☑	
Tower Upgrades	Feature Implementation	Try and upgrade a tower more than expected, and observ	Tower upgrades to a limit of 6, UI display hides when	As expected	☑	
Music Upgrades	Feature Implementation	Listen to the music as it moves between building and acti	Music changes between rounds and between going	As expected	☑	
					☐	
					☐	
					☐	
					☐	

Sprint Complete	24/04/2020
Prototyping	0
Feature Implementations	5
Design Changes	0
Bug Fixes	2
Success Count	7
Total Tasks	7
Percentage Success	100.00%
Testing to do	DONE
Tasks to Test	0
Percentage to Test	0.00%





Notes

- Equation for how enemy amounts are calculated is visible here http://prntscr.com/s4poda
- Clouds greatly improved during this sprint as transparency was added. This was a very simple addition so didn't warrant an entire task

Success Analysis

All the features implemented during this sprint were tested to a 100% success rate. All the smart objectives laid out were functioning as expected.

Further Refinement

None of the features added during this phase will be refined any further. This is due to both time restrictions and features being already fully implemented at this stage.

Appendix 10: Usability Testing Results and Analysis

The following Appendix details the usability testing that took place during the project, the data collection methods used and how the collected data was used during development.

Data Collection Methods

In order to ensure the project continues to deliver optimal user experience, consistent usability testing sessions are imperative. A variety of data collection methods will aid in analysing both qualitative and quantitative data collected from users.

Interviews + Note Taking

Note taking will be the primary source of feedback during all testing sessions with a playable demo. Notes will be considered after the testing session, where they are judged as to which pieces of feedback could be realistically taken on within the time constraints of the project. These will be turned into tasks in the Backlog, and then developed further and placed within a sprint ready for development.

Questionnaires

For the larger usability sessions, there will be tailored google form questionnaires to suit the subject of the testing. These will aim to ask Likert-style questions, as well as spaces for extra comments - giving a large variety of data for analysis. An ongoing high ground feedback form will also be setup, that will act as a constant source of feedback for people who want to leave additional comments.

These questionnaires will provide some easily quantified data on people's opinions, and possibly guide more helpful feedback than just note-taking. The results from these will run through very much the same process as note taking, where the feedback will be turned into a list of actionable tasks, deliberate whether they're within scope, and then turn appropriate feedback into actionable tasks on trello.

Automated Data Streaming

The live data streaming serves as the main source of numerical data during testing. It uses the helper class - "WWWForm", to formulate data we record into a form for posting to web servers, in this case, Google Forms. These can then stream data into a google sheet, which allows us to create graphs, use conditional formatting, and calculate averages of certain gameplay elements. This will be used to record various parts of a player's session, such as: playtime, number of kills, building choices, gameplay strategies and other data. All these values will aid in creating conclusions on various aspects of gameplay, such as how long on average it took somebody to complete a round.

This can then be used to direct the next sprint of development based on this data, for example, if the players were going for too long without killing enemies, the amount spawned per round

could be increased. This data collection method allows for some numerical backing behind people's opinions, so it's more justified when improvements are made to certain elements of gameplay.

Safety Measures Taken

During the usability testing of this project, participants regularly had their vision impaired by wearing a virtual reality headset. To ensure the safety of everybody during these sessions, People were asked whether they had used virtual reality before, and if they hadn't, they were given a while to get acquainted with the controllers and having the headset on. Players were always made aware of the tether wire, as to not create a trip hazard. The play area was always completely clear of any items the player may trip over, reducing the risk of harm during these sessions. Participants were also free to take off the headset at any time if they felt uncomfortable.

Testing Planning

Phase 1: Mechanic and Input Prototyping

Planned Usability Sessions: 2

Towards the start of the phase, a questionnaire will be written and distributed within the university to gain an idea on people's perceptions of movement systems and motion sickness and virtual reality. The data from this will be used to direct the development of the movement system within the project, as many prototypes will need to be developed and tested before full implementation.

By the end of the phase, the basic input methods and game mechanics will be prototyped. These will then need to be tested through a small demo session. This will take the form of a very simple form of tower defence, where the player can place buildings that are instantly destroyed by incoming enemies. If the enemies reach the centre gem, the session would be over. The test subject would then be interviewed on their opinions on the game board height, the VR movement systems and controls, the game board resolution and the general gameplay pacing. This information will then be used to tweak the values of various systems developed during this phase.

Phase 2: Gameplay Prototyping

Planned Usability Sessions: 1

The purpose of this phase is to develop the core gameplay systems into a working prototype. A usability testing session will aid in reviewing these systems and give some feedback on the core gameplay experience that will be used to guide development through the Minimum Viable Product Phase. This will take the form of a demo with working combat, pathfinding, building placement and refined controls. A test subject will play through a demo level, and then fill in a questionnaire that aims to retrieve some data on their overall experience. Notes will also be taken during these gameplay sessions, to catch extra feedback.

Phase 3: Minimum Viable Product Phase

Planned Usability Sessions: 1

The aim of this usability testing session is to gather feedback on the amalgamation of all planned features. By this point in development, I will have all core functionality in the project, and it will also have a full game loop ready for people to playtest. The feedback given from this will guide the project into completion and will mainly be feedback relating to general usability of controls, ease of understanding of mechanics and game balance. Previous usability testing sessions will have aided in reducing the amount of improvements received during this testing session.

Phase 6: Minimum Awesome Product Phase

Planned Usability Sessions: 1

Testing Results and Analysis

Phase 1: Mechanic and Input Prototyping

Session 1 - Questionnaire: 24/01/2020 to 04/02/2020

Links to Evidence

Empty Questionnaire
Form Responses

The questionnaire started off with a basic consent wall, ensuring test subjects understood that the information they provided will be used to aid in the development of the project, as well as possibly shared anonymously through social media. If the user did not consent, they were unable to fill in the questionnaire. All 18 recipients of the questionnaire were University students.

The first questions aimed to gain an idea of the percentage of people that suffer from motion sickness, and how it's typically caused. Figure 1.1 shows that over 65% of the research group either don't suffer or suffer very little from motion sickness, with just over 10% suffering considerably from it. 15 of these participants then went on to describe what causes motion sickness for them, with 7 of them describing movement in some form a vehicle, which is a very typical cause of motion sickness.

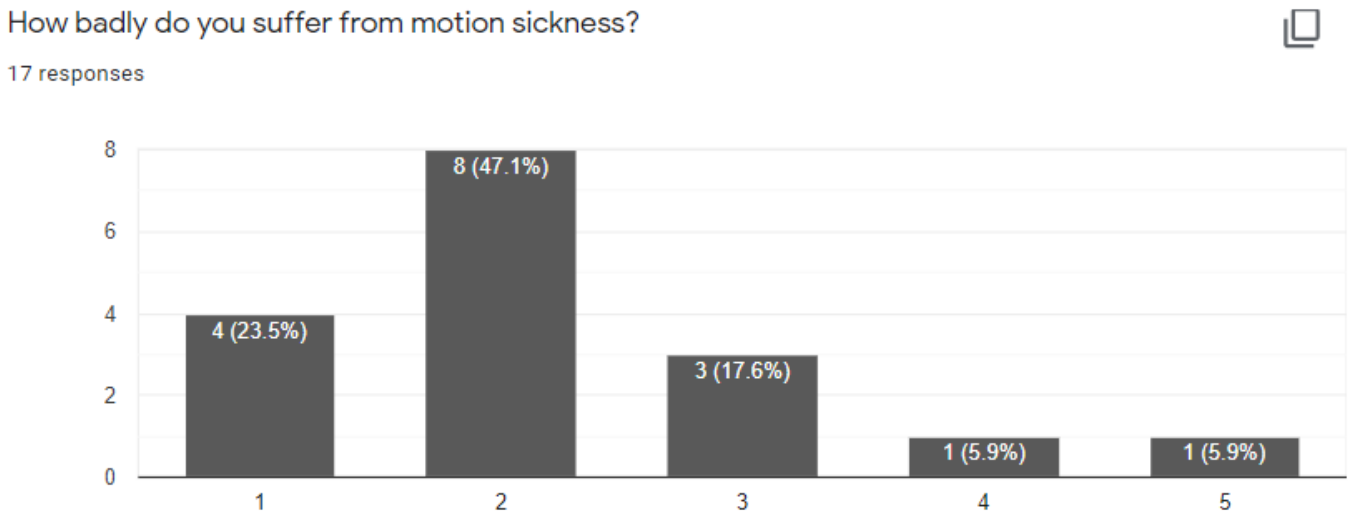


Figure 1.1 *- Results of "How badly do you suffer from motion sickness?" from 17 responses.*

Alongside asking participants what caused their motion sickness, they were also asked what they do to relieve it. There were 14 responses to this question, with over 9 of them saying they simply take a break from moving or get some fresh air. The purpose of these questions was to gain an idea of how many people suffer from motion sickness, why they suffer and how they combat it. This data can be used to help guide future usability testing, ensuring that participants are never uncomfortable whilst testing the project.

The questionnaire then moved onto their experiences in Virtual Reality, and how/whether they have suffered from motion sickness from this. The question shown in Figure 1.2 asks participants whether they have ever experienced motion sickness in VR, and whether they normally experience motion sickness under other circumstances. 47% of participants agreed that VR doesn't have any effect on them, and they usually don't get motion sickness. Over 17% said that they do suffer from motion sickness both in and out of VR, and over 29% of participants suffer in Virtual Reality only. This is a surprising figure, as it proves people may get sickness only from playing VR, and just shows how much consideration must be made when developing projects and movement systems within VR projects.

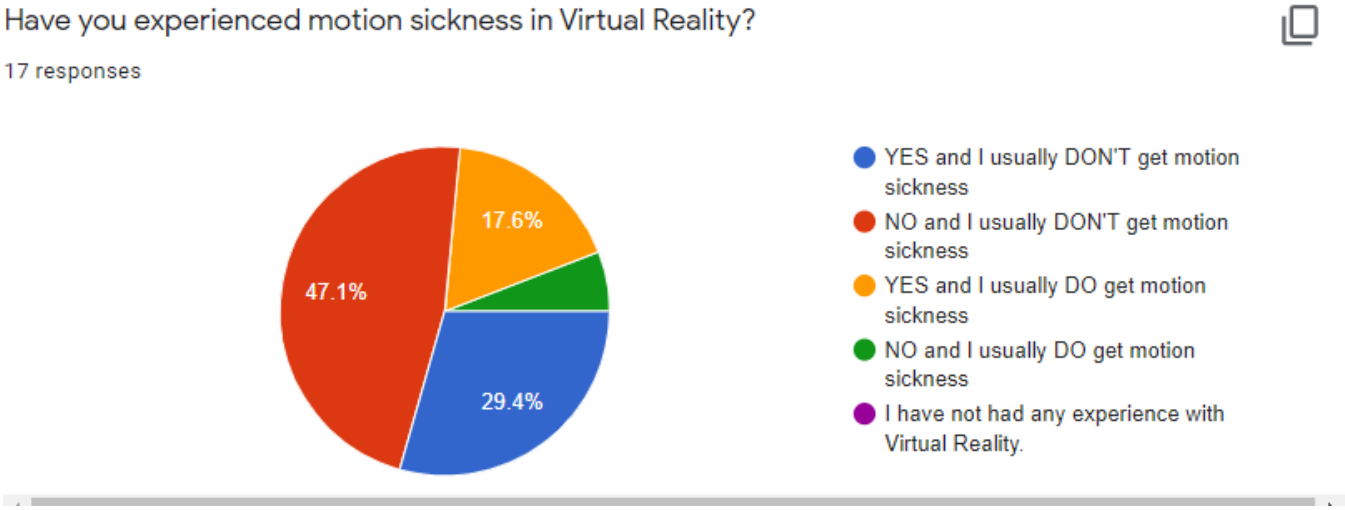


Figure 1.2 *- Results of "Have you experienced motion*

sickness in Virtual Reality?" from 17 responses.

The test subjects were then asked whether there were any VR experiences that may cause motion sickness. A lot of the examples provided were examples where the player has free locomotion, where the player is moved around in game without them moving themselves. This has been an issue with most VR titles, so hence should be a movement system that is avoided when developing a new VR project.

The questionnaire then moves on from motion sickness, specifically VR experiences. The aim of this section is to see whether there are any interactions in Virtual Reality that people really enjoy, and how that could be implemented into the development of High Ground. The first question was "What would you say is the best experience you've had in VR?", to which most participants answered games where gameplay was optimal at around 3-8 minutes. Beat Saber, GORN and The Lab were named most frequently, as players have short periods of intense gameplay and then breaks in which they can take the headset off. This will be something taken into consideration when designing the gameplay of this project.

Test subjects were then asked what particularly made their experience immersive. Out of 13 responses, only 2 mentioned graphics quality. Hence, a fair assumption could be made that graphics quality (textures, models and realism) don't particularly matter too much to the overall success of the project. A lot of participants suggested that interactions with the environment are the source of immersive gameplay, things like picking up and throwing objects.

Figure 1.3 displays the responses from the final question, which aimed to gather preferences on the movement systems available to VR. The goal of this project is not to reinvent a Virtual Reality movement system, so an already established one will be adopted in this project. 29.4% of participants agreed that Pointing and Teleporting is the favourite movement system, with an equal percentage also agreeing they have no preference. Therefore, this is the movement system that will be adopted in this project.

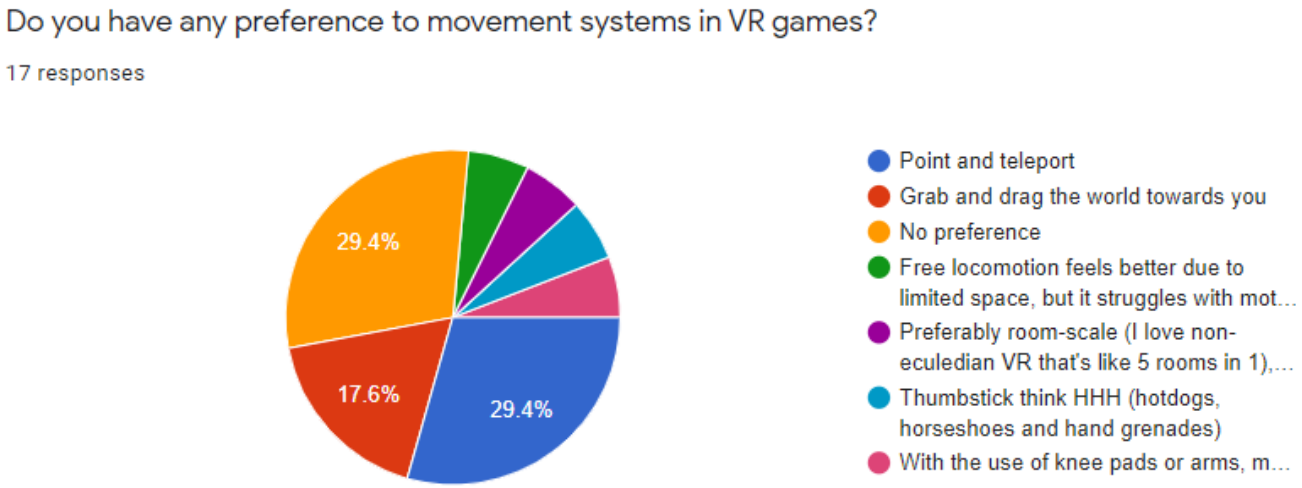


Figure 1.3 *- Results of "Do you have any preference to Movement systems in VR games?" from 17 responses.*

To conclude, this usability testing questionnaire achieved its goal of gathering the general preferences of players in VR experiences. From these results, the pointing and teleporting movement system has been chosen, and more consideration than originally planned will be put into the environment interactions. It seems the general planned gameplay of High Ground has a very low chance of causing motion sickness, however it is important that it constantly remains a consideration whilst developing.

Session 2 - Basic Mechanics and Inputs Demo: 11/02/2020

Links to Evidence

Gameplay Demo
Participant # 1 Footage
Participant # 2 Footage
Participant # 3 Footage
Participant # 4 Footage
Participant # 5 Footage

This session was a successful demonstration of the core input and interactions within the project. Ideally, more participants would have tested the project, but at this phase there was not much to test, so more testing wouldn't have been as beneficial as it would be later in development. After playing the demo, test subjects were asked their overall opinions on the gameplay pacing, the size of the game board, and the VR inputs currently implemented. All the notes were amalgamated into a list, visible in Figure 2.1, alongside a list of bugs that were spotted during this session.

Notes	**Bugs**
● Change angle of handle raycast as pointing is weird from the current hand angle ● Scriptable objects should be used for unit properties ● Height of the board is perfect ● Dimensions (Number of hexes) is hard to tell if it's appropriate as current gameplay isn't TOO representative of final gameplay. ● Pace of gameplay was good ● Too easy to place multiple buildings in one swipe, change it to single click single place ● Cost of all buildings is completely off (The game was not balanced to any extent during this session) ● Should cost money to damage units ● Gold should be generated at the end of a round, not continuously through it ● Money should be displayed on the book ● Timer should also be displayed on the book, or even on the wrist (Wrist is planned) ● Worth trying a higher resolution of hexagons for more strategic gameplay later.	● Placing barracks whilst shrunk down places it within the floor ● Player can teleport onto occupied nodes ● Enemies need to be children of the environment, so they scale appropriately ● Enemies can stack on top of each other whilst moving. ● First node of movement is broken, they teleport in from Vector. Zero instead. ● Player can build on nodes adjacent to enemy spawns

Figure 2.1 *- Notes from the Usability Testing Session.*

From this session, a task has been created to fix all the bugs noticed within the session. This task will be completed during the first week of the next phase, gameplay prototyping. All the notes taken have been considered, and due to the early stages of the project, they're all simply things that will be fixed through further development.

Phase 2: Gameplay Prototyping

Session 1 - Core Mechanics Testing Session: 25/02/2020 to 26/02/2020

Links to Evidence

Gameplay Demo
Participant # 1 Footage
Participant # 2 Footage

This testing session took the form of a short gameplay demo, followed by a questionnaire on the player's experience during the game. This was a relatively successful demo, with 7 participants from different VR gaming experiences all getting to different rounds within the game, using a variety of different tactics. This version of the game was still lacking the ability for the player to 'shoot' the enemies, so there were still moments where players attempted to interact with the units but couldn't. As is visible in the proportion of content that was recorded from this session, users regularly encountered issues with the pathfinding of the enemies. This was causing enemies to queue up and get stuck in certain spaces, with no chance of them ever navigating around them. This will be fixed in future updates, however the feedback received from this point of development was helpful to aid the start of MVP phase.

The questionnaire first had a few images to show users the controls and basic idea of the gameplay, and the purpose of the buildings. These images were there to replace a tutorial, as the project didn't yet have a tutorial implemented.

Figure 3.1 *- The images that players saw before trying the demo.*

The first question starts off asking the players their opinions with the intuitively of the controls. This took the form of a Likert scale question, ranging from 1 - “I had no idea how to do anything” to 5 - “They were very easy to learn!”. Although the previous usability testing had already gained some feedback on the controls, the mechanics had developed enough to ask again. The results (Figure 3.1) show that the controls remain intuitive to most users, due to their simplicity. This may change as development continues, and more interactions are added to the project.

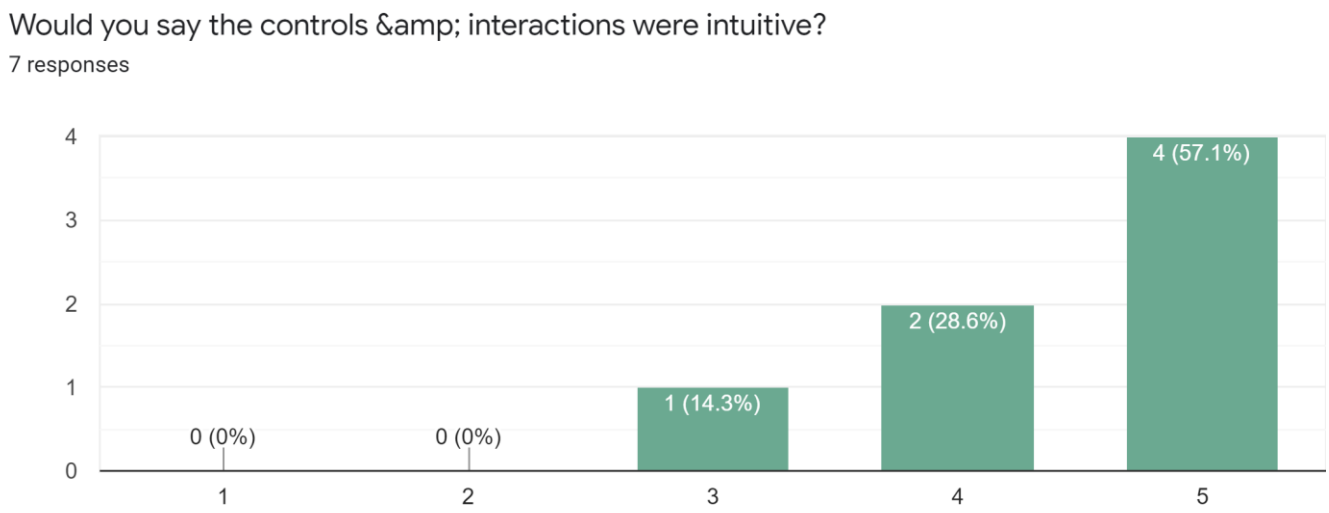


Figure 3.1 *- Results of “Would you say the controls and Interactions were intuitive?” from 7 responses.*

Participants were also asked for any extra comments on the controls and interactions. The main comments to take this list were that players wanted some sort of in-game controls, as well as extra feedback when placing/selecting buildings. Some participants also mentioned that they enjoyed the ability to pick up enemies after they had died, so some more consideration will be taken towards these sorts of interactions in future development.

The next question is yet another Likert question asking the user if they enjoyed the gameplay in the demo. The aim of the question was to gain an idea on the general pacing of the levels and balancing of the gameplay. Some participants needed an extra prompt as to the meaning of this question, as the description of the question assumed players knew what ‘balancing’ meant. The description of the questions will be improved in future questionnaires. As visible in Figure 3.2, more than 70% of players rated the gameplay as extremely enjoyable. This is a promising statistic for this stage of development, as despite the rudimentary game balancing and bugs, players still found the project fun.

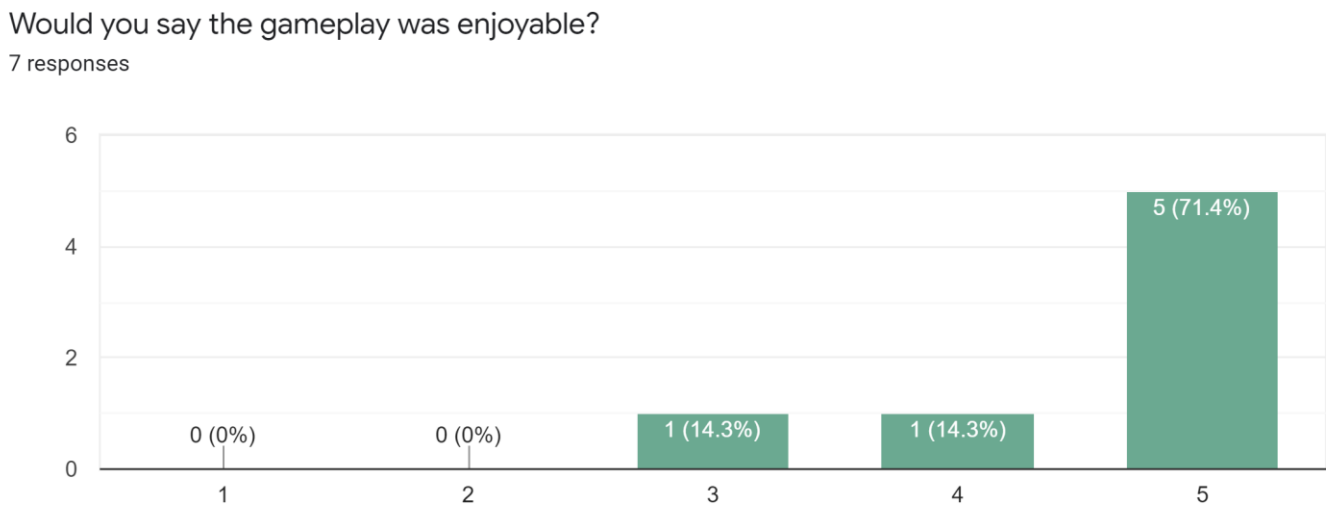


Figure 3.2 *- Results of "Would you say the gameplay was enjoyable?" from 7 responses.*

As with the previous question, players were asked if they had any extra comments on the gameplay. The main comments to be noted from this feedback was that players spent a lot of time waiting around, as queues of enemies started to form in front of their barracks. This will be fixed when enemies are able to destroy the player's walls, which will be implemented during MVP phase. At that point, enemies will become increasingly aggressive and more likely to navigate around the 'traffic'.

High Ground VR is going to be released on the Viveport store, so the next few questions are guided around people's inclinations towards playing or even purchasing the game once it's released. Firstly, the participants were asked whether they would play the game once it's complete. Figure 3.3 shows that 100% of participants agreed that they would play it, if they had the equipment available.

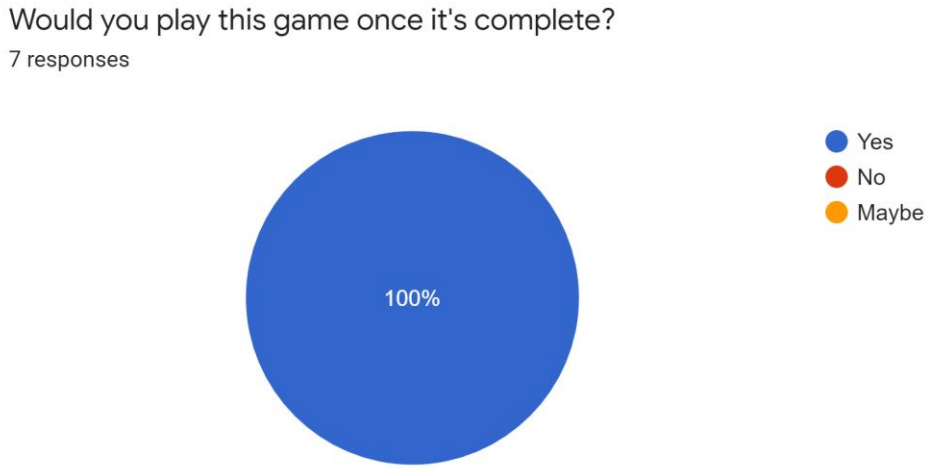


Figure 3.3 *- Results of "Would you play this game once it's complete?" from 7 responses.*

The next question was whether the participant would pay for this game, once it was complete. Figure 3.4 shows the results, with over 70% people saying they would, and 14% unsure. When asked how much they would expect to pay for this sort of project, £3 - £5 seemed to be the average price range given. This value will be used, alongside market research, to inform the pricing of this project on the Viveport store.

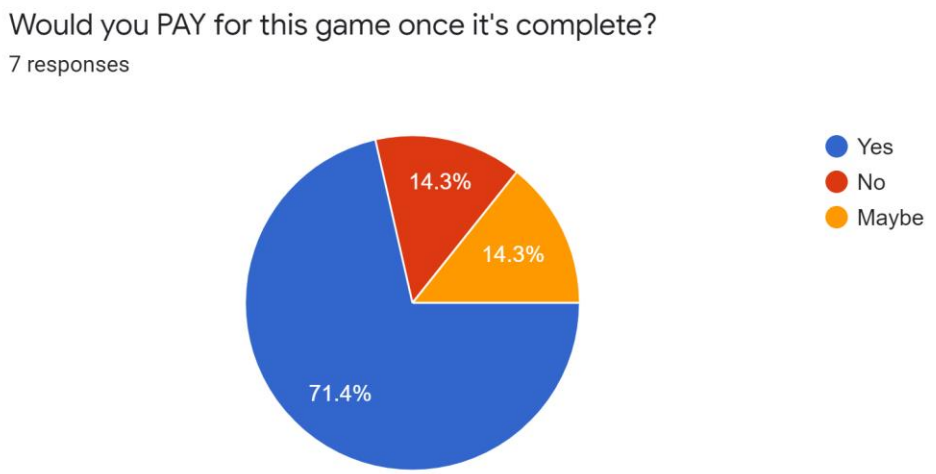


Figure 3.4 *- Results of "Would you pay for this game once it's complete?" from 7 responses.*

From this session, a lot was learnt about people's strategies when defending the gem. Many issues were also noted, which have since been fixed ready for the development in MVP phase. The project is in good standing ready for further development, and once player interactions with enemies are implemented, further game balancing can occur.

Phase 3: Minimum Viable Product Phase

Session 1 – Full Gameplay Loop VR Testing: 30/03/2020

Links to Evidence

Gameplay Demo

Participant # 1 Footage

Due to the coronavirus pandemic, there was only 1 play tester of the VR version of this project. That unfortunately meant that there wasn't a large amount of feedback. This session took the form of a gameplay demo alongside a questionnaire. The structure was the same as the previous usability session, starting with some instructional images and then moving onto questions after they play a short demo. This section of analysis aims to explain the justification behind each question, and will conclude with an amalgamation of all feedback and tasks generated from this session.

The questionnaire section started with asking the player's their opinions on the control scheme. The combination of the following 3 questions aimed to gain both qualitive and quantative feedback on the control scheme. However, with only one play tester, the quantative data is rather redundant.

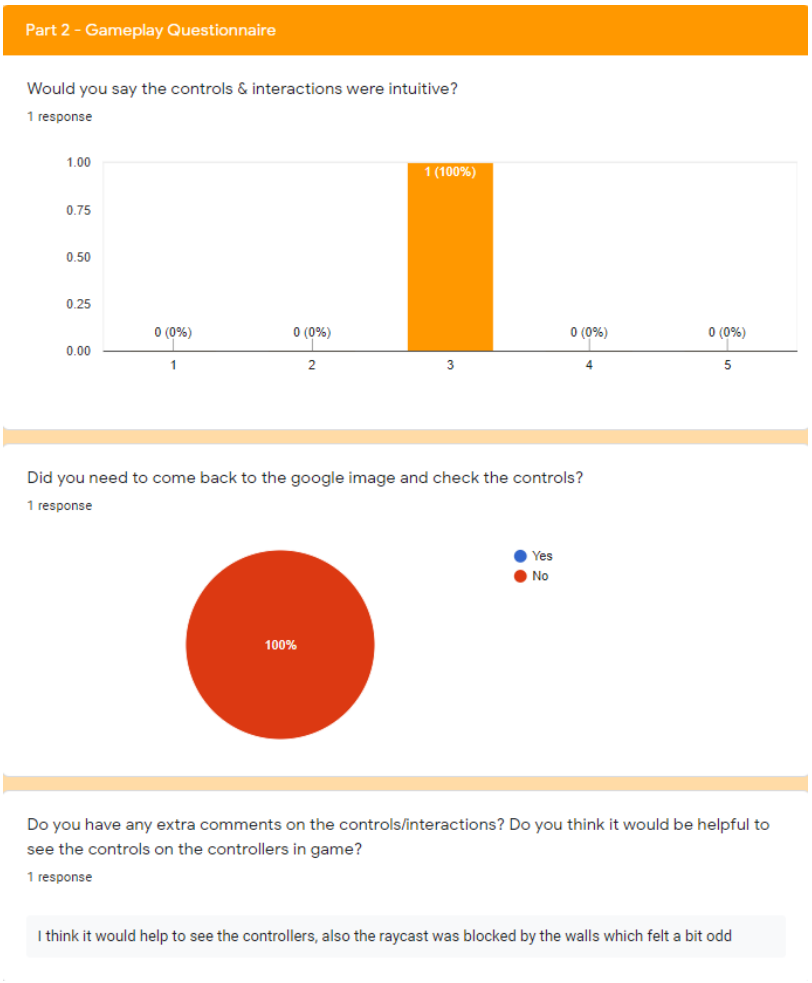


Figure 4.1 – *Controls Questions*

Enjoyable gameplay is one of the key objectives behind this project. The following two questions are focused on the actual gameplay and not interactions. The play testers this session was aimed at were experienced in the sector, so the differentiation behind gameplay and controls was known by the person testing it. The response is positive, receiving a 4/5. The feedback section below contains a well formulated list of small changes that can be done quite quickly and positively change the experience.

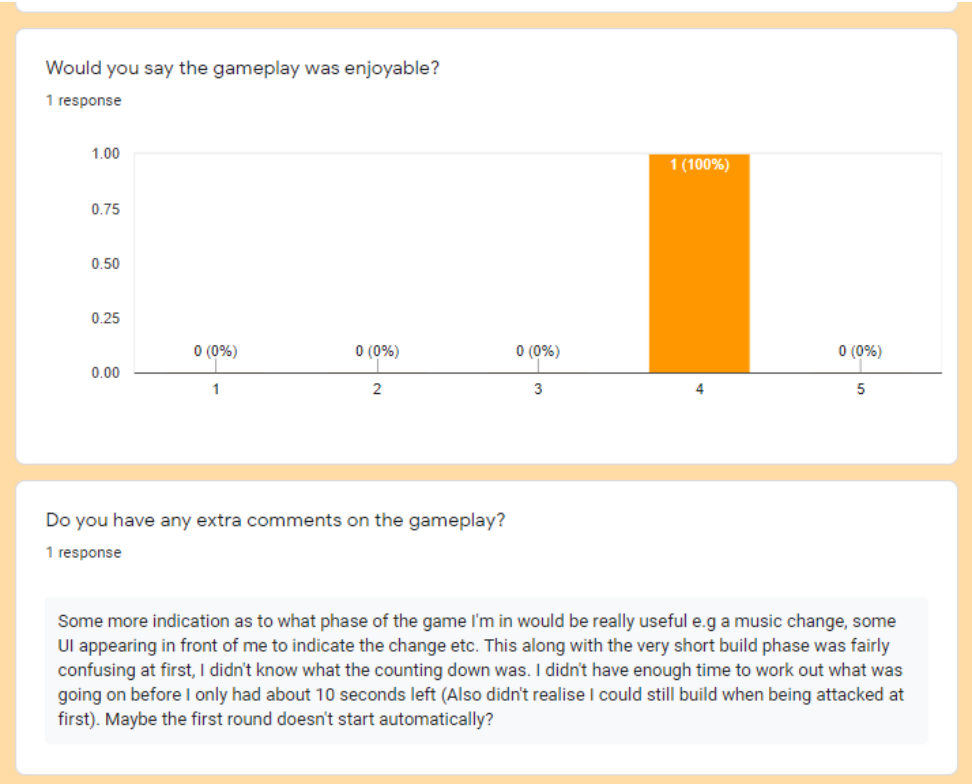


Figure 4.2 – *Gameplay Questions*

The following 3 questions were guided in a way that aims to gain feedback and suggestions from players. These were specifically aimed at the VR interactions, as this point of development aims to refine that area.

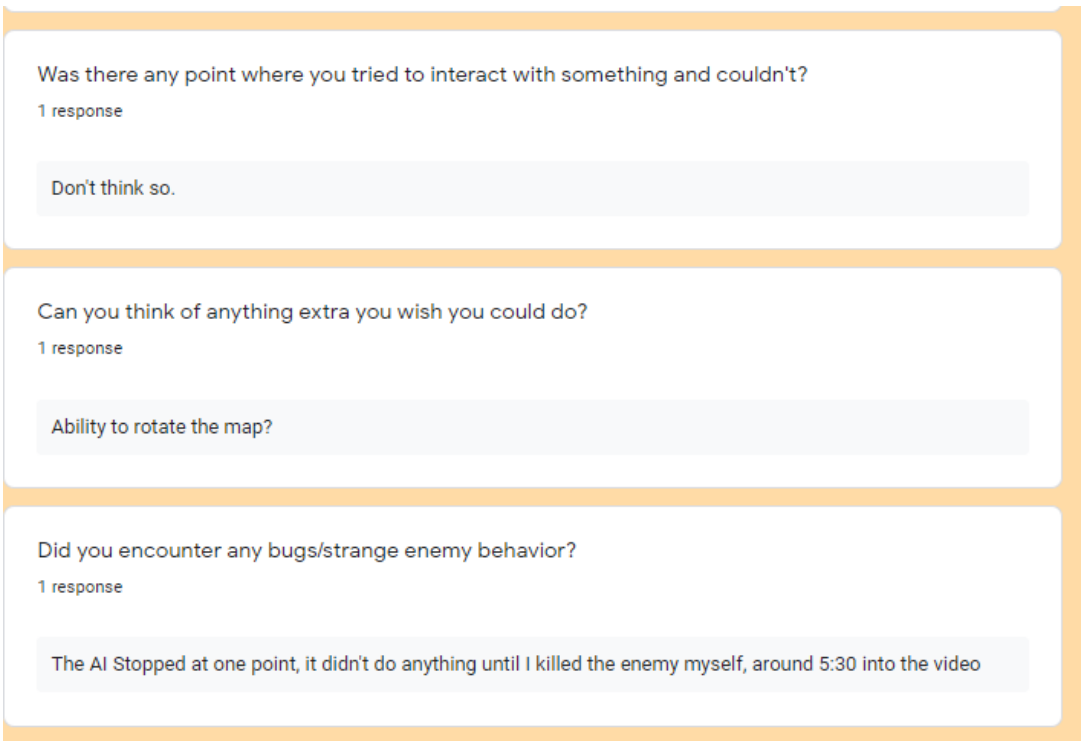


Figure 4.3 – *Extra Suggestions*

There was a minimal bug noticed during gameplay, which has been added to Trello so that it can be investigated. Otherwise, the suggestion of rotating the map has been trialled and removed earlier in development. To conclude the questionnaire, there was a pretty open question. “Do you have any comments on the overall experience?”. This proved to be the source of most feedback from this session.

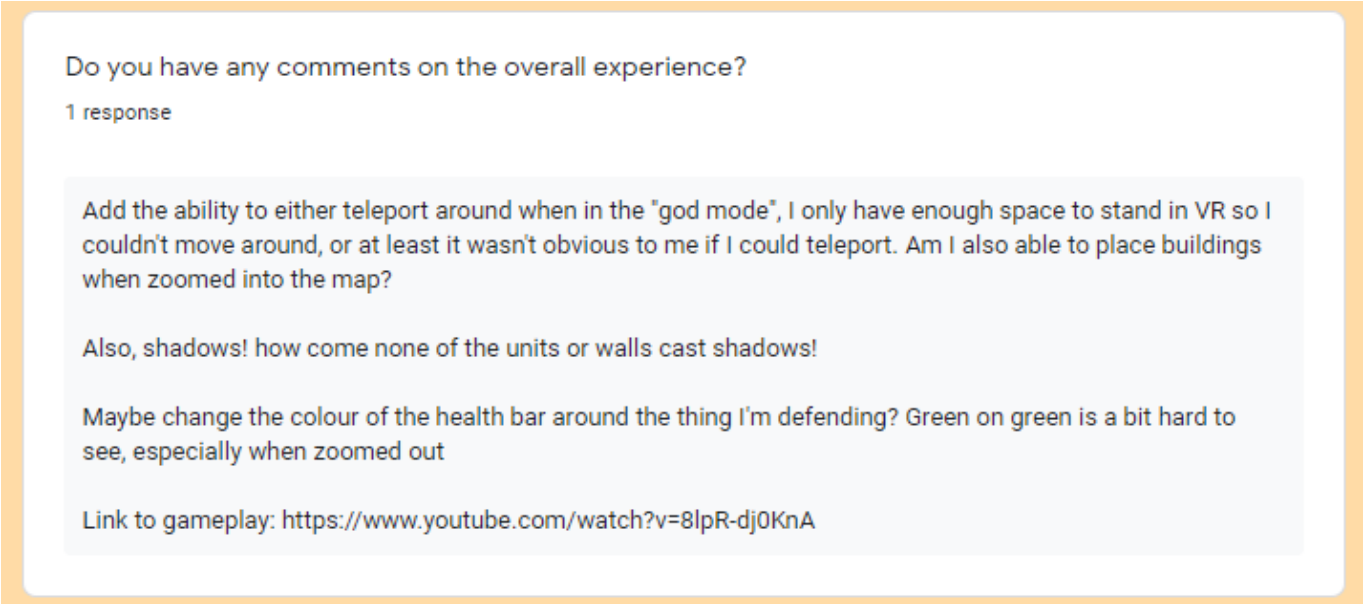


Figure 4.2 – *Other Comments*

The following table acts as an amalgamation of all feedback from this session. Although lacking in quantity, the suggestions are very helpful.

Easier representation of controls
Raycast blocked by walls
Round Changing music
Longer building time
Skip building time
AI stopped at one point
Shadows
Colour of health bar needs to be clearer

Session 2 – Consumer Perceptions: 30/03/2020

This was the non-VR alternative to usability testing, which had many more responses. The purpose of this testing was to gage some idea on consumer’s ideas about the game and it’s branding. This could then be used to inform the future release of the project.

In order to start the questionnaire, a few questions were asked regarding their perception of the title and VR games in general. There were a few participants which did not have experience with gaming, so the questions were very general, and when required, further described.

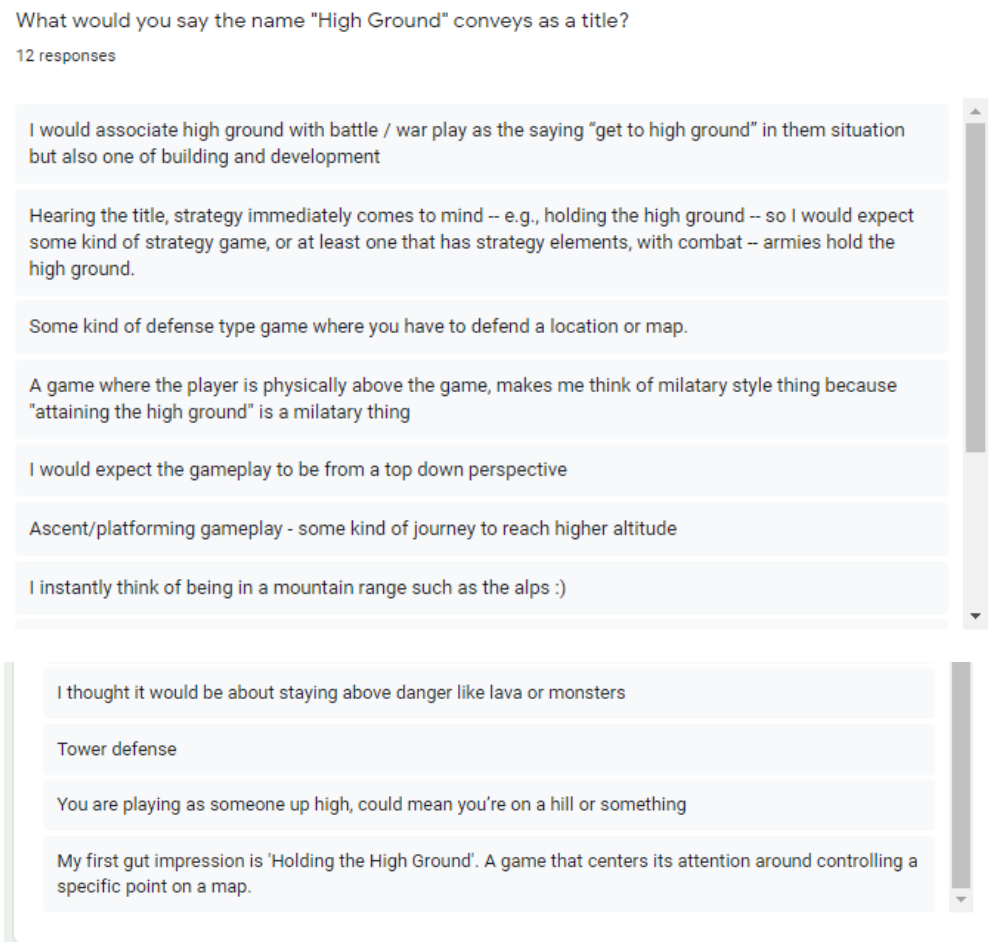


Figure 5.1 – *Perceptions on the name.*

The purpose of this question was to ask whether the name represented the project. From the responses, it’s clear that the name is appropriate, and consumers have correct assumptions for what the game title conveys.

The next question moves onto whether the game should have “VR” in the title. A link to the viveport store was given so that people could explore other titles in the VR market. The consensus was quite even, with slightly more towards no. Considering the nature of this project, being for university and by a single developer, I personally think that having VR in the title helps with understanding the project.

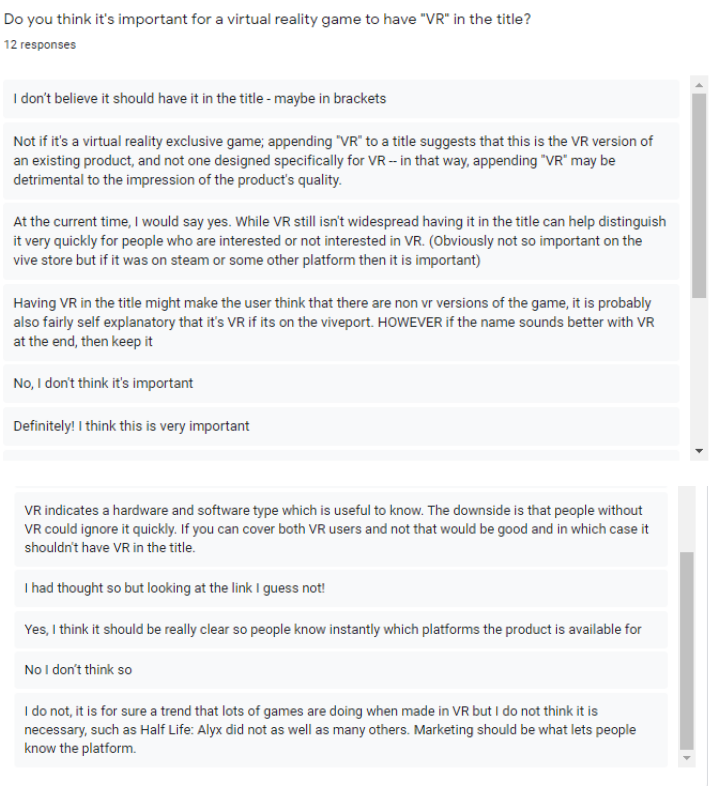


Figure 5.2 – *Perceptions on 'VR' in the title*

The final question in this introduction section was based on introducing people to the genre of tower defence. This was easy for those already familiar, but a Wikipedia link was provided to aid those who weren't. This question was more of an activity than to receive feedback, so the responses will not be posted here for analysis

The next section starts with a gameplay video, which people are encouraged to watch. A few questions follow, which aim to gather opinions on the project after a small gameplay clip. It also helps with understanding how much people can learn about the project just from a video of it. This started off with the following:

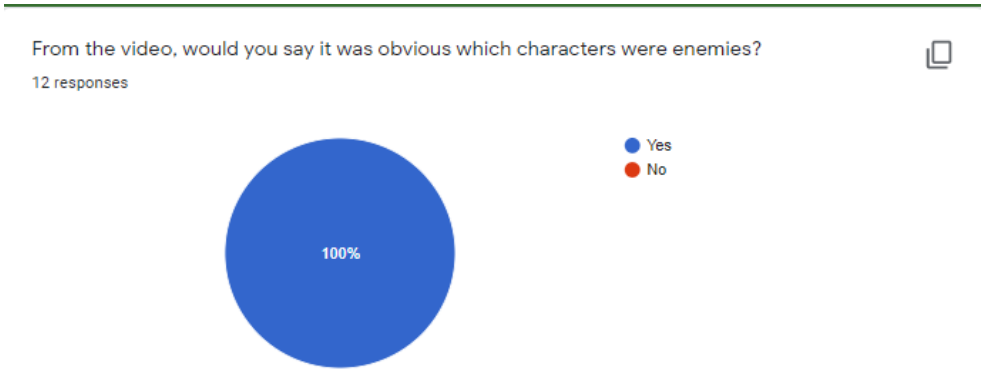


Figure 5.3 – *"would you say it was obvious which characters were enemies?"*

And then a similar question revolving on how the player interacted with the game. All participants got this correct, showing that the game isn't too complicated to understand from a video. This is beneficial to the concept of a trailer accurately representing the gameplay.

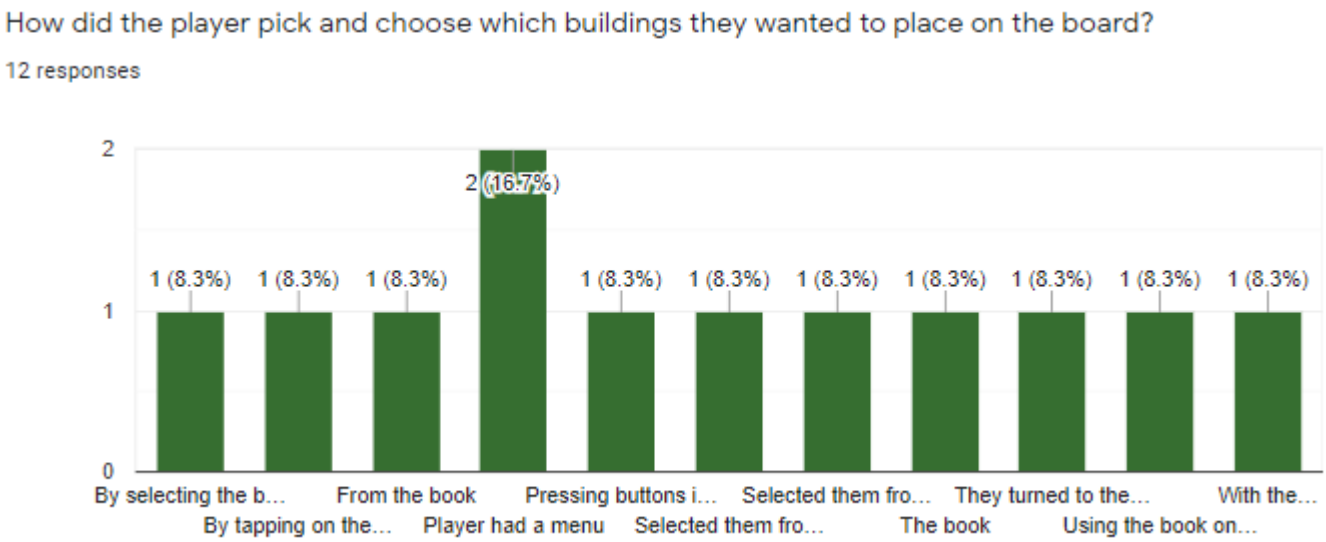


Figure 5.4 – *Asking how the player chose which buildings they wanted to place*

The following two questions were based upon marketing and branding of the project. These were purely to gain an idea on colour scheme and logo preferences.

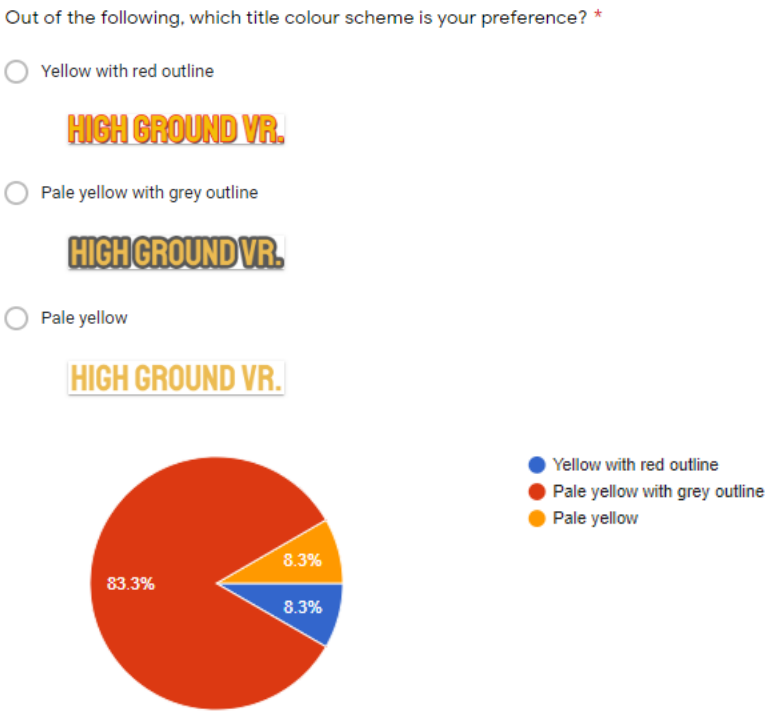


Figure 5.5 – *Banner colour scheme preferences*

Which do you think should be the feature icon for the game? *

This will be the face of desktop icons, app icons and other thumbnail images.

Units

Barracks

Mine

Defensive Walls

Which do you think should be the feature icon for the game?

12 responses

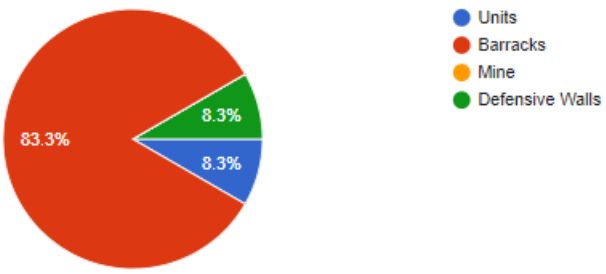


Figure 5.6 – *Feature Icon Preferences*

The final question asked was on the opinions on the potential price of the game. With another link to the viveport store, participants were asked to have a look through projects on the store

that may look of similar calibre to High Ground VR. Then, they were given a multiple-choice question, shown below.

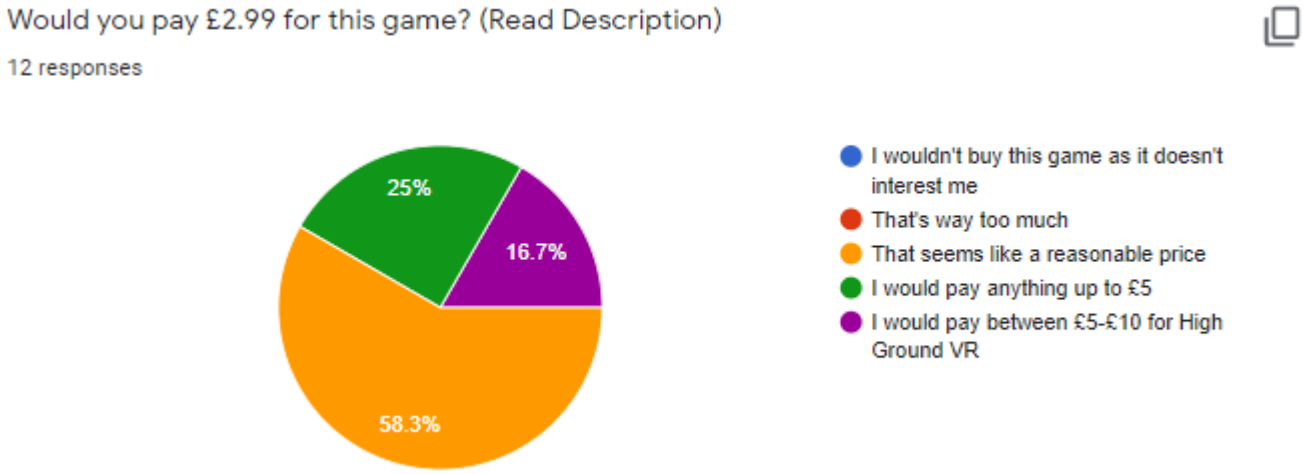


Figure 5.7 – *Opinions on price of High Ground VR*

With an intended price of £2.99, it's very positive that all responses said that they would purchase it for anything up to £5, with some even saying up to £10.

Although the feedback from this session isn't too helpful at this stage of development, it will be used to guide the future release of the project. The opinions on the pricing and branding have been taken on board and will be actioned before release. The fact that all participants, both non-gamers and gamers, understood the core concepts of the project shows that a trailer for the project will do the gameplay justice, which is perfect for a game of this scale.

Appendix 11: Market Research

Market Analysis	Release Date	Monthly Downloads	Total Downloads	Monthly Money Earnt (Ang Price * Monthly Downloads)	Total Money Earnt (Ang Price * Total Downloads)	Recent Activity (Is it doing badly or well?)	Strong Mid or weak	Popular Countries	Platform	How do they monetize?
Direct Competitors										
Castle Must Be Mine	July 2018	4	1,740.00	£ 49.56	£ 21,558.00	Top search result	Mid	UK, USA, Australia	SteamVR	Price on Steam - £12.39
Alchemist Defender	November 2017	1	210.00	£ 7.19	£ 1,509.90	No Activity	Weak	UK, USA, Australia	SteamVR	Price on Steam - £7.19
Hex Defense	Februar	2	112.00	£ 11.58	£ 648.48	No Activity	Weak	UK, USA,	SteamVR	Price on

	y 2019							Australia		Steam - £5.79
Similar Competitors										
Kingdom Rush	January 2014	138	15,442.00	£ 964.62	£ 107,939.58	Doing very well over many platforms	Strong	UK, USA, Europe	Mobile and Desktop	Price on Steam and Microtransactions - £5.79
Sanctum	April 2011	31	15,365.00	£ 222.89	£ 110,474.35	No Activity due to sequel	Mid	UK, USA, Europe	Desktop	Price on Steam - £5.79
Orcs Must Die	October 2011	52	8,337.00	£ 373.88	£ 59,943.03	No Activity due to sequel	Mid	UK, USA, Europe	Desktop	Price on Steam - £5.79

Customer Base	Target Audience (Type of player)	Average Reviews	Notes (Improvements from the community)	Age Range	Game Genre / Tags	Game theme (hardcore, casual or midcore)	Expected experience level	Expected Features (Controls and Mechanics)
Direct Competitors								
Castle Must Be Mine	VR Casual	Very Positive	Not enough tower variety, Poor controls, too Expensive	13 +	Strategy, Casual, Indie, VR	Casual	Basic VR Knowledge	Defend a tower, waves of enemies, build towers
Alchemist Defender	VR Casual	Mixed	No ability to upgrade towers, Only one form of locomotion	13 +	Strategy, Action, VR	Casual	Basic VR Knowledge	Defend a tower, waves of enemies, build towers. cast spells
Hex Defense	VR Casual	Mixed	Runs poorly, Bad tracking	16 +	Strategy, Casual, War, VR	Casual	Basic VR Knowledge	Defend a tower, waves of enemies, build towers
Similar Competitors								
Kingdom Rush	Mobile Casual	Overwhelmingly Positive	Bad balancing, Free mobile version preferred	13 +	Tower Defence, Strategy	Casual	Basic Tower Defence	Defend a tower, waves of enemies, build towers
Sanctum	Casual	Very Positive	Bad balancing	16 +	Tower Defence, Strategy, FPS	Casual	Basic Tower Defence	Defend a tower, waves of enemies, build towers, FPS
Orcs Must Die	Casual	Overwhelmingly Positive	Can't get refunds on buildings, Awkward controls	13 +	Tower Defence, Action, Strategy	Casual	Basic Tower Defence	Defend a tower, waves of enemies, build towers, ORCS

Customer Opinions	Similar Apps	Bugs	What players love about the game	Reasons players hate the game	Requested Features
Direct Competitors					
Castle Must Be Mine	Kingdom Rush, Blooms TD 6	Poorly optimised on Valve Index, Teleporting not accurate	Enjoy throwing fireballs at units, great VR experience for quick gameplay, Simplistic pick up and go gameplay	Not much sense of progress, Too expensive for repetitive gameplay	More tower types
Alchemist Defender	Bone Works, Sanctum, Castle Must Be Mine	Poorly optimised	Beautiful Environment, Good Balancing, Fun FPS combat,	VR controls awkward, No upgrade system for towers	Bosses, more levels, changing environment, different classes, More progression
Hex Defense	Plague Inc, Totally Accurate Battle Sim, Castle Must Be Mine	Broken Control Schemes	Great graphics, easy to get into, achievements, great mechanics	No difficulty setting so it's too easy, would be better as non-VR, rounds were too long	Difficulty options, players wish they could also attack (FPS Mode)
Similar Competitors					
Kingdom Rush	Taur, Plague Inc, GemCraft	Bad Balancing	Very addictive, relaxing, great on mobile, it's like an old flash game	It's too expensive on steam store, becomes predictable, doesn't tell players what they're doing wrong	More upgrade features, More content for their money
Sanctum	Rainbow Six Siege, Depp Rock Galactic	Bad Balancing, laggy	Very creative, many solutions to different levels, fun resource management	Gets boring quickly for the money, very predictable, easy on co-op impossible on solo	Better enemy designs, better graphics
Orcs Must Die	Bloons TD 6, Totally Accurate Battle Simulator	Bad Balancing, Black screen on certain devices	Great storyline + character growth, good art style, fun achievement hunting	Can't refund buildings, awkward control scheme, it’s broken on specific devices	Refundable traps,

Appendix 12: Software and Hardware

Game Engine and Programming Language

This project will be utilizing the Unity game engine developed by Unity Technologies. Unity is the most commonly used 3D game engine amongst developers and is therefore the most widely supported. Unity supports three scripting languages, C#, Unity Script and Boo. This project uses C#, as it’s the most widely documented programming language that is compatible

with the Unity Engine. It also supports features which Unity Script does not, such as events and delegates.

Integrated Development Environment (IDE)

Visual Studio 2019 was the IDE (Integrated Development Environment) chosen for this project. [4]An IDE provides an environment which is more feature-rich than a standard editor. They usually include such features as compilers and code completion tools, hence making them appropriate for the development of larger projects. Visual Studio also provides helpful implementation with Unity, including autocomplete and other debugging tools that aid in development within the game engine.

Asset Creation Software

Asset Creation Software is broken down into 3 subcategories based on the nature of the asset itself. Each tool was used with valid licensing regarding creating commercial works, hence is appropriate for this project and its release.

3D Modelling Software

There were multiple pieces of software used for 3D modelling during the development of this project. Neither of which require any additional licensing for commercial releases of products created using them.

MagicaVoxel was used to produce most of the 3D assets. It was chosen as it's a free lightweight voxel art editor that can quickly and easily create 3D models for the purpose of rapid prototyping and development.[5] It also includes an advanced GPU based interactive path tracing renderer that can be used to produce promotional images and sprites of models created within the program. Blender was the software chosen for more advanced and detailed 3D modelling. Blender is an open source alternative to Autodesk's 3ds Max and is also well documented with a variety of online tutorials and guides.[6] Blender is also becoming an industry standard as its open source approach appeals to smaller businesses and artists, this also makes it very accessible to student developers.

Sprites, UI Elements and Branding

Any assets that fall within the categories of sprites, UI or branding were created using Adobe Photoshop. Photoshop is a raster graphics editor developed and published by Adobe Inc, available on all major operating systems.[7] It provides powerful editing tools for a variety of

---

[4] https://visualstudio.microsoft.com/ : Visual Studio is an Integrated Development Environment (IDE) developed by Microsoft to develop GUI (Graphical User Interface), console, Web applications, web apps, mobile apps, cloud, and web services, etc.

[5] https://ephtracy.github.io/ : MagicaVoxel: A free lightweight 8-bit voxel art editor and GPU based interactive path tracing renderer. Learn more at this link.

[6] https://www.blender.org/ : Blender is the free and open source 3D creation suite. It supports the entirety of the 3D pipeline. Learn more at this link.

[7] https://www.photoshop.com/en : Adobe Photoshop is a raster graphics editor developed and published by Adobe Inc. for Windows and macOS. Learn more at this link.

purposes and is the industry standard software in its area. It's easily the most widely supported photo editing software and is hence the most accessible piece of software of its calibre.

Audio Assets

For any required editing of music or sound effects, Audacity was used. Like Blender, Audacity is an open source approach to a more expensive solution. It is a multi-track audio editor available for Windows, Mac OSX and Linux.[8] Although it's seemingly complicated interface, Audacity was very accessible for the minimal audio editing required during this project.

Virtual Reality Headset

The Virtual Reality headset used for the development of High Ground VR was the HTC Vive. This was chosen because of its availability at low prices second-hand, and hence its accessibility to the market is much greater than alternative headsets. It's a piece of hardware that's readily available within the University and is quick and easy to set up in the labs and at home. It also is a great entry level virtual reality headset and has a simple input method. As there aren't any advanced features, like finger tracking or advanced AR, players aren't expecting this project to have mechanics to cater to them.

[8] https://www.audacityteam.org/ : Audacity is an easy-to-use, multi-track audio editor and recorder for Windows, Mac OS X, GNU/Linux and other operating systems. Learn more at this link.