Turris  
Portfolio  
Computer Science  
COMP208: Group Software Project

Oliver Legg – Kieran Baker – Thomas Coupe – Daniel Taylor – Alessandro Wang

**Contents**

[Project report 1](#_Toc8215474)

[Abstract 1](#_Toc8215475)

[Design 2](#_Toc8215476)

[Summary 2](#_Toc8215477)

[Business rules 2](#_Toc8215478)

[Use Cases 3](#_Toc8215479)

[Use Case Descriptions 4](#_Toc8215480)

[Process and states of the game 11](#_Toc8215481)

[User interface 12](#_Toc8215482)

[Functional Description / Specification 16](#_Toc8215483)

[Introduction 16](#_Toc8215484)

[Gameplay 16](#_Toc8215485)

[Turrets / Towers 16](#_Toc8215486)

[Different Ammo Types 16](#_Toc8215487)

[Enemies 16](#_Toc8215488)

[Maps 17](#_Toc8215489)

[Difficulties 17](#_Toc8215490)

[Shop 17](#_Toc8215491)

[Currency 17](#_Toc8215492)

[Pseudocode 18](#_Toc8215493)

[Turret 18](#_Toc8215494)

[Enemy 21](#_Toc8215495)

[Player 22](#_Toc8215496)

[Table Structures 24](#_Toc8215497)

[Data Dictionary 26](#_Toc8215498)

[Class Diagram 28](#_Toc8215499)

[Message Sequence Diagram 29](#_Toc8215500)

[Original Gantt Chart 30](#_Toc8215501)

[Updated Gantt Chart 31](#_Toc8215502)

[Testing 32](#_Toc8215503)

[Introduction 32](#_Toc8215504)

[Main Menu 32](#_Toc8215505)

[Game Play 35](#_Toc8215506)

[Playing 35](#_Toc8215507)

[GUI that pauses the game 39](#_Toc8215508)

[Sound 41](#_Toc8215509)

# Project report

## Abstract

Turris is a tile-based tower defence strategy game where the user defends a path against waves of soldiers. To stop the soldiers the player must place defensive turrets that will attack and kill nearby enemies. When an enemy is killed, it will drop rewards which can be used to buy and upgrade turrets. As the game progresses further, the enemies will increase in numbers and strength. If an enemy reaches the end of the path, the player loses a life, when the player is out of lives it's game over.

# Design

## Summary

Our group intends to make a tower defence game that can entertain users for hours. The overall goal of the game is to:

* Entertain
* Challenge
* Intrigue
* Escape
* Compete

To complete these goals, we would like to give an experience that the user wants from a game. Not everyone would want to compete – therefore we’d like to give them choices and options within the game. To tackle these problems, we have some main objectives to achieve this:

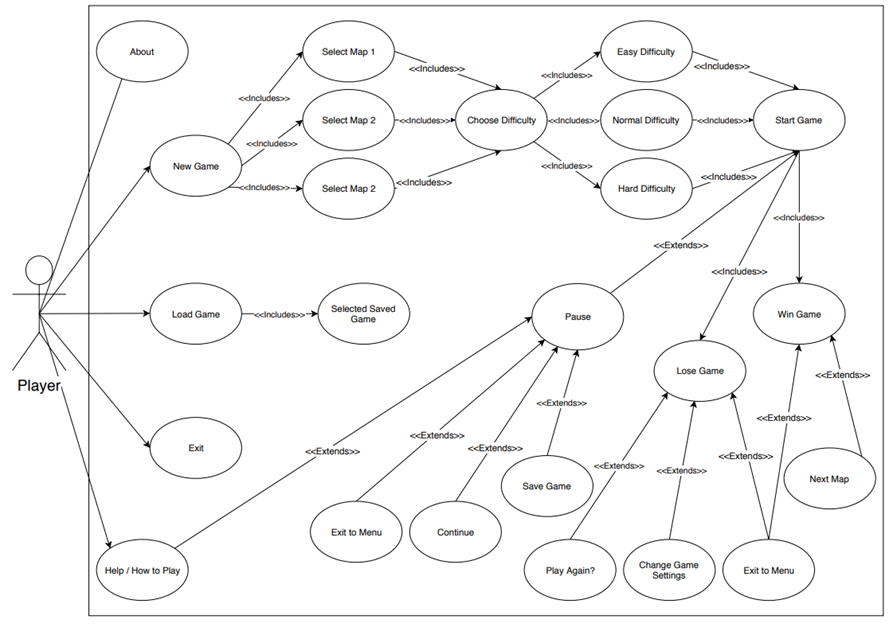
* Create a satisfying experience for the user to interact with the game and impress them with gameplay quality, graphics and sound effects.
* Create an environment in the game that will challenge the user’s ability. The user can challenge themselves by settings the difficulty from easy, medium and hard.
* Intrigue the user on what happens when you win the game or play a level with their own strategy.
* Create an environment of the game and overall experience that will captivate the user and help them to escape.
* Create endless mode which allows the user to have a certain time that they survive for. This can be compared with another player’s score – this will bring a competitive aspect of the game.

We want the user to feel good when playing the game and want them to feel rewarded for playing well. To do this, we have we do have to difficulty aspect and loss aspect, if you can only win at the game, it will feel undeserved and meaningless.

## Business rules

* To reduce storage space, each user will be limited to one save file which will store data for a single game. If the user wishes to save a different game, the old file must be deleted.
* The save file must only contain the relevant data that is required for the save such as the remaining lives, wave number and turret locations, any extra information should be omitted to reduce file size on the disk.
* In the event of an error occurring while the user is attempting to save or load their game any data for the current save file should not be modified or deleted.
* The name that the player enters must be at least three characters long and no longer than 15 characters. The name entered must only contain letters and numbered and must not have any special characters in it.
* Only 5000 enemies can be spawned as a time as this would be too computationally intensive to process for our users. This would also consume a lot of memory in the computer, eventually slowing it down.
* As there will be a grid-based placement system, there will be a certain number of towers to be placed on the map. If the screen in 800x600 and each tile is 100x100, there will be 48 tiles (8x6 = 48) and a lot of these will be taken up by paths. Therefore, there is a constraint on how many towers you can place on a map at a time.

## Use Cases



## Use Case Descriptions

|  |  |
| --- | --- |
| **ID** | **UC 1** |
| **Name** | New Game |
| **Description** | Allow the player to start a new game |
| **Actor** | Player |
| **Pre-condition** | Game is launched and player ready to play |
| **Post-condition** | Map options displayed |
| **Main flow** | 1. System displays available maps |
| **Includes** | UC 5 (Select Map) |
| **Extensions** | UC 17 (Play Again?) |

|  |  |
| --- | --- |
| **ID** | **UC 2** |
| **Name** | Load Game |
| **Description** | Allow the player to start a saved game |
| **Actor** | Player |
| **Pre-condition** | The player Saved previously a started game |
| **Post-condition** | Map options displayed |
| **Main flow** | 1. Player select desired saved game 2. Player loads in the desired saved game |
| **Includes** | Selected Saved Game |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 3** |
| **Name** | Help/How to play |
| **Description** | The system will display a rules and mechanics of the game  It can be also selected when you pause during a game |
| **Actor** | Player |
| **Pre-condition** | Game is launched |
| **Post-condition** | Help and guides displayed |
| **Main flow** | 1.Player get initial idea of how the game work and how to play it |
| **Includes** | None |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 4** |
| **Name** | Exit |
| **Description** | Allow the player to shut down the game and back to the desktop |
| **Actor** | Player |
| **Pre-condition** | Game is launched and plyer not willing to play |
| **Post-condition** | Game is exited |
| **Main flow** | 1. Player stops desiring to play 2. Game exited |
| **Includes** | None |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 5** |
| **Name** | Select Map |
| **Description** | Allow the player to choose a desired map |
| **Actor** | Player |
| **Pre-condition** | Game is launched and player ready to play |
| **Post-condition** | Map options displayed |
| **Main flow** | 1. Player given option to select map |
| **Includes** | UC 6 (Choose difficulty) |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 6** |
| **Name** | Choose Difficulty |
| **Description** | Allow the player to pick the desired difficulty in a new game |
| **Actor** | Player |
| **Pre-condition** | Player has already chosen the map |
| **Post-condition** | Map difficulties displayed |
| **Main flow** | 1. Player given option to select map difficulty |
| **Includes** | UC 7, UC8, UC 9 |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 7** |
| **Name** | Easy Difficulty |
| **Description** | Player is playing game on easy |
| **Actor** | Player |
| **Pre-condition** | Player has chosen the difficulty |
| **Post-condition** | Map ready to start |
| **Main flow** | 1. Player selected difficulty 2. Map ready to start |
| **Includes** | UC 10 (Start Game) |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 8** |
| **Name** | Normal Difficulty |
| **Description** | Player is playing game on normal |
| **Actor** | Player |
| **Pre-condition** | Player has chosen the difficulty |
| **Post-condition** | Map ready to start |
| **Main flow** | 1. Player selected difficulty 2. Map ready to start |
| **Includes** | UC 10 (Start Game) |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 9** |
| **Name** | Hard Difficulty |
| **Description** | Player is playing game on hard |
| **Actor** | Player |
| **Pre-condition** | Player has chosen the difficulty |
| **Post-condition** | Map ready to start |
| **Main flow** | 1. Player selected difficulty 2. Map ready to start |
| **Includes** | UC 10 (Start Game) |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 10** |
| **Name** | Start Game |
| **Description** | After player settled desired options the game will finally start |
| **Actor** | Player |
| **Pre-condition** | Player decided map and difficulty |
| **Post-condition** | Map started |
| **Main flow** | 1. Player selected options 2. Map started |
| **Includes** | UC 12, UC 13 (Lose Game, Win Game) |
| **Extensions** | UC 11 (Pause) |

|  |  |
| --- | --- |
| **ID** | **UC 11** |
| **Name** | Pause |
| **Description** | When the player is playing the game and want a break or want to exit or save the game |
| **Actor** | Player |
| **Pre-condition** | Player is currently playing the game |
| **Post-condition** | Game paused |
| **Main flow** | 1. Player decides to take a break 2. Game is paused |
| **Includes** | None |
| **Extensions** | UC3, UC 14, UC 15, UC 16 (Help/How to play, Exit to menu, Continue, Save game) |

|  |  |
| --- | --- |
| **ID** | **UC 12** |
| **Name** | Lose Game |
| **Description** | Player lose the game after reaching a limit |
| **Actor** | Player |
| **Pre-condition** | Player is playing the game |
| **Post-condition** | Player is not able to defend a certain amount of enemy |
| **Main flow** | 1. Player is playing 2. Player cannot defend a certain amount of enemy 3. Player loses the game |
| **Includes** | None |
| **Extensions** | UC 17, UC 14 (Play Again?, Exit to menu) |

|  |  |
| --- | --- |
| **ID** | **UC 13** |
| **Name** | Win Game |
| **Description** | Player win the game after defending a certain number of rounds |
| **Actor** | Player |
| **Pre-condition** | Player is playing the game |
| **Post-condition** | After several rounds if player health is more than zero |
| **Main flow** | 1. Player is playing 2. Player kills a needed amount of enemies 3. Player win the game |
| **Includes** | None |
| **Extensions** | UC 14, UC 18 (Exit to menu, Next Map) |

|  |  |
| --- | --- |
| **ID** | **UC 14** |
| **Name** | Exit to menu |
| **Description** | A option for the player to exit from the current map and back to the menu, game datas will be lost if player decides or forgets to save, it will also display in situation where the player lose the game of wins it |
| **Actor** | Player |
| **Pre-condition** | Player is currently playing in a map |
| **Post-condition** | Player wants to go back the menu |
| **Main flow** | 1. Player currently in a map 2. Player pause and want to save the game or the player won or lost the game 3. Exit to main menu option taken |
| **Includes** | None |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 15** |
| **Name** | Continue |
| **Description** | Continue is an option when the game is paused by the player |
| **Actor** | Player |
| **Pre-condition** | The game is paused |
| **Post-condition** | Player decides to continue |
| **Main flow** | 1. Game paused 2. Game unpaused |
| **Includes** | None |
| **Extensions** | None |

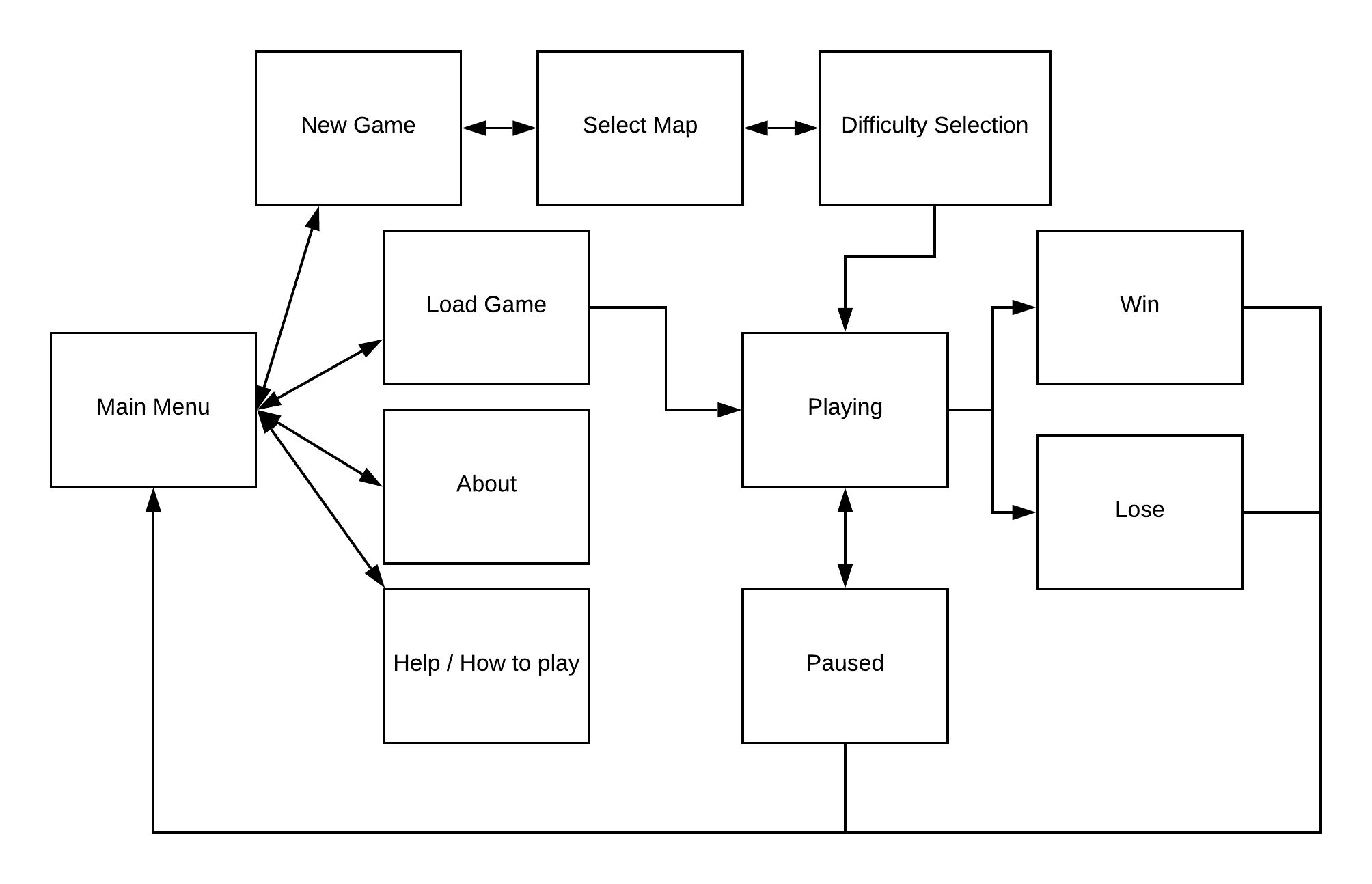
|  |  |
| --- | --- |
| **ID** | **UC 16** |
| **Name** | Save Game |
| **Description** | Save game is an option displayed when the game is paused, and it is a very important feature to preserve the player’s progress |
| **Actor** | Player |
| **Pre-condition** | Game paused |
| **Post-condition** | Player decides to save his progress |
| **Main flow** | 1. Game paused 2. Progress saved |
| **Includes** | None |
| **Extensions** | UC 19 (selected saved game) |

|  |  |
| --- | --- |
| **ID** | **UC 17** |
| **Name** | Play Again? |
| **Description** | Option displayed when the player lose the game |
| **Actor** | Player |
| **Pre-condition** | Player lost the game |
| **Post-condition** | Player decides to play or no to play again |
| **Main flow** | 1. Player lose the game 2. System displays play again message 3. Player play again |
| **Includes** | None |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | UC 18 |
| **Name** | Next Map |
| **Description** | Option displayed when the player wins the game |
| **Actor** | Player |
| **Pre-condition** | Player wins in current map |
| **Post-condition** | Player decides to play the next map |
| **Main flow** | 1. Player wins in current map 2. Next map option displayed 3. Player enters in the next map |
| **Includes** | None |
| **Extensions** | None |

|  |  |
| --- | --- |
| **ID** | **UC 19** |
| **Name** | Selected saved game |
| **Description** | Player in the menu can choose a saved progress through the load game option |
| **Actor** | Player |
| **Pre-condition** | Player has saved at least one progress previously |
| **Post-condition** | Map ready to start |
| **Main flow** | 1. Player select saved game 2. Player loads in saved game 3. Map starts |
| **Includes** | None |
| **Extensions** | None |

## Process and states of the game



The above diagram shows the states in which the games will be in. You can see that when you access the main menu, you can create a new game, load the game, go to the about screen and final the help screen. Once you are on all these states, you can go back to the main menu. However, if you proceed onto another state, for example, the playing state, you are not able to navigate directly back to the main menu without pausing the game, winning the game or losing the game.

## User interface

|  |  |
| --- | --- |
| This is the main menu state. When you start the game. This is the first thing you’ll see. As you can see you will have access to:   * New Games * Loading old games * Document on how to play the game * About screen * Settings menu |  |
| The how to play menu will go into detail about not only how to play the game, but it will also show you how to be much better. You can revert to the main menu by pressing the bottom left button. |  |
| The settings menu will have technical settings and options that you might want to select. A darker coloured button will mean that setting is active, whereas a light button will mean that setting is off or inactive. For example, in the picture you will see that fullscreen is on and vsync is off. The music is set to 33% as you can see from the slider and the sfx is set to 78%. This is clarified by the label on the button too. |  |
| This is the about menu which shows what the game is about and who developed the game. |  |
| The player selects the difficulty of the game here. They’re also able to revert to the main menu. |  |
| The selection of the map is here, they have 3 choices (and might be given more in the future), there is also an endless mode which selects a special map. |  |
| This is the playing state of the game. This shows the path that the enemies will go down in on the field. The enemies will start from the top left and finish at the bottom left. Anywhere where the grass is on the grid, you will be able to place turrets. As you can see, turrets are on the field. |  |
| This is the playing state of the game again, but this is taken with the GUI minimised. When you press on the button, it will create the menu shown on the next page. |  |
| This is the GUI maximised. The button functionality works like so:   * X (closes the GUI) * T1 (places turret tier 1) * T2 (places turret tier 2) * T3 (places turret tier 3) * £ (sells) turrets * Pause (pauses the game) |  |
| The game is paused here, and you can choose to resume the game, save the game and quit to the main menu. Movements of the enemies and bullets will stop as well as the shooting of turrets. Time will also not increase in endless mode when paused. The pause state can be initiated by pressing the pause button on the previous state, or by pressing the ‘Esc’ key. |  |

## Functional Description / Specification

### Introduction

A Functional Description allows us to explain all the details of what we are creating to someone who has no idea or prior information regarding the project. This means that we must be very accurate when writing the specific details for our game. I have considered the feedback that we received from Sebastian Coope which was to go into a lot more detail when we are talking about our game, this makes sense as we want to be able to explain to the outside audience as clearly as possible.

### Gameplay

Turris is the name of the game we have come up with, we have decided that it is going to be a tower defense game with turrets / towers, different ammo types, multiple enemy types, multiple maps and 3 difficulties.

The game will include waves of enemies that approach from one side of the map and attempt to make it to the other, if a certain number of enemies reach the other side then they will defeat the player. To stop these waves of enemies, the player can place turrets that are going to be firing / attacking the enemies to stop them from making it to the end. Depending on the difficulty and or map, this can be made harder for the player.

### Turrets / Towers

We do not want to go over the top with the details for the game, in fact we want to be able to make it relatively simple for us to create and then make changes as we go along to improve the game. Therefore, we are setting the default number of different towers to 3, this will be useful to us when we are first trying to understand how our game is going to work.

The towers will each have a different cost, the more powerful the tower the higher the cost. It is simple to understand, however saving money to survive the higher rounds is going to be a player-based decision that could allow them to pass certain waves, it is a game of skill after all.

The towers will have a certain range, we will use grid references within a list, to calculate the distance from the towers and the enemies, again the higher cost of the tower the larger range it will have, this will give the player a large advantage as they proceed through the waves.

There are going to be multiple enemies, so placing multiple towers around the map will be a common strategy for the player due to the fact they will have to defeat many enemies at different areas of the map, obviously a single tower will not be able to fire across the whole map, so it is down to the player to decide where these towers are placed (this will again be done with grid references so that we can snap the towers into certain places, we might have harder levels where placements in some parts of the map are out of bounds).

### Different Ammo Types

Within the game there are also going to be different ammo types, this means that the towers can have different types of ammo that the player can either purchase or select if they have already purchased some. Obviously, this is going to be a strategic move by the player, as they are going to be the ones to decide whether they will need to purchase the ammo from the shop within the next wave of enemies. They will be able to select the different ammo types for each individual tower, they will be limited to how many times they can fire that specific ammo by how much of it they purchased. There are going to be 3 different ammo types, these will have different properties such as increased damage or can hit multiple targets.

### Enemies

We have to have different enemies within our game to make it a challenge and more interesting for the player, I have also stated that we do not want to go over the top and make our game too complicated as we are first building it, so again we are going to have 3 main enemies that are going to spawn in the game. Usually, the game will start off as easy on any difficulty, with the easiest enemies being the in the first few waves, however as the game progresses, we are going to introduce the other enemies that are going to be a concern to the player. The enemies will have different amount of health points that require you to hit over time to defeat them, the further the player gets in the game, the more enemies are spawned, as well as more of the higher tier / harder enemies will spawn and give the player a challenge. These enemies might have different speeds as well, meaning they will travel through the map faster, putting pressure on the player for them to be quick and defeat the enemies before they reach the end goal.

### Maps

We have discussed the maps in detail, as we believe they are going to be the foundation of the game and how we are going to control all the tower placements, enemy pathing and other key factors that are going to create the game. We are designing the maps very basic at first to get an understanding of how we are going to move the enemies in terms of flags and grid references, however we are going to have 3 maps that will all get harder when the player completes one and moves to the next. I mentioned previously that we might have areas where a player can’t place a turret, such as a river or a lake in which there is no way a player can place a turret as it is meant to be positioned on land. (We might include towers that can only be placed on water in the future to give the game a little more detail)

The maps are going to be based around a certain path, the design is up to us on how the path changes throughout the map, as there are going to be twists and turns that the enemies follow. The enemies will always follow that specific path so that the user knows where he or she can or can’t place the turrets to fight the enemies. The path will change on each map and might be longer or shorter which will add to the difficulty of the game.

### Difficulties

The difficulties are self-explanatory, we are going to have Easy, Normal and Hard. In terms of how that changes the game, the difficulties will increase the health of the enemies and possibly the speed. The main thing that will change is the health of the player, as the player loses once a certain number of enemies pass through the end goal, so the amount it takes for the player to lose will lower the harder the difficulty is. Therefore, to explain a little better, Easy mode will have 150 health, Normal will have 100 and Hard will have 50, this makes the game more intense for the player and allows us to increase the difficulty of the game automatically without making and drastic changes. We can also increase the number of enemies that spawn within the levels when we change the difficulty so that it is not always the same. Hard mode might even have a bonus final round in which you have to defeat an overall boss to win the game.

We believe that adding multiple difficulties gives the player a sense of accomplishment once they have completed a map, they will then feel like they are progressing through the game as they go through the levels and increase the difficulty if they choose.

### Shop

The shop is a very important part of the game, it is the main area of player interaction outside of the map, we have to make sure that the shop is integrated perfectly into the game, so that there are no issues when the player is attempting to spend his points / currency to get a different tower or ammunition.

The shop must be very easy to access as the player will need to purchase towers as the waves of enemies are progressing, the shop will have towers and ammunition to purchase, we might update this later so that we have upgrades although that will be once the main aspects of the game are implemented.

### Currency

The currency is closely related to the shop as it is going to be how we purchase towers and ammunition, the currency will be gathered by defeating enemies, a certain amount of currency will be given to the player once an enemy has been defeated. The harder the enemy is to defeat the more currency the player will receive once it has been defeated. The player should be able to spend their currency in the shop as the game / waves are progressing, meaning we must make sure that the shop is always accessible for the player so that they can spend their earned currency.

## Pseudocode

### Turret

ABSTRACT CLASS Turret(x, y)

FILE texture

FLOAT rateOfFire

INTEGER range

INTEGER x

INTEGER y

INTEGER level

INTEGER MAX\_UPGRADE = 3

INTEGER damage

FUNCTION ABSTRACT upgrade()

FUNCTION getX()

RETURN x

END

FUNCTION getY()

RETURN y

END

FUNCTION fire\_at\_enemy(enemy)

AIM\_AT(enemy)

FIRE()

WAIT(rateOfFire)

END

END

CLASS Turret\_I EXTENDS Turret

INTEGER range = 500

INTEGER level = 1

INTEGER damage = 10

FLOAT rateOfFire = 0.5

FUNCTION create(x, y)

PUSH\_TO\_ABSTRACT\_CLASS(x, y, "Turret\_I.png")

END

FUNCTION getX()

RETURN x

END

FUNCTION getY()

RETURN y

END

FUNCTION fire\_at\_enemy(enemy)

AIM\_AT(enemy)

FIRE()

WAIT(rateOfFire)

END

FUNCTION upgrade()

IF level < MAX\_LEVEL THEN

rateOfFire = rateOfFire - 0.2

range = range + 100

damage = damage + 2

level = level + 1

END

END

END

CLASS Turret\_II EXTENDS Turret

INTEGER range = 600

INTEGER level = 1

INTEGER damage = 20

FLOAT rateOfFire = 0.5

FUNCTION create(x, y)

PUSH\_TO\_ABSTRACT\_CLASS(x, y, "Turret\_II.png")

END

FUNCTION getX()

RETURN x

END

FUNCTION getY()

RETURN y

END

FUNCTION fire\_at\_enemy(enemy)

AIM\_AT(enemy)

FIRE()

WAIT(rateOfFire)

END

FUNCTION upgrade()

IF level < MAX\_LEVEL THEN

rateOfFire = rateOfFire - 0.2

range = range + 120

damage = damage + 3

level = level + 1

END

END

END

CLASS Turret\_III EXTENDS Turret

INTEGER range = 300

INTEGER level = 1

INTEGER damage = 70

FLOAT rateOfFire = 2.5

FUNCTION create(x, y)

PUSH\_TO\_ABSTRACT\_CLASS(x, y, "Turret\_III.png")

END

FUNCTION getX()

RETURN x

END

FUNCTION getY()

RETURN y

END

FUNCTION fire\_at\_enemy(enemy)

AIM\_AT(enemy)

FIRE()

WAIT(rateOfFire)

END

FUNCTION upgrade()

IF level < MAX\_LEVEL THEN

rateOfFire = rateOfFire - 0.2

range = range + 50

damage = damage + 7

level = level + 1

END

END

END

### Enemy

ABSTRACT CLASS enemy\_1(INTEGER x, INTEGER y)

FILE texture

FLOAT rateOfFire

INTEGER range

INTEGER x

INTEGER y

INTEGER speed

INTEGER damage

FUNCTION ABSTRACT upgrade()

FUNCTION getX()

RETURN x

END

FUNCTION getY()

RETURN y

END

FUNCTION move\_to(INTEGER pos\_x, INTEGER pos\_y)

IF pos\_x != x AND x THEN

MOVE FORWARD

ELSE

RETURN FALSE

END

END

END

CLASS enemy\_2(INTEGER x, INTEGER y) EXTENDS enemy\_1

INTEGER range = 25

INTEGER x = 0.1

INTEGER y = 0.4

INTEGER speed = 0.35

INTEGER damage = 25

FUNCTION ABSTRACT upgrade()

FUNCTION getX()

RETURN x

END

FUNCTION getY()

RETURN y

END

FUNCTION move\_to(INTEGER pos\_x, INTEGER pos\_y)

IF pos\_x != x AND x THEN

+pos\_x

ELSE

+pos\_y

END

FUNCTION takeDamage()

IF arrowhit

health - 100

ELSE

health = health

END

END

END

CLASS enemy\_3(INTEGER x, INTEGER y) EXTENDS enemy\_1

INTEGER range = 35

INTEGER x = 0.1

INTEGER y = 0.4

INTEGER speed = 0.50

INTEGER damage = 35

FUNCTION ABSTRACT upgrade()

FUNCTION getX()

RETURN x

END

FUNCTION getY()

RETURN y

END

FUNCTION move\_to(INTEGER pos\_x, INTEGER pos\_y)

IF pos\_x != x AND x THEN

+pos\_x

ELSE

+pos\_y

END

FUNCTION takeDamage()

IF arrowhit

health - 100

ELSE

health = health

END

END

END

### Player

CLASS player\_shop

INTEGER itemPrice

STRING itemName

INTEGER stock

INTEGER playerCurrency

FUNCTION purchaseItem()

IF itemPurchased

playerCurrency - itemPrice

stock - 1

player +itemName

ELSE

playerCurrency = playerCurrency

END

FUNCTION sellItem()

IF itemSold

playerCurrency + itemPrice / 3

END

END

Class player

INTEGER playerCurrency

INTEGER playerHealth

STRING playerName

INTEGER playerLevel

INTEGER ammoCount

STRING ammoType

INTEGER towerAvailable

FUNCTION enemyKilled()

IF enemy\_health = 0

enemyKilled +1

playerCurrency +50

END

FUNCTION enemyCrossed()

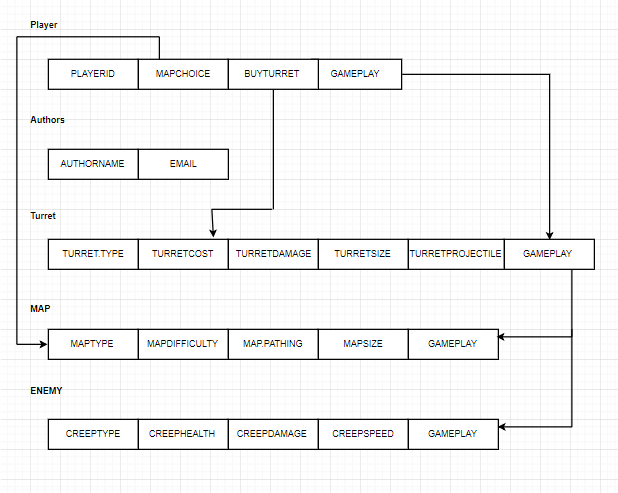
IF enemyAtEnd

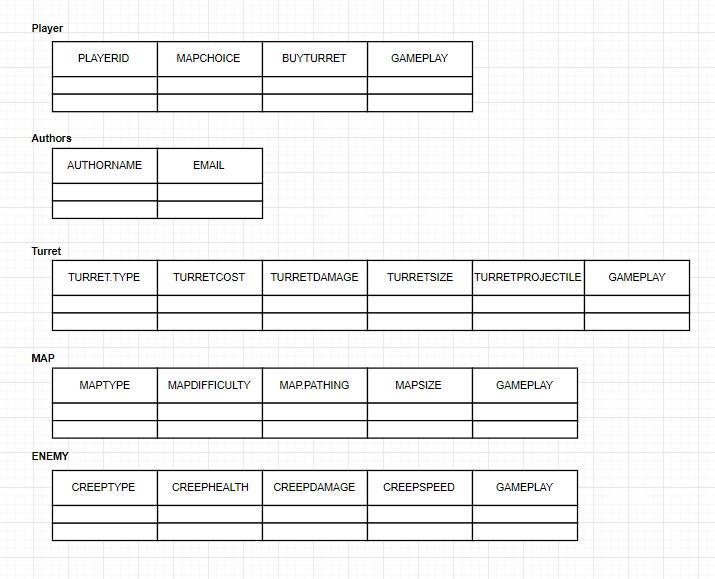
playerHealth - 1

END

END

## Table Structures

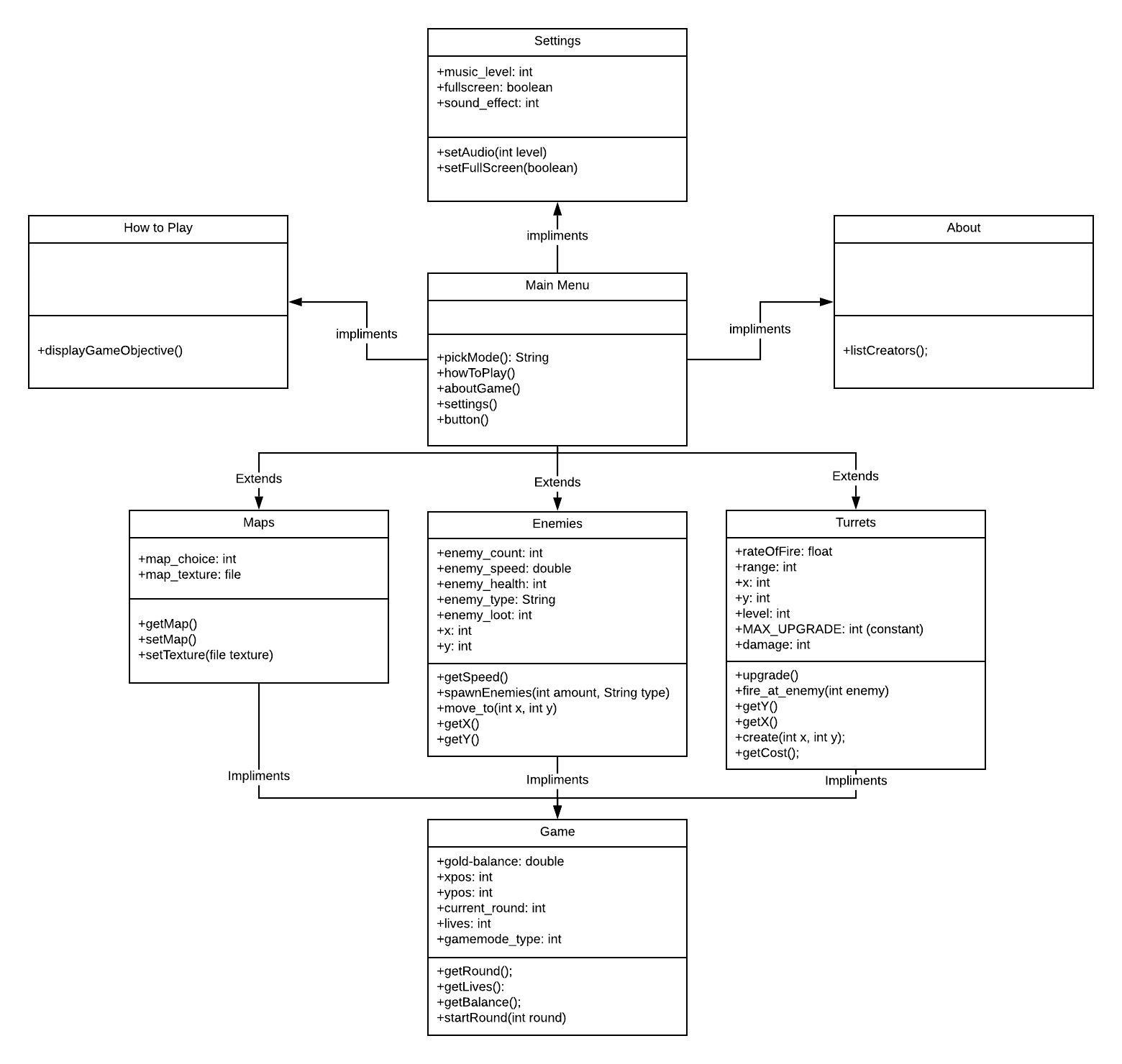




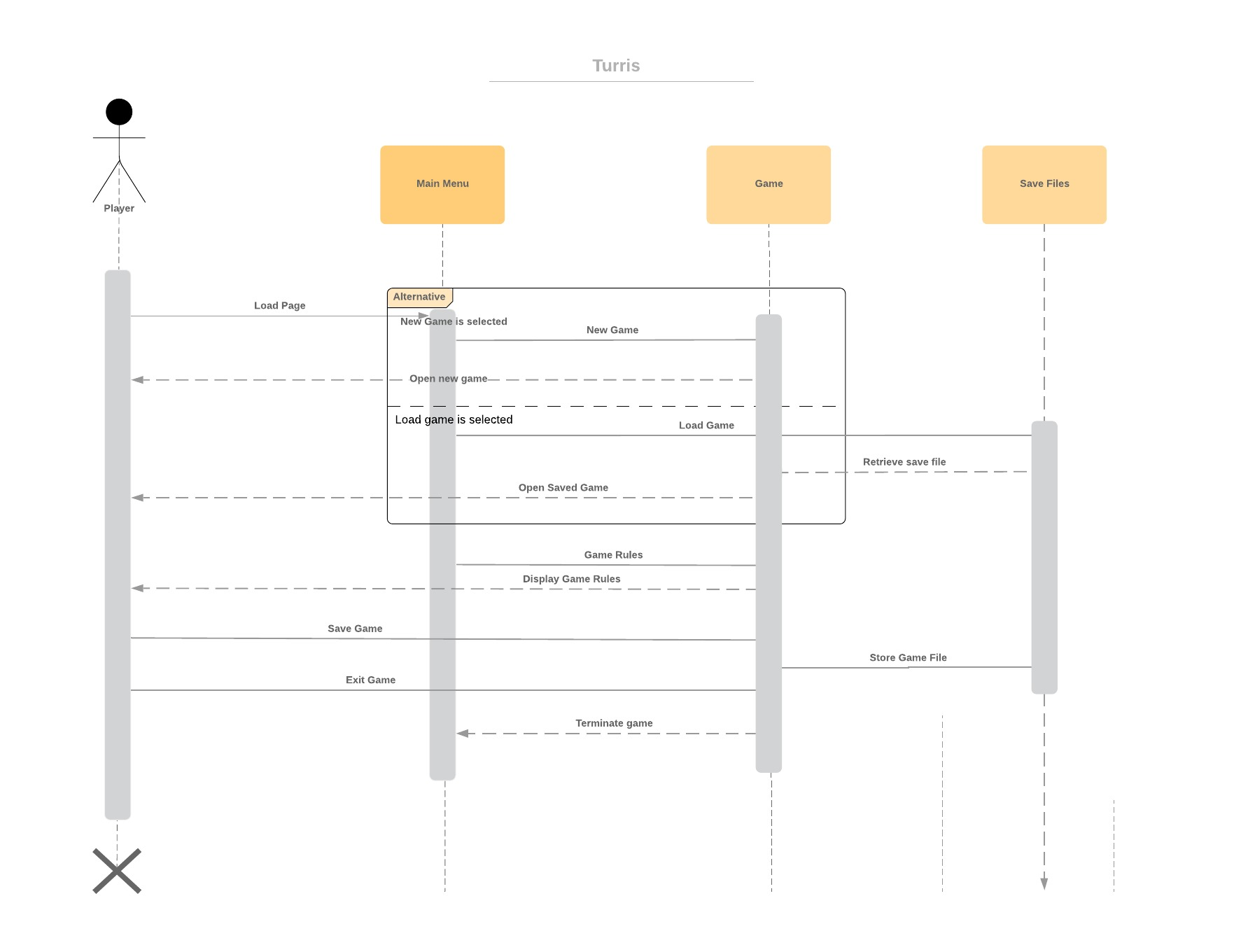
## Data Dictionary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data Dictionary - Turris | | | | |
| Name | Data Type | Length | Scope | Purpose |
| player\_lives | Integer | 1 - 20 | public | This value holds the remaining number of lives the player has. Once this value is at 0 the game will be over. |
| gold\_balance | double | 0.00 - 20000.00 | public | When the player kills an enemy, they receive a gold reward to buy and upgrade towers. This value represents the total amount of gold the player has. |
| xPosition | Integer | Width of frame | public | xPosition represents the location of where the turret is along the width of the frame. |
| yPosition | Integer | Height of frame | public | yPosition represents the location of where the turret is along height of the frame. |
| Turret\_x | Integer | 0-800 | public | This value represents where the player is location along the x axis giving the x position of the player on the frame. |
| Turret\_y | Integer | 0-600 | public | This value represents where the player is location along the y axis giving the position of the player on the frame. |
| Current\_round | Integer | 1 – 30 or if on endless mode its unlimited | public | This value represents the current round that the player is on, as the round value increases so does the enemy count. |
| Enemy\_count | Integer | unlimited | public | The enemy count represents how many enemies are to be spawned in on that round. As there is an endless mode there is no fixed maximum number of enemies that can be spawned. |
| enemy\_speed | Double | 0.01 – 5.00 | public | This value represents the speed that the enemies move along the path. |
| Turret1\_price | Double | 200.00 | Public | This value represents the total cost for Turret 1 |
| Turret2\_price | Double | 500.00 | Public | This value represents the total cost for Turret 2 |
| Turret3\_price | double | 1000.00 | public | This value represents the total cost for Turret 3 |
| Turret1\_level | Integer | 1 – 5 | Public | This value represents the current tower level for tower 1 |
| Turret2\_level | Integer | 1 – 5 | Public | This value represents the current tower level for tower 2 |
| Turret3\_level | Integer | 1 – 5 | public | This value represents the current tower level for tower 3 |
| turret\_damage | Integer | 0 - 100 | public | The integer value represents the amount of damage a tower does to an enemy per each shot from the tower. |
| map\_choice | Integer | 1 - 3 | public | The map selected is represented as an integer value. As there is three maps the user can choose between 1, 2 and 3 to select which map they want to play. |
| enemy\_health | Integer | 0 - 100 | public | This value represents how much help each of the enemies have. |
| enemy\_type | String | “Gladiator”, “Warrior”,  “Fighter” | public | The enemy type can be chosen between three types of enemies. These enemies are defined by Strings. |
| turret\_range | Integer | Amount of blocks the turret can reach | public | This value represents how many blocks the tower projectiles can reach from their position. |
| Enemy\_loot | Integer | The amount of gold an enemy drops when killed | public | This value represents how much gold the enemy will when it is |
| gamemode\_type | Integer | 1 - 2 | public | This value represents which game mode has been selected. |

## Class Diagram



## Message Sequence Diagram

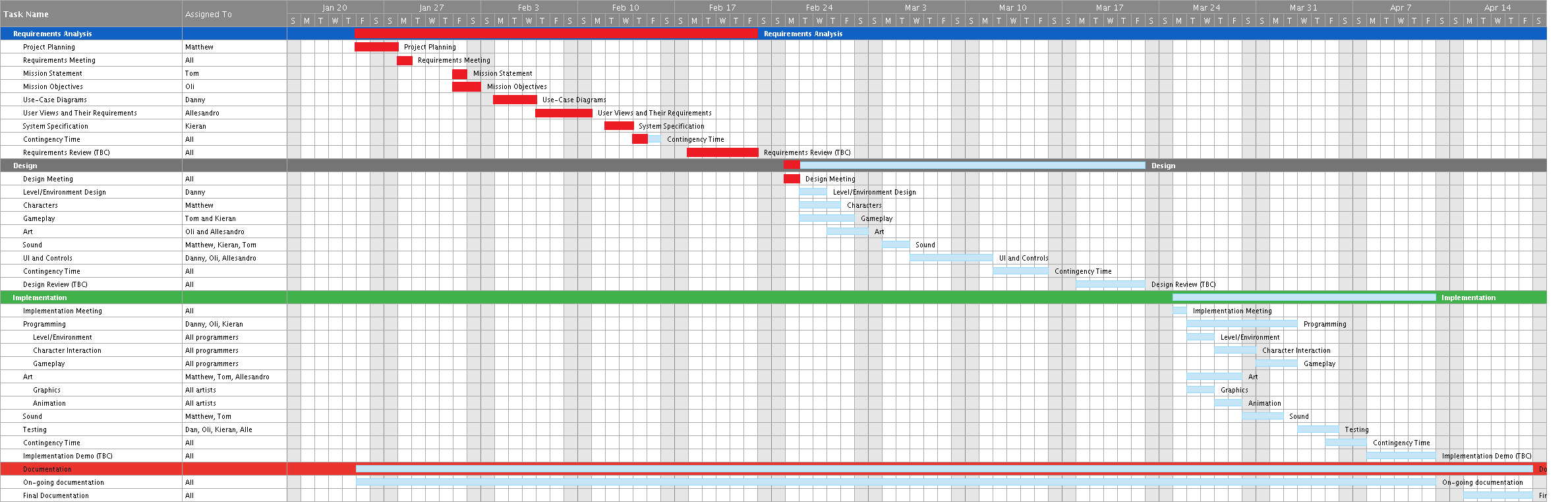


## Original Gantt Chart

## Updated Gantt Chart

Up till now, we have completed every task within the time constraints set out by our original plan. This includes all tasks associated with the requirements analysis. We had our review of our requirements document and have been given valuable feedback. Currently we are on track to completing our design document. The review has already been decided to take place on 22nd March.

We have not made any major changes to the Gantt chart as when we had our design meeting, every task we discussed fulfilled the plan. We have indicated in red what tasks have already been completed. You may notice we did not use all our contingency time for requirements analysis. This is because we worked efficiently and didn’t come across too many hurdle (this has been indicated on the Gantt Chart below).

****

# Testing

## Introduction

We have tested the game to see if the functionality of the game works. Something we checked a lot is to see if elements loaded correctly and how the user’s interactions with the input got the correct output. For example, I tested the fullscreen button. When the user presses the button, the user should be put in fullscreen. If the user is in fullscreen mode, they should be put into windowed mode. The button should change graphically depending if the user’s window is fullscreen too. Depending on a test’s performance, it will be colour coded like so:

|  |  |
| --- | --- |
| Test passed |  |
| Test failed – fixed |  |
| Test failed – unable to fix |  |

## Main Menu

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test No** | **Test Name** | **Description** | **Data Input** | **Expected Outcome** | **Actual Outcome** | **Remedial Action** |
| **1** | Main menu appears | Main menu displaying all graphics | - | Main menu appears with the background displayed and clouds moving in the background correctly | As expected | - |
| **2** | All Buttons highlight | Testing if the buttons can be interacted with graphically | mouse\_x mouse\_y | Buttons can be highlighted when hovering over it with the mouse | As expected | - |
| **3** | New game button | Testing to see if the new game button clicks | mouse\_x mouse\_y  mouse\_leftb | Button is pressed and user is taken to the map selection state | As expected | - |
| **4** | Load game button | Testing to see if the load game button clicks | mouse\_x mouse\_y  mouse\_leftb | Button is pressed and user is taken to the loaded save | As expected | - |
| **5** | Settings button | Testing to see if the settings button clicks | mouse\_x mouse\_y  mouse\_leftb | Button is pressed and the user is taken to the settings menu | As expected | - |
| **6** | How to play button | Testing to see if the how to play button clicks | mouse\_x mouse\_y  mouse\_leftb | How to play button is pressed and the documentation is then found online on the Turris website | As expected | - |
| **7** | About button | Testing to see if the about button clicks | mouse\_x mouse\_y  mouse\_leftb | The about button is pressed and then the user is taken to the about state | As expected | - |
| **8** | Back button | Testing to see if the back button clicks. The same back button is used on the map selection, settings and about state. If you press the button, it should take you back to the main menu state. | mouse\_x mouse\_y  mouse\_leftb | When you hover over the button, it should also change colour | As expected | - |
| **9** | Exit button | Testing to see if the exit button clicks | mouse\_x mouse\_y  mouse\_leftb | When you press the exit button, it should close the program. | As expected | - |
| **10** | Map selection screen | Testing to see if the map selection screen works | - | The map selection screen should display the main menu backgrounds, the 3 maps to choose from and the ‘Select map’ header at the top of the page | As expected | - |
| **11** | Map Selection Highlighting maps | Test to see if interacting with the maps tells you anything | mouse\_x mouse\_y | Hovering you mouse over the maps should change the ‘Select map’ header to the map name to either ‘Grassy Greens’, ‘Sand Dunes’ or ‘Tricky Track’. There should also be a border around the map with appearing buttons over the map image that you can then click on. | As expected | - |
| **12** | Standard mode | Clicking on maps on standard mode | mouse\_x mouse\_y  mouse\_leftb | Clicking on maps on standard mode should take you to the map you specified on that standard mode. | As expected | - |
| **13** | Continuous mode | Clicking on maps on continuous mode | mouse\_x mouse\_y  mouse\_leftb | Clicking on maps on continuous should take you to the map you specified on that standard mode. | As expected | - |
| **14** | Settings screen | Settings screen should display all buttons and sliders | - | Clicking on the settings button should display all the buttons and sliders | As expected | - |
| **15** | Settings buttons | The buttons (apart from back) checked if interactable | mouse\_x mouse\_y | If the user hovers the mouse over the Fullscreen and Toggle mute button, the background of the mute button should dim to seem interactable | The buttons didn’t dim when you hover the mouse over it. | Fixed the issue and uploaded to github. The colours now change to the following: hovering colour when button is on is darker green; hovering colour when button is off is lighter grey. |
| **16** | Settings sliders | Moved the sound effects and music volume sliders on the settings window | mouse\_x mouse\_y  mouse\_leftb | Moving the sliders should scale from numbers 0-100 | As expected | - |
| **17** | Settings Volume | Moving the music and sound effects slider will scale the volume of the game | mouse\_x mouse\_y  mouse\_leftb | Moving the music and sound effects will change the volume of the music ranging from 0-100. I know if this works by hearing the volume of the music and clicking on buttons plays | As expected | - |
| **18** | Settings fullscreen button | Fullscreen button toggles fullscreen | mouse\_x mouse\_y  mouse\_leftb | If you press the fullscreen button and the game is in windowed mode, it should make the game fullscreen. If you press the fullscreen button and the game is in fullscreen mode, it should put the game in windowed mode | As expected | - |
| **19** | Settings mute button | Mute button mutes sound | mouse\_x mouse\_y  mouse\_leftb | Settings button disables all sound when pressed and re-enables sound when pressed again | As expected | - |
| **20** | About screen | About screen should show and draw the correct details and buttons | - | When switching to the about state, the game displays the Turris creators and two buttons: GitHub and Website | As expected | - |
| **21** | About screen button interactions | Test to see if the buttons are interacted with the mouse | mouse\_x mouse\_y | When hovering over github and website button with the mouse, the buttons display a paler colour to display that they can be interacted with or clicked | As expected | - |
| **22** | About screen button press | Test to see if the functions of the buttons perform the correct task. | mouse\_x mouse\_y  mouse\_leftb | When clicking the buttons with the mouse, | As expected | - |

## Game Play

### Playing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test No** | **Test Name** | **Description** | **Data Input** | **Expected Outcome** | **Actual Outcome** | **Remedial Action** |
| **1** | Grassy Greens | Tiles load in correctly from the level\_1.csv file | Mouse\_X  Mouse\_y  Mouse\_leftb | Tiles display correctly and the enemy takes the correct path from the beginning of the map to the end. I will also check to see if turrets should place where they’re supposed to place and don’t place where they’re not supposed to place | As expected | - |
| **2** | Sand Dunes | Tiles load in correctly from the level\_2.csv file | Mouse\_X  Mouse\_y  Mouse\_leftb | Tiles display correctly and the enemy takes the correct path from the beginning of the map to the end. I will also check to see if turrets should place where they’re supposed to place and don’t place where they’re not supposed to place | As expected | - |
| **3** | Tricky Track | Tiles load in correctly from the level\_3.csv file | Mouse\_X  Mouse\_y  Mouse\_leftb | Tiles display correctly and the enemy takes the correct path from the beginning of the map to the end. I will also check to see if turrets should place where they’re supposed to place and don’t place where they’re not supposed to place | As expected | - |
| **4** | Standard Mode | Check if game ends on winning | Mouse\_X  Mouse\_y  Mouse\_leftb | The game ends when you reach the level’s max level. And says congratulations, you win. | As expected | - |
| **5** | Continuous Mode | Check if game doesn’t end | Mouse\_X  Mouse\_y  Mouse\_leftb | The game doesn’t end when surpassing max level. Waves come through as specified by the wave multiplier algorithm in the code. The wave templates should follow the | As expected | - |
| **6** | Enemies | Enemy behaviour | - | When starting the game, the enemies follow the paths that were laid out. The spawns are delayed as specified in the round file. | As expected | - |
| **7** | Rounds end | Rounds should end | - | When all enemies of the round are dead, the round ends | As expected | - |
| **8** | Enemy death rewards | Enemy death provides you with gold | - | Enemy death provides you with the correct amount of gold | As expected | - |
| **9** | Losing lives | When enemies pass to the end off the track and go off the screen, you should lose a life | - | Enemy surpassing the track should remove a life. | As expected | - |
| **10** | Enemies health | Enemies get shot with arrows | - | When enemies are shot with arrows, they lose health according to the damage of the arrow | As expected | - |
| **11** | Enemy speed | Enemies move at the correct speed | - | Enemies move at the correct speed that they’re given, relative to the speed modifier of the game | As expected | - |
| **12** | Button Interaction | Buttons should dim when hovering over it with the mouse | Mouse\_X  Mouse\_Y | Hovering the mouse over buttons dims the button to show that you can interact with it. | As expected | - |
| **13** | Start Button | Starts the round | Mouse\_X  Mouse\_Y  Mouse\_leftb | Pressing the start button starts the game | As expected | - |
| **14** | Pause Button | Pauses the game | Mouse\_X  Mouse\_y  Mouse\_leftb | Pressing the pause button pauses the game and sets the button text to ‘Resume’ | As expected | - |
| **15** | Resume Button | Resumes the game | Mouse\_X  Mouse\_y  Mouse\_leftb | Pressing the resume button unpauses the game and sets the button text to ‘Pause’ | As expected | - |
| **16** | Speed Modifier Button | Changes the pace of the game. | Mouse\_X  Mouse\_y  Mouse\_leftb | 1x is regular speed 2x is double regular speed 4x is four times regular spd. | As expected | - |
| **17** | GUI box for player’s progress | Shows a box for coins, lives, the round they’re on and the cost of the turrets | - | The gui box shows | As expected | - |
| **18** | Players progress stats | Shows the stats: coins, lives, the round they’re on and the cost of the turrets | - | Shows the player stats in the gui box | As expected | - |
| **19** | Towers selection | Towers can only be selected if you have the correct amount of money | Mouse\_X  Mouse\_y  Mouse\_leftb | A tower can be selected if you have the correct amount of money. The game should now display a grid of where to place the turret if you can afford. Otherwise, the turret buttons should appear red and they should be unable to be interacted with. | As expected | - |
| **20** | Tower Cost | Testing to see if the turrets display the cost and displays them correctly | - | Hovering over the tower button should display the cost (in green if you can afford them and in red if you can’t afford then). | As expected | - |
| **21** | Turret range placing | Turret should display turret range when placing | Mouse\_X  Mouse\_y  Mouse\_leftb | The turret shows the correct turret radius when placing | As expected | - |
| **22** | Turret range | Turret should display turret range when interacting with the specified turret | Mouse\_X  Mouse\_Y | The turret shows the correct turret radius when interacting with the mouse | As expected | - |
| **23** | Placing turret | Testing to see if placing the turret works and place onto the grid | Mouse\_X  Mouse\_y  Mouse\_leftb | Placing the turret onto the grid and displays correctly and able to interact with it. Placing the turret should scale with upgrades too | As expected | - |
| **24** | Turret Functionality | Turret stats | - | Depending on what turret you place, it will have the correct textures, damages and damages that increase with upgrade. The turrets display the correct texture and only fire arrows when they have an arrow ready and they can only fire them an enemy if they’re in their radius. | As expected | - |
| **25** | Turret upgrades | Correct turret functionality on upgrade | Mouse\_X  Mouse\_Y Mouse\_rightb | Upgrading the turret, the turret displays the correct upgrade cost. This also increases the turret damage and other stats depending on the turret on upgrade. Finally, the upgrade only happens if you have the right amount of coins. This would then take that amount of coins away from the player. | As expected | - |
| **26** | Arrows | Arrow functionality | - | Arrows fire in the correct direction at the correct speed and when collided, they do the correct amount of damage to the enemy. The arrows also display the correct texture | As expected | - |
| **27** | Selling | Towers can be sold | Mouse\_X  Mouse\_Y  Mouse\_rightb | Towers are the only cells that can be sold. When the tower is sold, you are given a small portion of money returned. The tile then returns to its previous tile. The cursor displays red on tiles that can’t be sold and green on tiles that can be. The portion of money returned is clearly shown above the tile to be sold in yellow text. The radius of the turret should also be shown | As expected | - |
| **28** | Save Game button | Save game button should save the game to the start of the round | Mouse\_X  Mouse\_Y  Mouse\_leftb | Save game button should save the game to the start of the round. Saving the game should also pull up a menu and pause the game. | As expected | - |
| **29** | Settings button | Settings button should open the settings | Mouse\_X  Mouse\_Y  Mouse\_leftb | The settings button opens the settings menu and pauses the game. | As expected | - |
| **30** | Quit game button | Quit game button should open a menu | Mouse\_X  Mouse\_Y  Mouse\_leftb | Pressing the quit button opens a quitting menu and pauses the game | As expected | - |
| **31** | Finishing the game | Winning or losing the game in standard mode | Mouse\_X  Mouse\_Y  Mouse\_leftb | Winning or losing the game in standard mode shows you these correct statistics: difficulty, round reached, lives, total revenue, total kills, arrows fired, buildings built, buildings upgraded. Winning the game will display ‘Congratulations! You win!’, losing the game will show, ‘Unlucky you lose’. There’s a new button that can be interacted with and when clicked, it exits the game to the main menu. If you lose the game, the save of that game is removed as you have lost | As expected | - |
| **32** | GUI button | Testing if the GUI navigation panel works | Mouse\_X  Mouse\_Y  Mouse\_leftb | The GUI navigation panel button opens the navigation panel and changes direction to open and close the menu. The navigation panel should display all buttons. | As expected | - |

### GUI that pauses the game

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test No** | **Test Name** | **Description** | **Data Input** | **Expected Outcome** | **Actual Outcome** | **Remedial Action** |
| **1** | Settings screen | Settings screen should display all buttons and sliders | - | Clicking on the settings button should display all the buttons and sliders | As expected | - |
| **2** | Settings sliders | Moved the sound effects and music volume sliders on the settings window | mouse\_x mouse\_y  mouse\_leftb | Moving the sliders should scale from numbers 0-100 | As expected | - |
| **3** | Settings Volume | Moving the music and sound effects slider will scale the volume of the game | mouse\_x mouse\_y  mouse\_leftb | Moving the music and sound effects will change the volume of the music ranging from 0-100. I know if this works by hearing the volume of the music and clicking on buttons plays | As expected | - |
| **4** | Settings fullscreen button | Fullscreen button toggles fullscreen | mouse\_x mouse\_y  mouse\_leftb | If you press the fullscreen button and the game is in windowed mode, it should make the game fullscreen. If you press the fullscreen button and the game is in fullscreen mode, it should put the game in windowed mode | As expected | - |
| **5** | Settings mute button | Mute button mutes sound | mouse\_x mouse\_y  mouse\_leftb | Settings button disables all sound when pressed and re-enables sound when pressed again | As expected | - |
| **6** | Settings buttons | The buttons (apart from back) checked if interactable | mouse\_x mouse\_y | If the user hovers the mouse over the Fullscreen and Toggle mute button, the background of the mute button should dim to seem interactable | The buttons didn’t dim when you hover the mouse over it. | Fixed the issue and uploaded to github. The colours now change to the following: hovering colour when button is on is darker green; hovering colour when button is off is lighter grey. |
| **7** | Quit save | The quit button should display a menu when clicked | mouse\_x mouse\_y  mouse\_leftb | The quit button should display a menu when clicked and create text asking if you want to save your progress as quitting deletes your progress. The menu also creates three buttons, quit, continue and ‘save and quit’. The functionality of the buttons works and are completely interactable. | As expected | - |
| **8** | Save game | The save button should display a menu when clicked | mouse\_x mouse\_y  mouse\_leftb | The save button should display a menu when clicked. This should create some text which says, ‘Game saved’ and 2 buttons to quit and Continue. You can interact with these buttons and when you press quit, you quit to the main menu. When you press continue, you continue the game that you’re currently on. | As expected | - |

## Sound

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test No** | **Test Name** | **Description** | **Data Input** | **Expected Outcome** | **Actual Outcome** | **Remedial Action** |
| **1** | Volume | The volume of the sliders set in the settings changes the volume | Mouse\_x  Mouse\_y | The volume of the sliders set in the settings changes the volume | As expected | - |
| **2** | Music Main menu | Music on the main menu plays and stops in the correct state | - | The main menu music plays on the main menu and in the settings, on about and map selection correctly and at the correct volume | As expected | - |
| **3** | Music playing in the Playing state | Music plays on the playing state | - | The playing state music plays on the playing state and in the settings in the playing state and any other menu that appears within the playing states | As expected | - |
| **4** | Lose life | SFX | - | plays when you lose a life | As expected | - |
| **5** | Loss | SFX | - | plays when lose the game | As expected | - |
| **6** | Enemy dies | SFX | - | 1 of 2 plays when enemies die | As expected | - |
| **7** | Enemy hurt | SFX | - | 1 of 2 plays when enemies get hurt | As expected | - |
| **8** | Menu Click | SFX | - | Plays when you click something in the main menu | As expected | - |
| **9** | Round complete | SFX | - | Plays when a round is completed in the game | As expected | - |
| **10** | Turret place | SFX | - | plays when you place a turret | As expected | - |
| **11** | Turret shoot | SFX | - | Plays when a turret shoots an arrow | As expected | - |
| **12** | Turret upgrade | SFX | - | Plays when you upgrade a turret | As expected | - |
| **13** | Victory | SFX | - | Plays when you win the game | As expected | - |