1633018_EWANANFIELD_WSOA3003_EXAMREFLECTION

Reflection

Firstly I would like to insert a disclaimer about this game, Goblin explorer. During the last hours of it's creation of which I was on zero hours of sleep for the last 2 days unity decided to crash and corrupt my files causing me to lose what could be called 80% of my progress. As of such I was unable to submit the finished game as my last backup was a few days ago. Although this is unfortunate I will still be able to talk about what I did do before the loss and what changes I made from the initial game design plan. I hope this is adequate and I apologise for the disappointing gameplay.

The intended game

Goblin Explorer was set out to be a Top-down tower defence game where the player is able to explore their surroundings, fight monsters, collect loot, upgrade items and buildings, and protect their village. Although this was already a fair bit of an over scope, a lot of the mechanics individually were not hard to implement. It was more or less the balancing of the game that would take a large amount of time, but this will be discussed later in the document.

As a group we had a few design goals that were wanted to achieve as well as the methodology we were going to use. First of all we decided to use the scrum methodology as we wanted to be able to react to changes and problems as quickly as possible, This type of methodology allows for rapid prototyping and iteration, what this means is that we are able to quickly figure out problems and ways to fix them. Almost every new implementation into this game was tested to see how it interacted within the game environment and if it truly belonged. Our design goals for this game was that we wanted to create a responsive movement system that felt nice to use, we wanted to create a intuitive UI that clearly communicated assets (as in what did what), we wanted to create a reactive and challenging combat system that forced the player to use skill rather than blindly swing their sword at their opponent, and lastly we wanted to create a meaningful upgrade system. Although all of this together seems like a lot of work in terms of design goals, when made separately and then pieced together it really isn't a lot of work.

The gameplay of the game was also quite simple, but made up of lots of little individual parts. First of all the aim of the game was to survive the raids that were done by the goblins (don’t let the goblins destroy the town) but to do this the player had to venture out of the town and battle goblins in order to obtain materials that were used to upgrade the village itself (tower defence aspect of the game) and the player. This was to be regulated by day and night cycles, although this was almost completely controlled by the player. The player would start their day in the village where they are able to buy resources such as bullets and upgrade their equipment/buildings if they had the proper materials. Once the player decided they were done in the village they are able to leave by means of the main gate. Once they do this they enter into the next phase of the day (the fighting phase) where they use their equipment to defeat stray goblins and gather resources from then. Once the player either runs out of resources or decides they are done with this part of the game they can then return to the main gate where they are told once they enter the village they cannot return to the forest for that day. Once they are in the village they may once again spend materials or go to sleep which starts the next day (this is the way we force players to progress through days but not make them feel pressured by time). Every seven days a raid begins and the goblins now attack the village.

Every couple of days a Goblin king will appear while the player is in the forest, they must defeat this king. Once this happens the difficulty of all the goblins increases as they are enraged by their lost king. After the player has killed 3 goblin kings and the difficulty is at its max there will be 1 more raid upon the village and if the player survives this then the game is won.

If the player dies at any point in time but the village remains alive they are simply transported to their house and lose a portion of their resources. The day ticks over as well.

The materials that the player is able to obtain is Gold, Wood, Iron, and finally stone. All of these materials except Gold is dropped at random rates. Killing a Goblin king drops a large amount of these materials.

Development process

Throughout the development of this game it was indeed evident that we over scoped quite a fair amount. Not in terms of the mechanical ideas that we had but rather their implementation and fine tuning. Each idea we had for this game, be it the type of enemies or aspects such as how to control the day and night cycle, was fairly simple to code and implement on their own and some of these ideas were even extremely simple to string together but a lot of ideas had to either be abandoned or changed completely in order to make the game work (at least within the given time frame).

Once again I would like to apologise for the state of the game handed in because it doesn't show any of the work I actually did but let me speak with the experience I gained from making the game.

I started off focusing on the art aspect of the game as I wanted to be able to code with the proper visuals in place. With this being said I chose to use the pixel art style for my art as it is what I specialize in but I tried to be as quick as I possibly could with it because I know it isn't the most important aspect of this game. It didn't occur to me that I have never made a top down style game, I have only ever made side scrollers so while I was implementing the walk animations and the like I tried to make use of 'blend trees' but it didn't work with the animations I had so didn't use them in the end but rather just used simple bools withing the animator. Because of how movement works in a top down environment I failed to make smooth animations as I wasn't able to smoothly transition between animations, especially if the character is in their run cycle.

Once I finished all the Art for the game I focused on the level design. I made a fairly large level by the end of the project but due to the loss of progress I had to limit myself to a very small area outside the village that the player is able to explore. The village itself stayed the same where the player is able to interact with 4 important buildings. The Blacksmith, the Gunsmith, the general store, and finally the wall (all of these interactions were also lost). In the forest the player is able to encounter goblins and fight them for resources. After I finished the level design I moved onto the players storage system which would keep track of all of their items. With this I included a UI that showed the player what they had and to add things to their inventory they would just have to ‘pick up’ dropped materials or purchase resources from the village buildings.

Once this was done I focused on the drop rates of items from monsters as well as how many resources were required from the buildings to be upgraded as well as what the upgrades themselves did. With this I went through many different trials (this was most possibly the most intense testing period as this determined how important materials were as well as how easy they were to obtain). After 2 days of just testing this I came to a good balance of drop rate / resources required. Once this was done I finished up the upgrade system for the player as well as the buildings (all of this was also lost). Next I focused more on the enemies themselves. I incorporated a simple enemy UI that allowed the enemy to track the player and walk towards them. This was done by (in code) creating a circle around the enemy unit and if anything with the tag “Player” walked into this circle the enemies would begin to walk towards the player. There was a secondary smaller circle that picked up if the player was close to the enemy and from which direction, that if triggered would cause the enemy to attack. The enemies health and damage was variables that I was able to change which was important for when I Implemented the Goblin Kings.

Due to time constraints I decided to remove the tower defence aspect of the game as it was going to be too intense to work around. Instead I decided to make this game a pure goblin fighting game where the condition for victory was to defeat the last Goblin King. I made it so that the Goblin Kings spawned once every week (always after the third day of the week). This gave the player enough time to ‘kit’ up. The goblin Kings had the same system as the goblins, they just had more health and did more damage. Once they were incorporated I made sure that upon their death they increased the damage of all subsequent goblins whom spawned.

I also decided to remove the gun from the game as it was causing too many issues and was taking up too much of my time. Once everything was coded and in the game and the game was running, I finally worked on finishing the UI of the game or rather the HUD. I included the players current health, the health of the enemies (this would float above them as a bar), I neatened up the part that showed the resources owned, included proper menus for the stores as well as UI for all the items in the game. I also included a small mouse symbol on the bottom of the screen that showed what right clicking did vs left clicking (The right click didn’t do anything since I removed the gun). I made a main menu as well as a pause menu.

I did not include sounds and simple animations such as smoke coming out of a chimney as I was running out of time. Unfortunately as I was about to finish everything up by cleaning up a few errors I had floating around unity froze and when I tried to open up my game files they wouldn’t load and as such I had to go to a much earlier iteration which hardly included any of the work I had actually done and given the time constraints I now face I am simply unable to resurrect this dead game.

Critical Reflection

Upon presenting our game idea to the tutors we were told that we were over scoping of which I disagreed at the time (which might I add quite sourly we did not get feedback or marks from that presentation, this did not help in the re-scoping process). Through creation of the game I decided that we did over scope a fair amount. Luckily I started work on this game fairly early so I was able to do a fair amount before it was due but if I didn’t cut out the tower defence part of the game I most likely wouldn’t have finished in time (not that I did anyway, ironically enough). I didn’t realise how much I would have to fine tune with this game, from resources dropped, to health of the goblins, to the cost of upgrades, I had to test over and over again. Every time I added in something new to the game I had to test it with conjunction to everything pre-existing just to make sure it didn’t throw everything out of balance. I could have spent possibly an extra week on the art for the game, fine tuning the animations for the main character as well as the enemies and environment. There were many things I wish I had time to complete but if I tried to complete them before this was due I definitely wouldn’t have come anywhere near to finishing this game.

In the previous version of this game that no longer exists I made a well-functioning game that followed the guide lines we set for ourselves as much as possible .

The GD Project Plan that I worked off of

Ewan Anfield (1633018) Nhlanhla Langa (2119331) Bonga Malusi(1882446) Nikiel Gopichand (1844475)

Erik Prinz [Tutor]

Game Concept

Goblin Explorer is a Top-Down tower defence adventure game where the player can explore the surrounding areas for resources to protect their village.

Intent

1. Create gameplay that is engaging and gives the player agency.
2. Consider how mechanics interact with one another.
3. Do not over-scope.

Design Goals and Methodology

1. Scrum Methodology.
    a. We want to be able to react to changes as quickly as possible.
    b. It allows for rapid prototyping and iteration.
    c. It allows for faster testing and bug fixes.

Our design goals are to:

- Create a responsive movement system.
- Create an intuitive UI and clear communication of assets.
- Create a reactive and challenging combat system.
- Create a meaningful upgrade system.

Specifications

Start and End Dates

24 May 2021 - 14 June 2021.

Game Outcomes and Requirements

The final game must adhere to the details specified in this project plan and the design document.

Game Mechanics and Dynamics

Gameplay States

In the game, days do not pass based on real-time. To pass time to the next day, the player must sleep.

In The Village

- You can upgrade buildings, upgrade weapons and upgrade yourself ● You can buy items.
- You can sleep to remove fatigue.
- You cannot leave the village if you are fatigued.

Outside The Village

- Fight and kill monsters to obtain resources.
- You're limited to the resources that allow you to keep fighting.
- You can choose to return to the village.
- If your health is reduced to 0, you will lose a portion of the resources you found and will be taken to the village.
- Returning to the village by any means gives the player fatigue.

Every Seven Days

The Player has a counter that tells them when the village will be attacked/raided

- After the raid/attack time returns to the morning if the village's health is still positive

Controls

All of the controls listed below are the default values. These values can be changed in the main settings for the game.

Movement

WASD controls the movement of the main character:

W - Up
A - Left
S -
Down D
- Right

The player is constantly facing the direction of the mouse cursor.

Interaction with UI

Press E To interact with buildings, this will open a menu that can be used to upgrade the building. Menu items can be selected by left-clicking on them. The ESC key closes a buildings menu or opens the pause menu.

Combat actions

These combat actions can only be performed outside the village and during raids.

Left-click attacks with the player's sword.

Right-click attacks with the player's gun, this uses ammunition in the player's inventory.

The 1-key uses a healing item on the player.

The 2-key uses a damage boosting item on the player.

Combat Items

The player has a Health recovery item as well as a damage boosting item that can be used in combat. Only 3 of each item can be carried at the beginning of the game but through upgrades, this number can be increased.

Player

The player has three stats:

- Health.
- Sword Damage.
- Gun Damage.

These can be upgraded using **Gold**.

When the player dies they respawn in the village.

Weapon Stats

- The Sword:
  - The sword has durability.
  - When the sword's durability becomes zero, the damage is also reduced to zero.
- The Gun:
  - The gun has limited ammunition.
  - Once the player runs out of ammunition the gun can no longer be used.

Resources used to Upgrade

There are four resources that the player can collect:

- Wood ● Stone
- Iron
- Gold

These resources are used to purchase upgrades for buildings and the player.

There is no limit to the number of resources the player can hold.

Looting

Upon defeating an enemy, the player receives resources. The type of resource rewarded and the amount is randomised. This reward will increase based on the enemy's difficulty.

Buildings and Building Upgrades

Buildings may be upgraded by using a range of resources. Once upgraded these buildings provide better protections for the town as well as upgrades for the player. These buildings include:

- The Wall:
    - The wall affects the outer defences of the village.
        - Upgrading the wall increases the health it provides to the village.
- The wall can be upgraded using iron and stone.
- The Blacksmith:
    - The blacksmith affects the damage of the walls.
        - Upgrading the blacksmith increases this damage.
    - The blacksmith can be upgraded with iron and wood.
    - The Gunsmith:
    - The gunsmith affects the damage done by the gun nests.
    - The player can buy ammunition for Gold.
        - Upgrading the gunsmith increases this damage.
    - The gunsmith can be upgraded with iron and wood but is more expensive than the blacksmith.
    - The General Store:
    - The player can buy combat items from here.

        - Upgrading this increases the number of combat items the player can carry at once.

- The general store can be upgraded using stone and wood.

Player Upgrades

Upgrading the buildings in the village allows the player to purchase an upgrade to their stats with Gold. As the buildings are upgraded, more player stat upgrades will be available for purchase.

- Health:
    - The player’s maximum health can be upgraded at the wall.

- Sword Damage:
    - The player’s sword damage can be upgraded at the blacksmith.

- Gun Damage:
    - The player’s gun damage can be upgraded at the gunsmith.

Enemies

- Goblins
    - These enemies primarily reside within the forests.
    - They attack the village during raids.
    - They have low attack damage and health.
    - These units drop loot.
- Goblin Kings
    - There are 3 goblin kings within the game.
    - These units are programmed to encounter the player at set intervals.
    - Killing a goblin king increases the health and damage of all goblins within the game.
    - After killing the last goblin king the village will be attacked by one last wave.

Game Over States

- Losing state.
    - This state is achieved when the village’s health reaches 0.

- Winning state.

- This state is achieved when the player survives the final wave.

Feature List

I. Data Design

II. The data displayed here is subject to change. It is open to iteration.

Player Stats

Player	Health	Sword damage	Gun damage
Base	100	12.5	25
1st upgrade	150	30	50
2nd upgrade	200	50	75
3rd upgrade	225	70	100

Goblin Stats

Goblins	Health	Damage
Base	50	10
1st Goblin King Kill	100	20
2nd Goblin King Kill	150	30
3rd Goblin King Kill	200	40

Goblin king Stats

Goblin King	Health	Damage
1st goblin king	250	35
2nd goblin king	550	65
3rd goblin king	900	100

Village Stats

Village	Health	Wall damage	Gun nest damage
Base	1000	12.5	25
1st wall upgrade	1200	30	50

2nd wall upgrade	1400	50	75
3rd wall upgrade	1600	70	100

Wall Upgrade

Upgrade Level	**Iron Cost**	**Wood Cost**	**Stone Cost**	**Village Effect**	**Available player upgrade**
1	5	0	10	+200 village health	+50 Player Health
2	10	0	20	+200 village health	+50 Player Health
3	20	0	40	+200 village health	+25 Player Health

The Blacksmith Upgrade

Upgrade Level	**Iron Cost**	**Wood Cost**	**Stone Cost**	**Village Effect**	**Available player upgrade**
1	5	10	0	+17.5 wall damage	+17.5 Sword Damage
2	10	15	0	+20 wall damage	+20 Sword Damage
3	20	25	0	+20 wall damage	+20 Sword damage

The Gunsmith Upgrade

Upgrade Level	**Iron Cost**	**Wood Cost**	**Stone Cost**	**Village Effect**	**Available player upgrade**

1	5	10	0	+25 gun nest damage	+25 Player Gun Damage
2	10	15	0	+25 gun nest damage	+25 Player Gun Damage
3	20	20	0	+25 gun nest damage	+25 Player Gun Damage

The General Store

Upgrade Level	**Iron Cost**	**Wood Cost**	**Stone Cost**	**Available Player Upgrade**
1	0	5	10	+1 max capacity per combat item
2	0	10	20	+1 max capacity per combat item
3	0	20	40	+1 max capacity per combat item

The Wall Store items

Item	**cost**
1st player health upgrade	20
2nd player health upgrade	30
3rd player health upgrade	50

The Blacksmith Store items

Item	cost
1st sword upgrade	20
2nd sword upgrade	30
3rd sword upgrade	50

The Gunsmith Store items

Item	cost
Ammunition	1 per bullet
1st gun upgrade	20
2nd gun upgrade	30
3rd gun upgrade	50

The General Store items

Item	cost
Health recovery	15
Damage boosting	20

The resource reward from combat is a maximum of 2 * the difficulty modifier, split evenly between a maximum of 2 different resources. There is a 25% chance of receiving a reward of 2 resources. If only 1 resource is rewarded, each resource has the same chance of being selected (33.3%). If 2 resources are rewarded, the first will be selected from the 3 resources with a 33.3% chance, the second will be selected from the remaining 2 resources with a 50% chance. The gold reward from combat is 4*difficulty modifier gold. The resulting equations will look like this:

Resource Reward Amount = 2 * Difficulty Modifier

Gold Reward Amount = 4 * Difficulty Modifier

III. Communication Design

The player's health bar will be displayed on the top left of the screen.

The player's combat items will be displayed to the right of the health bar. A small number will be adjacent to these images indicating the amount of this item that the player is currently holding. On top of this image will be a letter that indicates which button on the keyboard activates it.

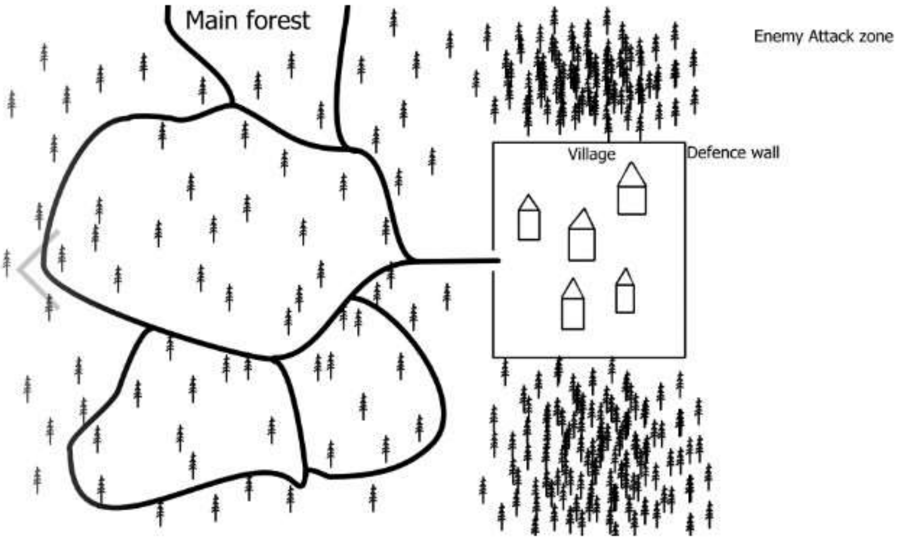
There will be two icons at the bottom centre of the screen. The left icon will be an image of a sword and the right a gun as per default settings. These icons represent the player's weapons. There will be a bar underneath the sword icon that represents the durability of the sword and a number underneath the gun that represents the ammunition amount available. An image of a mouse below these icons will indicate which mouse button activates which weapon.

The loot that the player is currently holding will be displayed on the top right of the screen.

When walking next to an ‘interactable’ the player will be presented with an ‘E’.

This indicates the button that they have to push to interact with the object.

IV. Level Design



- The main forest is where the player will venture out to kill goblins and gain loot. The further away the player gets from the town, the more the goblins appear.
- The village is what the player must protect from the raids. This is the place where the player can upgrade buildings as well as upgrade themselves.
- The defence wall is what gets upgraded. If this wall falls then the goblins will destroy the town.
- The enemy attack zone is where the raids will come from.

V.Feedback Loops

There are 2 main feedback loops in the game.

The first feedback loop is the upgrade system. This is a positive feedback loop as using resources to upgrade buildings allows the player to become stronger. Thus being able to fight and gain resources more effectively.

The second feedback loop is the difficulty system. This is both a positive and negative feedback loop. The positive aspect of this loop is that as the difficulty increases so do the rewards the player gets from combat, this feeds back into the upgrade system. The negative aspect of the loop is that the difficulty determines how hard enemies are to kill, thus shifting the state of the game to make the player weaker in comparison to their enemies.

Task Breakdown

VI. Needs & Wants

Needs-

Player stats

Movement system

Combat system

Combat reward system

Building upgrade system

Player upgrading system

Enemy stats

Enemy difficulty system (Goblin King system)

Enemy Combat system

Tower Defense system

- ○ Enemy waves attacking the village.
- ○ The village attacks the enemies in real-time.

Wants-

- Dodge mechanic
- Multiple unique bosses

Dreams-

- Multiple weapons with different stats.
- Multiple enemy types with different stats.
- More buildings with unique upgrades.

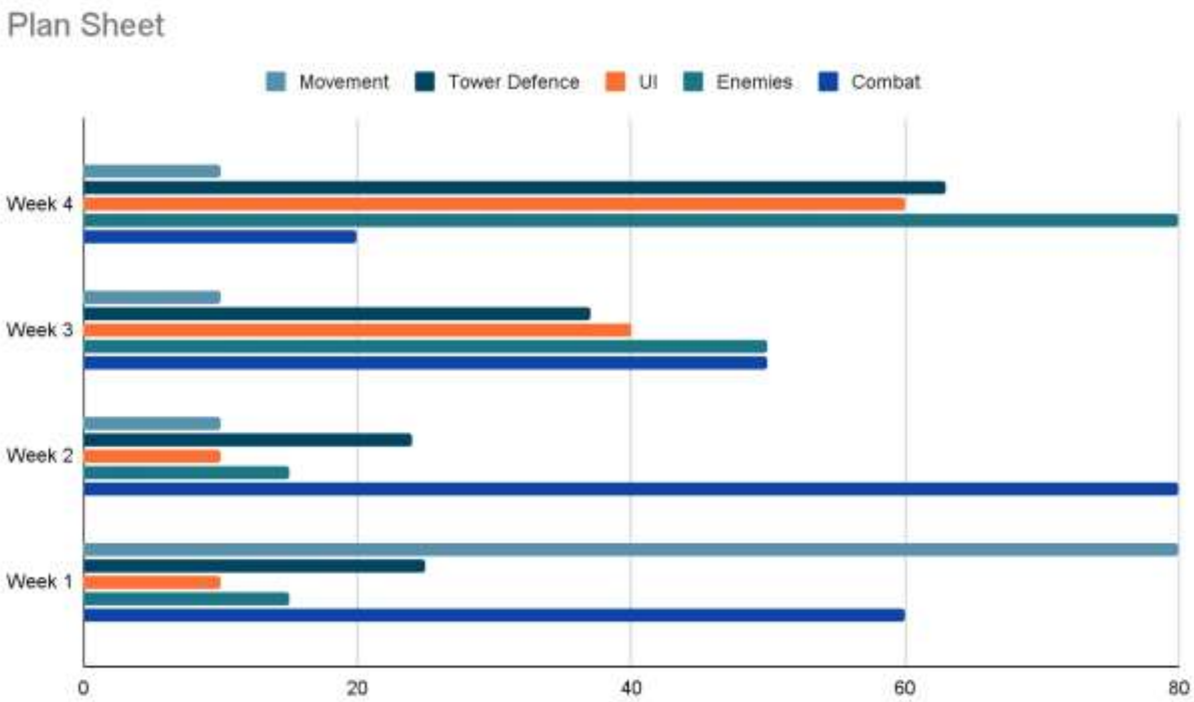
VII.Features to Tasks

Priority works in terms of numbers. The lower the number, the higher the priority. For instance, priority 1 is the most important task we have to complete.

Features	Importance
Player stats	Priority 1

Movement system	Priority 2
Combat system	Priority 3
Combat reward system	Priority 3
Building upgrade system	Priority 4
Player upgrading system	Priority 4
Enemy stats	Priority 5
Enemy difficulty system (Goblin King system)	Priority 5
Enemy combat system	Priority 5
Tower Defense system	Priority 6

VIII.Product Plan Sheet

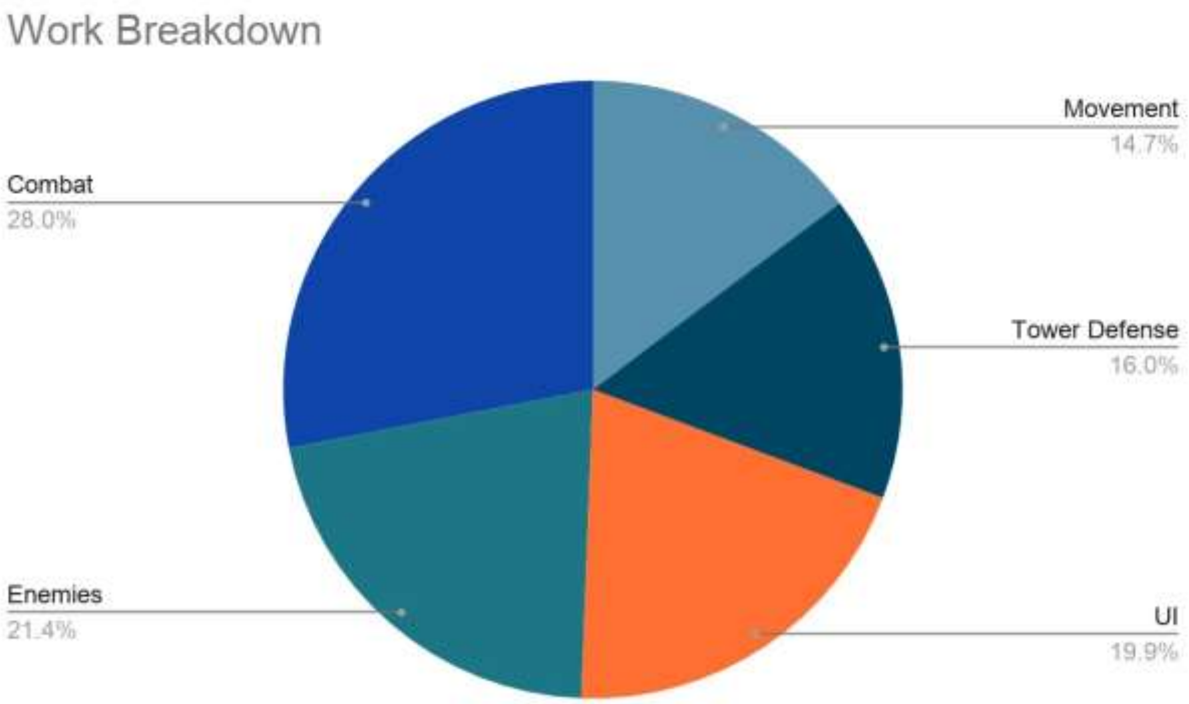


Dependencies

- Player stats have no dependencies.
- The movement has no dependencies.
- Combat depends on movement and player stats.
- Combat rewards depend on combat and enemy difficulty.
- Building upgrades rely on combat rewards.
- Player upgrades rely on building upgrades and player stats.
- Enemy stats have no dependencies.
- Enemy difficulty relies on enemy stats.
- Enemy combat relies on enemy difficulty.
- Tower defence relies on everything.

Scheduling

IX.Work Breakdown as a Percentage of Total Work Time



Risks

We may have over scoped and therefore the game may be incomplete. The game may also not be what we were meant to make and as a result, we would get a bad mark.

Milestones

Finish player stats.

Finish the movement system.

Finish combat system.

Finish combat reward system.

Finish building upgrade system.

Finish player upgrading system.

Finish enemy stats.

Finish enemy difficulty system.

Finish enemy Combat system.

Finish tower Defence system.

Submit.