README Spellslingers Orbital 2022

Team Name:

Spellslingers
Alvin Tan and Tan Yi Zhe

Proposed Level of Achievement:

Apollo 11

Motivation

Both of us are avid gamers... and we have lots of fantastical ideas on the many different things we want to put in each game we play.

This is an opportunity for us to start working towards crafting the game we both had in our minds for a long time: a very customizable RPG that incorporates roguelike elements, bosses, different enemies, different combat types and overall good time in the world of our game.

We want to start off with something simple as a basis for (maybe) more development in the future.

Aim

We hope to create a 2D top-down roguelike pixel game that is hugely customizable, with engaging mechanics, and most importantly, fun to play.

User Stories

1. As a gamer, I want to have a different experience each time I start up the game.

2. As a gamer I want to be challenged by the game, fighting bosses with different mechanics that I can learn and get better at playing against.

3. As a gamer, I want to learn about how to make a game.

4. As a student, I want to enjoy fast gameplay so that I have time to study, but immersive gameplay to take my mind off the stressful academic environment.

5. As a gamer, I want to dive right into playing the game, so I hope that the game is intuitive and easy to pick up.

Game Ideation

The driving force behind our game's idea is customizability. We want to create a game that plays differently each time we start it up. One of my favourite games is <Path of Exile>. It is an ARPG with immense customizability through a large passive skill tree and hundreds of skills.

One game that might not look similar at first glance but upholds the same aspect of customizability is <Vampire Survivors>. Through the level up mechanic and different characters, <Vampire Survivors> is able to create a fresh experience every playthrough. The satisfaction of clearing multiple waves of monsters is also a major selling point of the game.

We are also inspired by how <Terraria> uses different terrain and bosses to diversify players' gameplay experience. They have cleverly designed bosses that are mechanically challenging and fun to battle against. This gives players goal-oriented gameplay, it is something they can work towards.

Character Designs

We used Aseprite to create pixel art, as it is more forgiving for beginners like us to make art. We also like pixel art - it has a certain "charm" to it (personal opinion).

Maps - the World of Ashlands

We have created tilesets from Aseprite as well to add variety and different terrain to our world. We are using free assets from Unity Store called Tiny RPG Forest 2D tilesets to create one of our maps.

Music and SFX

We are using creative commons license CC BY music:

- Volcano BGM - Solace by Scott Buckley
- Dry Sea BGM - Hiraeth by Scott Buckley
- AshBasin BGM - Signal To Noise by Scott Buckley
- Main Menu BGM - Helios by Scott Buckley
- Jungle BGM - Clarion by Scott Buckley
- Snowfields BGM - Jul by Scott Buckley

We are using SFX from freesound.org and freesfx.co.uk.

Tech Stack

- Unity Game Engine
  - Game engine used for implementation for our game.
- Aseprite
  - The software we used to make the sprites for the game.
- C#
  - The language to write scripts for Unity.

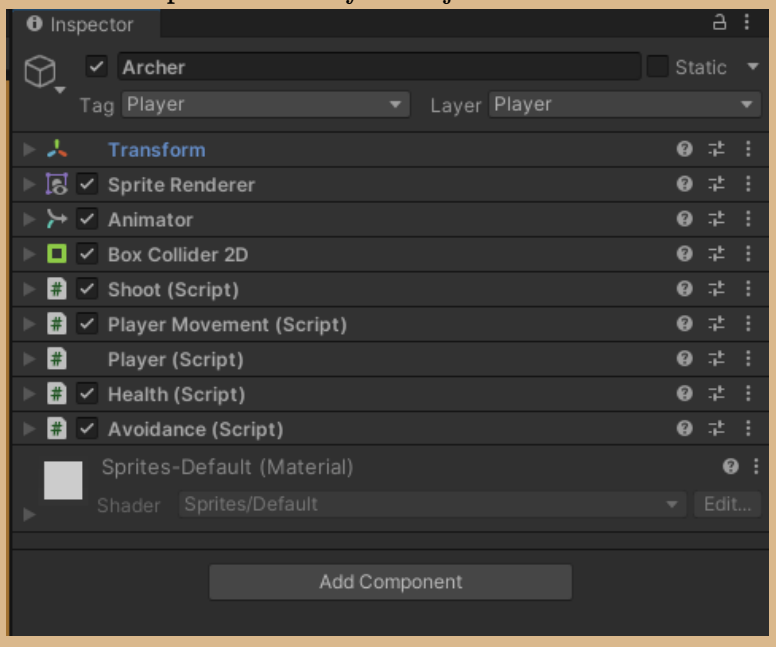
Timeline

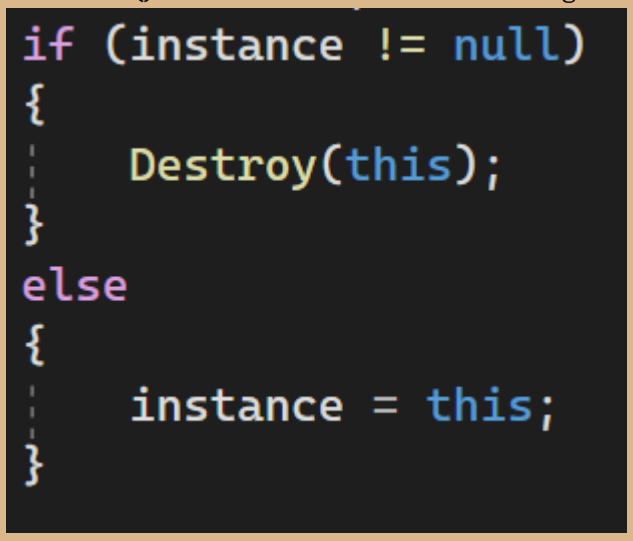
Date	Features
10th May - 17th May	**Unity** Learn and familiarise ourselves with unity's system **Gameplay** Prototype basic gameplay (movement, enemy AI, player attack) Enemy Spawner **Sprites** Create sprites and animations **UI** Working Main Menu, Exp bar, health bar **Others** Work on submissions for liftoff
18th May - 24th May	**Gameplay** Create new skills for players, create new monster types Create 3 new enemy Create 3 new skills

	Sprites Create new sprites for UI and new enemies **UI** Pause menu, death screen menu
25th May - 31st May	**Sprites and Tiles** Create at least 1 map from Tiny Forest RPG tileset Create second tileset from Aseprite Create new UI and sprites for skills and enemies **Gameplay** Look into terrain effects Create 3 new enemy Create 3 new skills **UI** Character sheet for stats **Others** Work on submissions for milestone 1
1st June - 7th June	**Gameplay** Create 3 new enemy Create 3 new signature skills Look into boss AI
8th June - 14th June	**Gameplay** Work on Minotaur boss and Dragon boss, implement their AI Create a new tileset on Aseprite Create a new map (total of 4 now)
15th June - 21st June	**Gameplay** Work on Minotaur boss and Dragon boss, implement their AI Create new skills, sfx, enemies and music if time allows Create a new tileset on Aseprite Create a new map (total of 5 now)
22nd June - 28th June	**Gameplay** Work on Phoenix and Chimera bosses Create new skills, sfx, enemies and music if time allows **UI** Map UI **Others**

	Work on submissions for milestone 2
29th June - 5th July	**Gameplay** Work on Mage and Warrior classes, add their basic skills Create new skills for Mage and Warrior to match archer
6th July - 12th July	**Gameplay** Final boss Add music and SFX Create terrain effects (like acid rain, Frozen ground)
13th July - 19th July	**Gameplay** Catch up on an other unfinished parts/polish the game **Others** Playtesting, feedback and user testing Testing and debugging
20th July - 26th July	**Others** Work on submissions for milestone 3

Software Engineering

Design Principle / Pattern	Implementation
Component Pattern	Unity uses a component system that allows us to modify GameObjects by adding and removing components. That includes scripts that modify the object. 

Singleton Pattern	A singleton is a globally accessible class that exists in the scene once. For example, the player character is a singleton, so is the Death Screen Menu, and many other things.
	To make something a singleton, we put the following code in the Awake() function to make the class a singleton.
	 
	`if (instance != null)`
	`{`
	`  Destroy(this);`
	`}`
	`else`
	`{`
	`  instance = this;`
	`}`

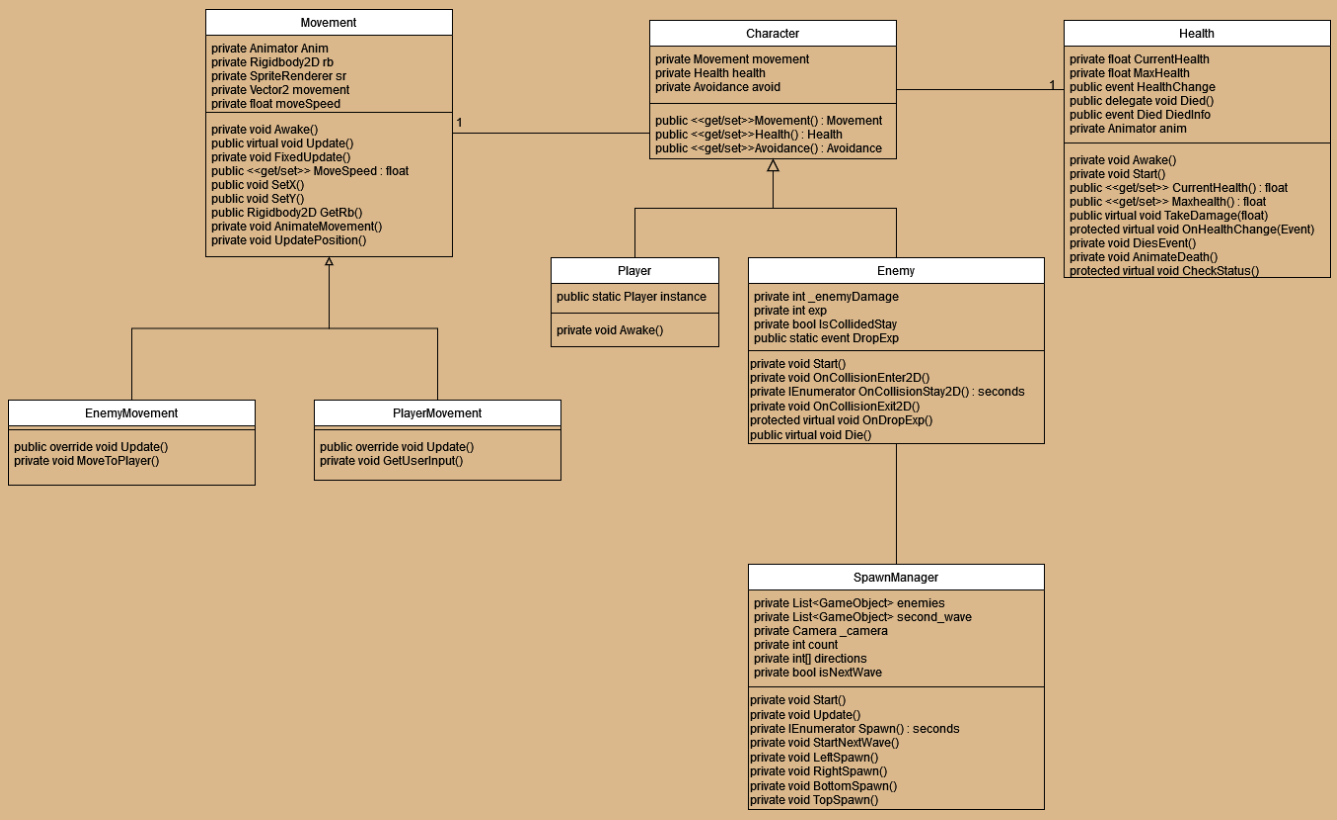
Polymorphism	We have a base skill and character class that can turn into other skills, players and enemies respectively.
Single Responsibility Principle	We split a large class into smaller classes so that each class can be more easily maintained.
Observer Design Pattern	Using this pattern, we have the user interfaces subscribe to events created by the classes holding data and are notified when any data change occurs.
Continuous Development	We use GitHub to develop simultaneously.

System Design *(Updated for Milestone 3)*

1. Character

Using the single responsibility principle, we separated the health and movement into their own classes so that the character class has less responsibilities. (Refer to the UML diagram section for more details.) Using polymorphism, we then have the character class polymorph into player and enemy classes. Since there is only one player in the game, we make use of the singleton pattern to ensure that the player is globally accessible to the other classes. Use of component pattern to attach different components to different enemies for them to have different skills.

UML Diagram:



2. Health Bar

Using the single responsibility principle, we separated the health ui from the health class so that the health class only needs to manage the health data while the health ui only needs to manage how the health data looks like. Through the observer design pattern, we have the health ui observe changes to the health data in the health class and then update the health bar accordingly. When the player's character and enemies spawn, the health bar is then attached to them. This allows the player to easily see both their character's and enemies' health, with different coloured health bars for more clarity.

3. Enemy Spawner

We have a spawner manager that manages when the spawners can start spawning enemies. This follows the single responsibility principle where each spawner is only responsible for spawning their own set of enemies. Enemies are spawned through waves made with prefabs and using the observer design pattern, we observe the player's level to determine when to spawn new waves. To ensure variety in each wave, we spawn the enemies randomly from the set of available enemies in each spawner. We then ensure constant engagement with enemies by spawning around the camera view. We also use coroutines to ensure that players do not

get overwhelmed in the early stages of the game and to increase spawn rate once the player gets stronger.

4. Experience + Skills system

Through the observer design pattern, we have the experience manager observe the enemies' death to increase exp. The experience manager then determines whether the player can level up. This is in turn observed by the skills manager which then generates the skills. The level up menu UI is then notified when the skills are ready, displaying the skills that the player can choose from.

For the skills manager, we use the Fisher-Yates shuffle to get a random permutation of the list of skills then we can choose from for randomisation. A hash set is also used to ensure that the same skills are not offered to the player more than once in the same level up.

For the skills, we use polymorphism to have an abstract base skill class that can polymorph into different skills. Each skill has a button attached to them, which they observe to determine which skill the player has selected and give the corresponding upgrades. The skills also have a level attached to them to allow us to balance them accordingly, with stronger skills spread out over more levels with weaker upgrades and weaker skills spread out over less levels with stronger upgrades.

We also introduce signature skills, a one time skill upgrade at level 10, 20 and 30, which can drastically modify the player's skills to improve the variety of builds that the player can come up with.

5. Status Effects

Using polymorphism, we have an abstract status effect class that can polymorph into different status effects. This allows us to introduce more mechanics into the skills that are available in the game. To prevent the same status effects from stacking, characters have a list of status effects currently on them. When they receive a pre-existing status effect, the pre-existing status effect is removed and the duration will be refreshed to the duration of the status effect just received, with the strongest effect of either. Coroutines are used to manage the duration of the status effects.

6. Navigating the Ashlands (Minimap + Map UI)

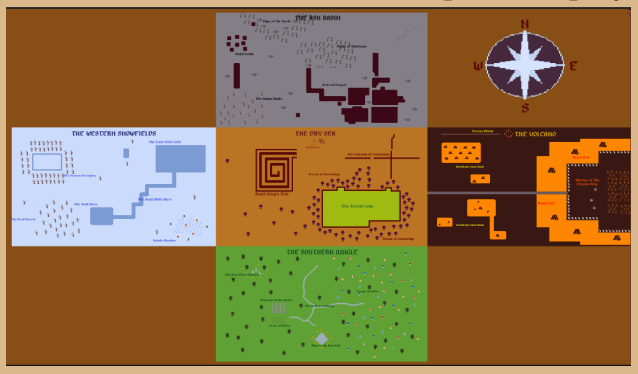
We constantly keep track of the player's position and then update the minimap accordingly to let the player know which zone they are at. We use the observer design pattern to then observe items in the game, updating map icons and unlocking new zones based on those items. The map icons serve to let the player know what else they can still do. The glow represents which zone the player is in.

Item not yet picked up:

Boss not yet defeated:

Map **(New in Milestone 3)**

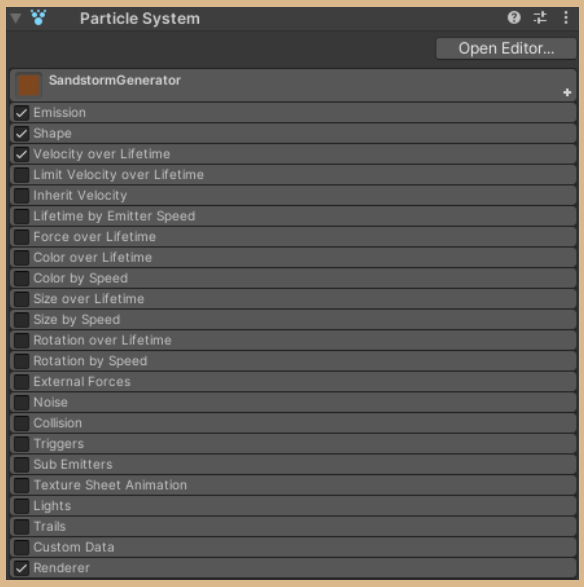
We have also added a world map for the player to navigate the Ashlands.



More about this in the Gameplay Concept section.

7. Weather System

We used Unity's particle system to generate different particles that depict different weather effects.



By changing the emission, shape, velocity of each particle emitted, we were able to emulate weather effects based on the zone the player is in.

<u>Weather effects</u>

1. Sandstorm in the arid lake. We wanted the sandstorm to travel horizontally to imitate how the wind usually blows, from left to right. We wanted the sand particles to feel like grains of sand blowing from right to left, rather than large clumps of sand.

2. Blizzard in the snowfields. In contrast to the sandstorm, we wanted the particles to be large clumps of snowfall and hail raining down onto the player.

3. Rain in the jungle. The weather in the jungle is the more generic rain. Rain falls vertically, and is thin and long, similar to raindrops in the real world.

4. No weather in the lava zone, but we used the particle system to show the shadows of smoke clouds emitted by the nearby volcano.

5. Ashfall in the Ash Basin, less visible weather effects than the rest but we just wanted to depict a light fall of ash to give it ambience, nothing flashy.

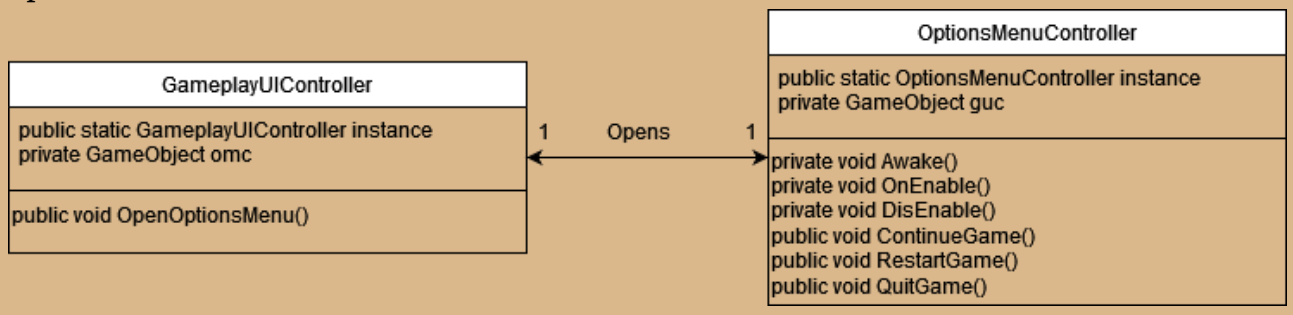
8. Event Messages
We use the observer design pattern to observe events in the game and send out messages accordingly to guide the player on progression in the game.

Event	Purpose
Start of game Controls: wasd for movement, mouse to aim	Inform player of controls
Entering zone not yet unlocked It is too cold to enter now.	Inform player that they cannot enter yet
Unlock new zone You are no longer bothered by the cold. Fields of snow to the west is now available.	Inform player of new zone
Start of boss fight You see a bright flash...	Inform player of boss
When the game is paused. PAUSED	Inform the player the game is paused.

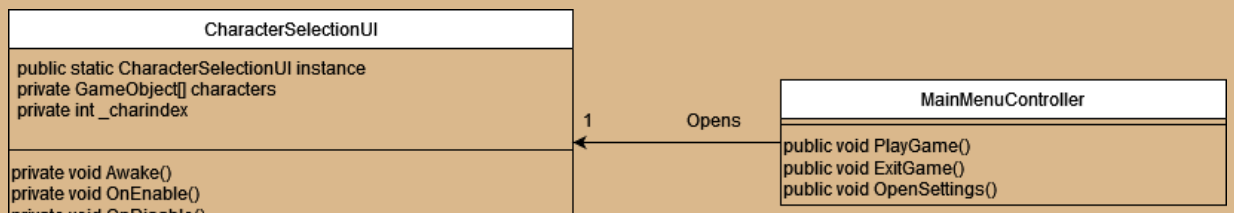
9. Unlocking new zones
We wanted player progression to be more defined, rather than it being confusing (due to the nature of open worlds). We still have a relatively open world to explore, but new zones are now locked behind finding items and defeating bosses.

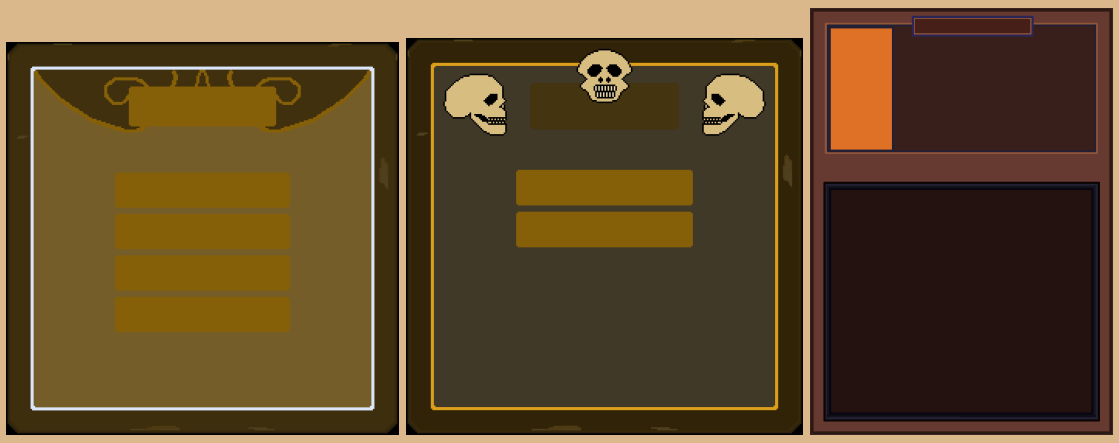
10. In-game User Interface

Options/Pause Menu



Main Menu

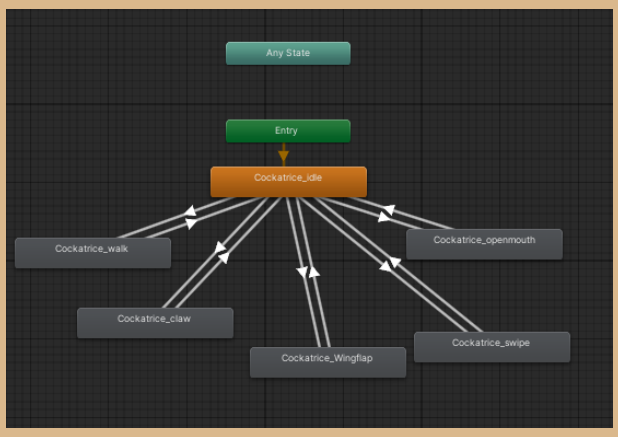


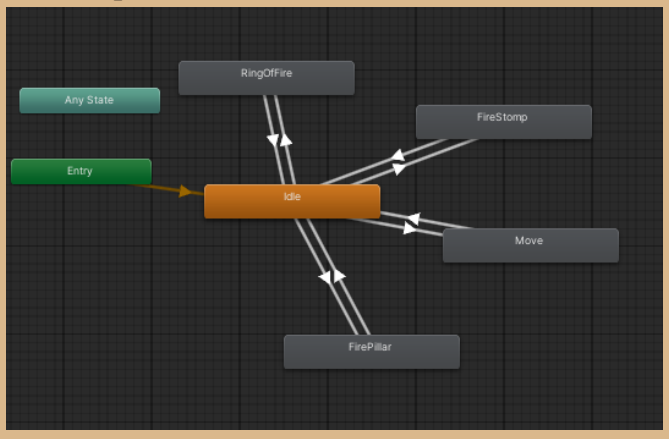


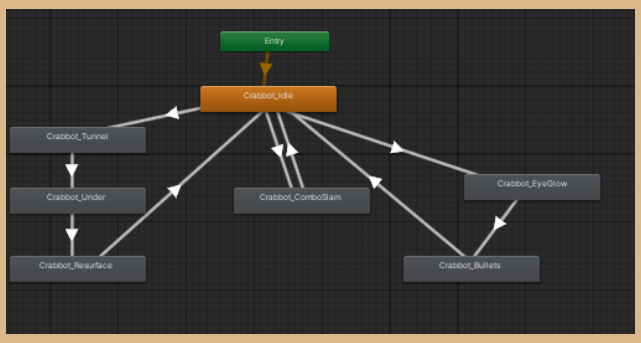
We created our own menu sprites that fit the theme of our game.

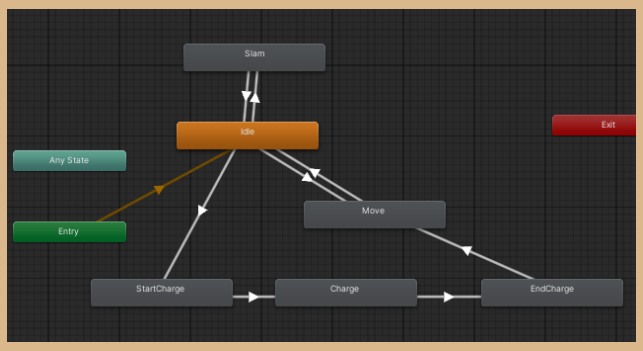
11. Boss Behaviour

We used Unity's Animator system as well as C# scripts to control each boss' behaviour.









We used parameters to control the transitions between Boss abilities.

```
private void CastRingOfFire()
{
    this.anim.SetTrigger("RingOfFire");
}

private void CastFireStomp()
{
    this.anim.SetTrigger("FireStomp");
}

private void CastFirePillar()
{
    this.anim.SetTrigger("FirePillar");
}
```

By triggering a parameter or setting a boolean to a parameter, we can control the boss behaviour, making sure each boss does not repeat the same ability twice, and do not use another ability in the middle of casting a different ability.

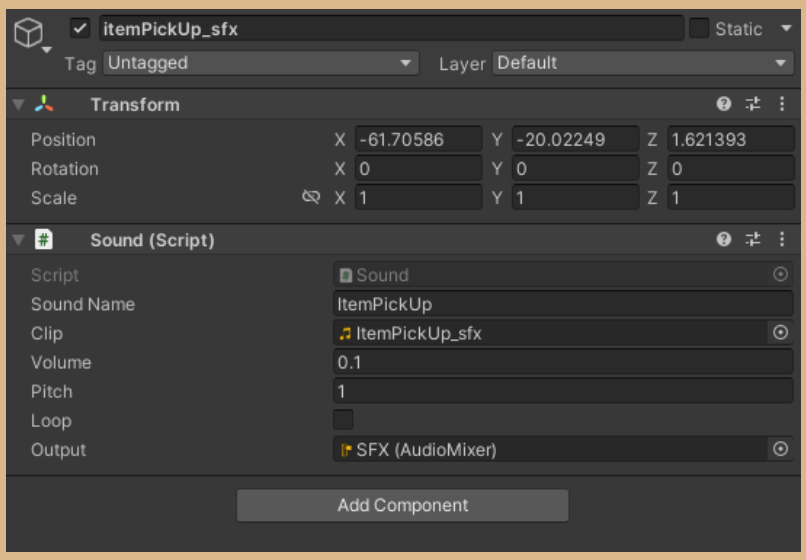
12. Character Select

We use the observer design pattern to observe which character the player will choose. We then load the character, character sheet and skills based on that.

13. Settings

We use the singleton design pattern to ensure that there is only one instance of the settings manager. This ensures that the settings for both the settings menu in our game and the settings option in the main menu are the same. To ensure that we are able to adjust the volume of both the background music and sound effect, we set the output of the different sound effects and music to Unity's audio mixer, which allows us to set the volume before outputting to an audio listener.

14. Sound and Music



We wrap the different sound effects and music in a sound class. We then have an audio manager that holds all sound objects, creating an audio source for each sound object when the game starts. We use the singleton design pattern to ensure that the audio manager is globally accessible as many things in the game have sound effects such as the player, enemies, bosses and skills. They can simply look up the name of the sound effect they want to play in the dictionary of sounds in the audio manager in constant time. With the audio manager we can also ensure that only one instance of a sound effect is played once, this prevents the sound effects from overlapping with one another when multiple enemies of the same time try to play their sound effects at the same time. The background music also changes based on the biome the player is currently in.

Gameplay Concept

You awake in the Ashlands with only your clothes and your weapons to your name. You must fight through hordes of enemies who want to kill you. Try your best to survive the unforgiving Ashlands...

There are 3 classes that are playable:

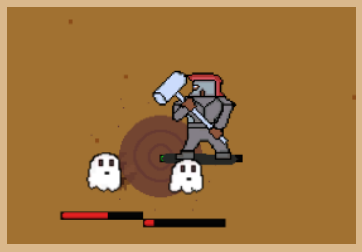
Gameplay Idea

Player character starts in the center of the world map. Objective of the game is to level up, explore the world, and eventually become strong enough to kill the final boss.

There are main parts gameplay wise:

1. Choosing a class

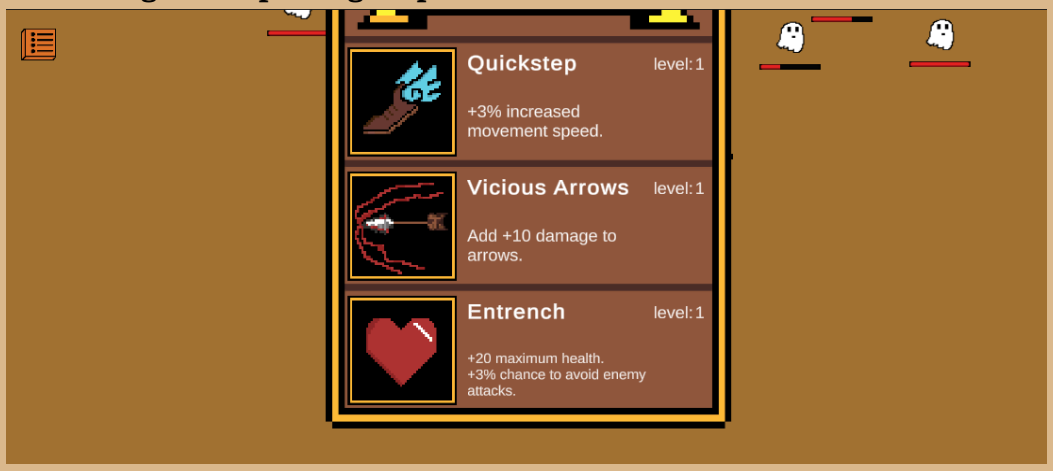
The game has 3 classes to choose from, each with a different basic attacks, upgrades and unique signature skills they can unlock at levels 10, 20, 30. We aim to create different playstyles/approaches to games with these classes, and signature skills that can further specialise each character class.

Warrior - Slam 	The Warrior is a Melee class for players that enjoy bashing enemies up close. Basic Attack: The Warrior slams his hammer in front of him, dealing damage in an area.
Archer - Arrow 	The Archer is a ranged class that has very good damage and range with arrows. Basic Attack: The Archer fires 1 arrow at a target enemy
Mage - Lightning 	The Mage class is a short-ranged class that has powerful and unique spells. Basic Attack: The mage casts a lightning spell at a target enemy, dealing damage in an area..

2. Combat

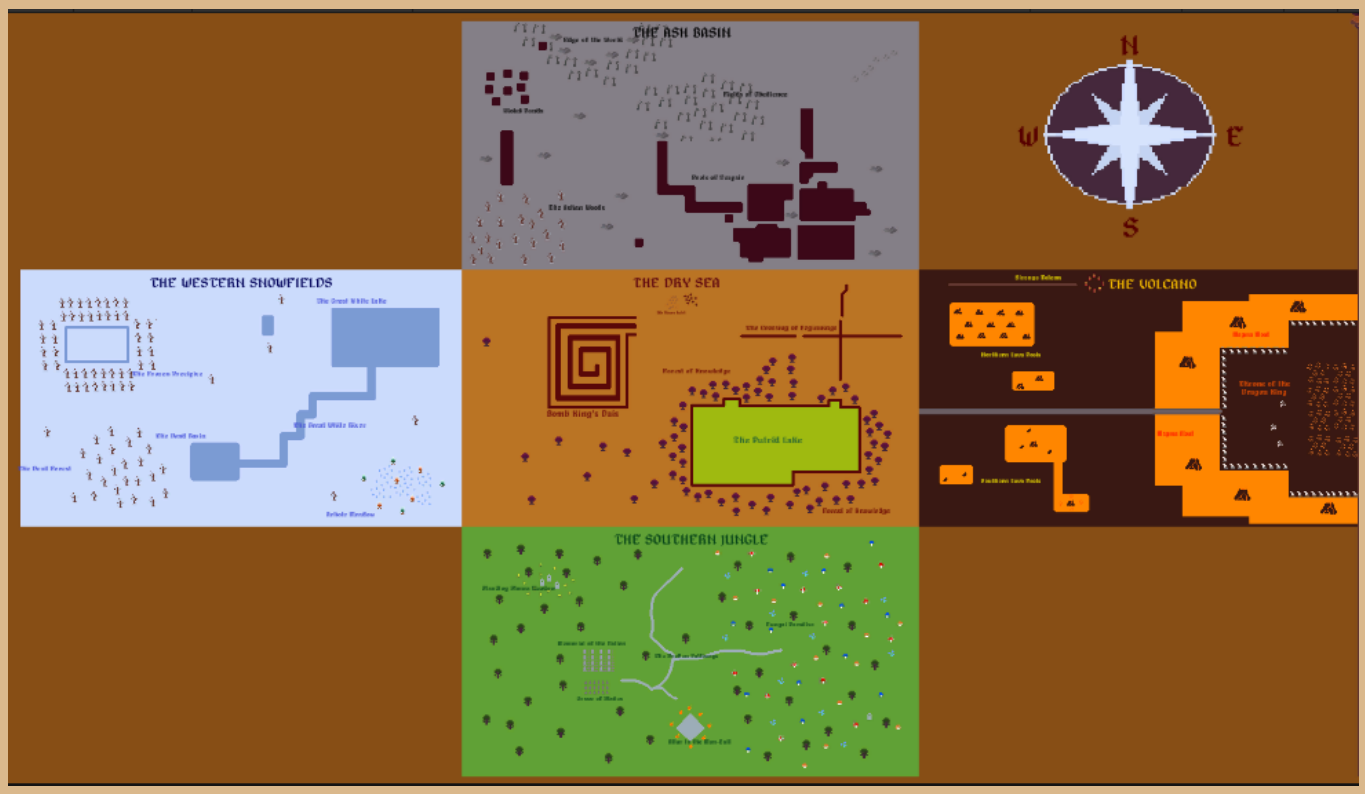
Players will start fighting monsters by using WASD to move and mouse movement to target enemies. Enemies will track the player down and follow the player in an attempt to kill the player (in game).

3. Leveling and exploring map



Players will explore the map, finding good spots to kill monsters and gain XP. Each region will have different enemies the player can face, with varying mechanics and different XP gains. We believe this will make the game more diverse.

We have created a world map to facilitate exploration.



The Map contains many landmarks that are labeled. During gameplay, the player may observe certain landmarks in the world. The player can find the landmark location on the map, and deduce their own location relative to the position of the landmarks they find.

4. Fight harder Waves

As the game progresses, you fight stronger enemies. Enemies gain new mechanics. One example is the new enemy, bomber. The bomber explodes on death, dealing damage to players in its proximity.

Please refer to the appendix for the full list of enemies in the game.

5. Finishing the build

Players will try to go for final levels before attempting the bosses.
There will be skills that augment the player's single target and enemy clear speed throughout the game.
Signature skills make every playthrough interesting and unexpected. Different signature skills can make different builds.

6. Endgame *(Updated for Milestone 3)*

- Players will challenge the 5 bosses which will drop their unique items for players to advance the game... if they beat them, that is.
- The 5 bosses will be:

Minotaur

A bull “blessed” with conscious thoughts, emotions, intelligence and bipedalism as though it were a human. Lives in the jungle region.

Moveset:

- Slam

Minotaur slams the ground. After a brief delay, spikes erupt from the ground in a cone-like area.

- Charge

Minotaur prepares to charge. After a brief delay, charges in the direction of the player.

Dragonewt

The last remaining creature of an ancient race of lizard people that loves to eat humans. Lives in the volcanic region.

Moveset:

- Fire Stomp

Dragonewt stomps on the ground, alternating between each leg. When Dragonewt's feet contact the ground, a circle of flame projectiles will spawn. Draonewt stomps a total of 4 times.

- Flame Pillar

Dragonewt casts a summoning circle at the player's location. After a brief delay, a pillar of fire erupts.

- Ring of Fire

Dragonewt channels a ring of fire. The ring of fire then expands outwards.

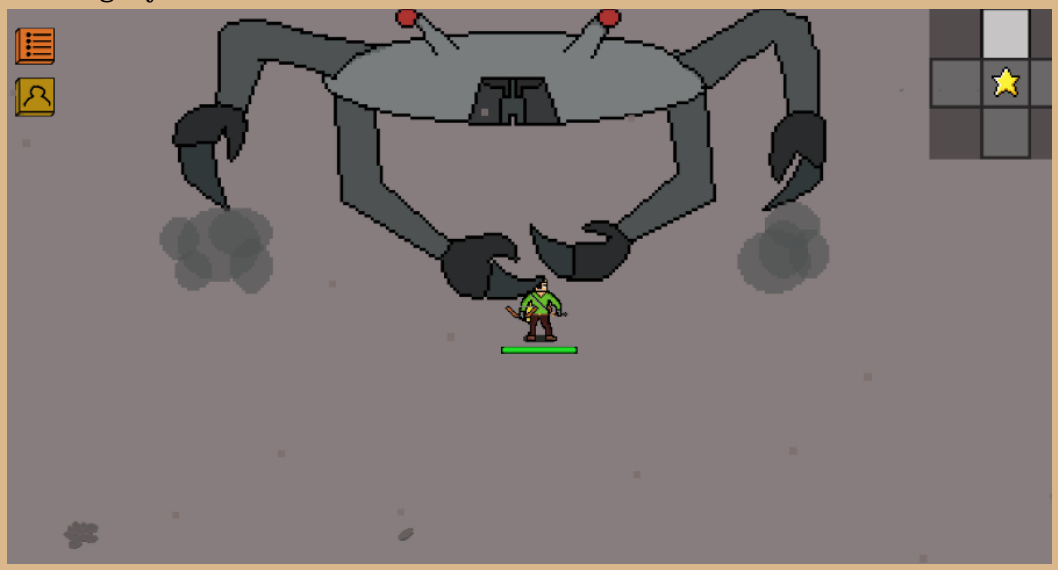
Crabulon

A remnant of an ancient civilization that lives at the edge of the world. Said to be eating the world piece by piece every century.

Moveset:

- Claw Combo Slam

Crabulon slashes in front of him with 2 claws, then quickly brings down his other 2 claws in a mighty slam.

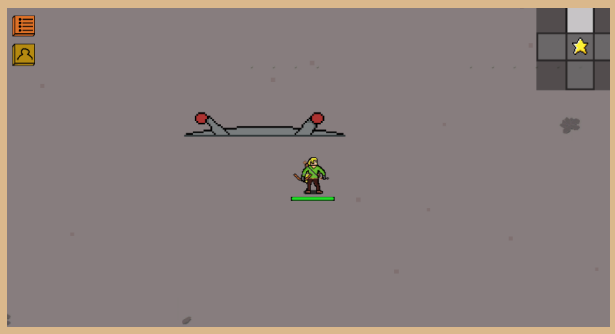


- Exterminate

Crabulon takes on a stance that allows it to accumulate power and blasts its enemies with a flurry of bullets.

- Tunnel

Crabulon tunnels deep into the ash, skewering its enemies with its claws from deep within the ground.



Cockatrice

The cruel cockatrice, the abominable offspring of an ice dragon and large rooster.

Moveset:

- Air cutter

Summons a deadly gale of wind that rips through the air, dealing devastating damage.

- Icicle Rain

Summons a row of icicles that rain down on its enemies, piercing through their flesh.

- Wing Attack

When in melee range, the cockatrice swipes at its enemies with its wings, in an attempt to cripple the enemy.

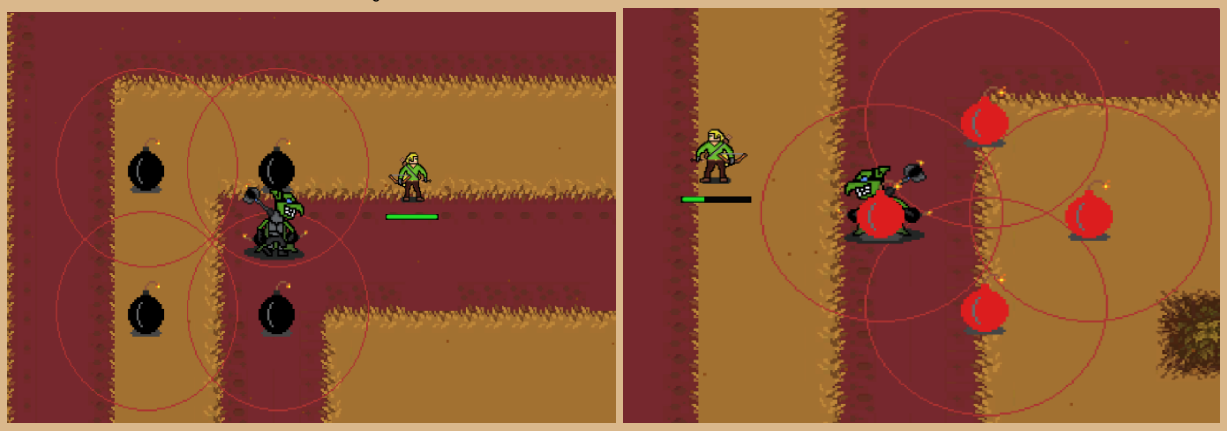
Bomb King (New in Milestone 3)

The tyrannical Bomb King, rules by fear with his terrifying bombs.
Moveset:

- Throw Bomb

Bomb King throws a bomb in the direction of the player. The bomb slows down when it is near the player, exploding after a while.

- Throw Bombs Everywhere

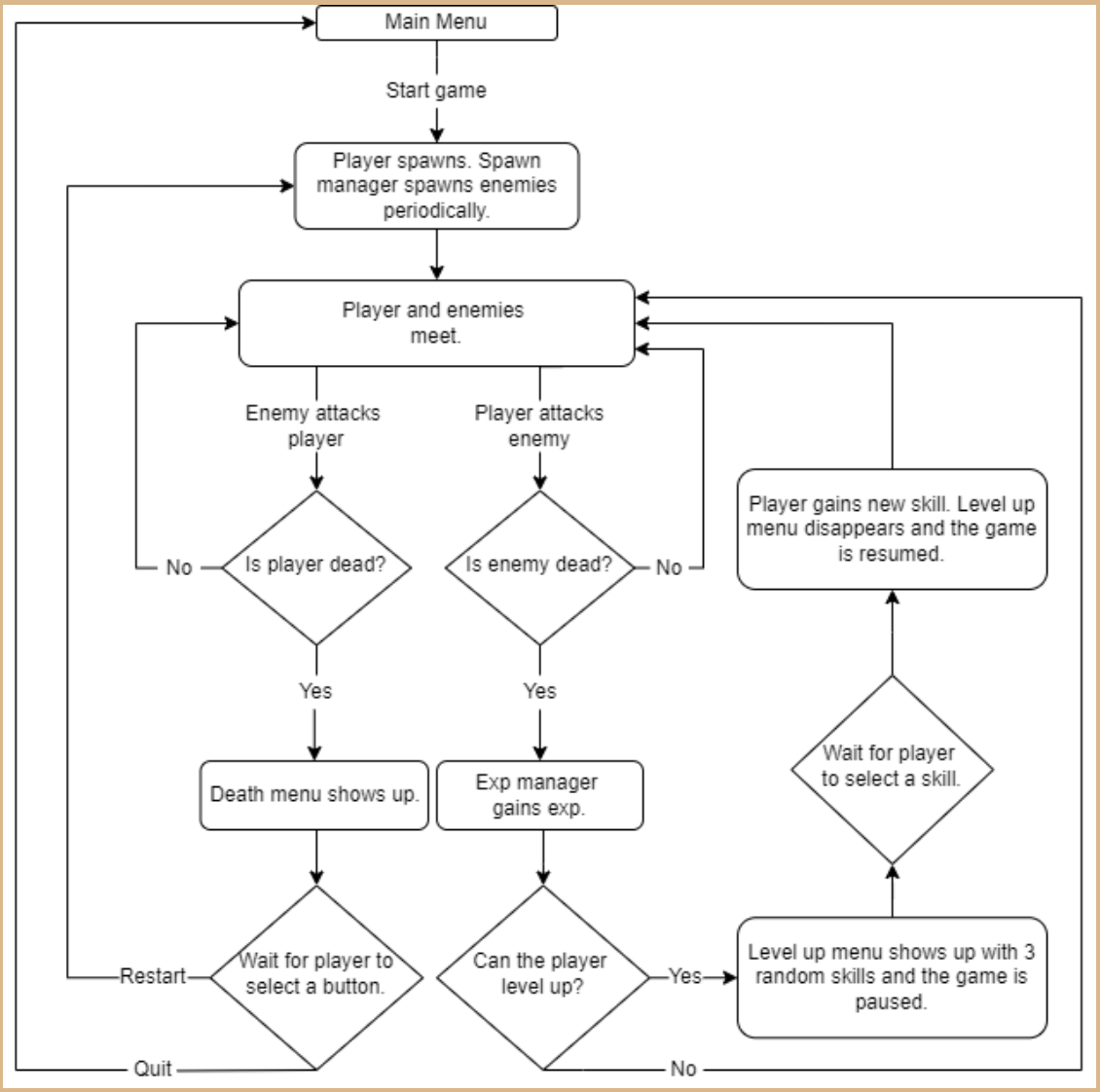


Bomb king throws 4 bombs in 4 different directions. There are 2 versions of this move.

7. Death

- If players die halfway through the game, a death screen will show and the game will be restarted. It is a roguelike afterall.
- A death screen will pop-up, taking the player back to the start of the game or the main menu.

How the game will work:



Conclusion

The game has achieved the objectives that we planned to achieve by milestone 3. We are proud of our progress in these past few months. However the game still has room for improvement, there could be a lot more enemies and skills, and the game balance could be better.

We have also chosen not to animate certain animation we considered optional to have more time to implement and test our features. This includes death animations, experience points getting picked up by the player, celebratory effects when leveling up and so on, which would have made our game more visually pleasing to look at.

Testing and evaluation (Updated for Milestone 3)

For testing, we playtest the game directly, using Unity's console to watch for errors and the in-game visuals to see if any bugs have occured. We have also done more user testing by getting more people to try our game through play testing sessions held by NUS GDC and our friends.

Bugs	Fixes
Taking Damage The player was taking more damage than the amount we intended.	Removed the method causing the player to take damage on initial collision with the enemy and kept the method that causes the player to take damage on maintaining contact with the enemy.
Health Bar Enemies that were too big blocked the health bar from being seen.	Modified health bar to position based on sprite size instead of a fixed position.
More Shots + Frost Shots Enemies slowed by multiple arrows that can apply slow were not getting their original movespeed back. Additionally, the slow was stacking.	Introduce a method to reset their movespeed back. Made the character remove all instances of the same status effect instead of just one when refreshing status effects.
Enemy Movement Animation The movement animation of enemies were still playing even when their movement speed was set to 0.	Modified animator to animate based on movement speed instead of using a Vector2 object which we already used for the enemies to move towards the player.
Boss skill casting Bosses were getting interrupted by another skill cast/moving in the middle of casting a different skill.	Added isCasting boolean to make sure the boss does not try to use another skill in the middle of casting. Set move speed to 0 when casting. Added an event at the end of each skill's animation to set isCasting back to false and move speed back to original.
Cockatrice's Icicle Rain	Removed the manual reference to the

Cockatrice's Icicle rain appeared at a random location when casted.	main camera. Added the predefined Camera.main scripting API to reference the main camera.
Crabulon's Slam/Slash Combo Crabulon's Slam and slash damaged the player, but errors appeared in the console saying "OutOfRange".	Physics2D.OverlapCircleAll returns a list, but if nothing is detected, an empty list is returned. Added a check to make sure the list is not empty before reading the index in the list.
Dragonewt's Fire Stomp The player was taking more damage than the amount we intended from some of the fire projectiles.	Add a hash set to make sure that each direction has only one fire projectile firing in that direction as multiple fire projectiles were stacking on top of one another for some of the directions.
Minotaur's Charge When the minotaur ends its charge animation and re-enables its collider while maintaining contact with the player, they can get displaced.	Have the minotaur charge again and away from the player.
Restart + Skills When restarting the game, the player still had their skills from the previous game.	Add a reset method that skills can call when they are initialized.
Warrior's Slam ground effect (New in Milestone 3) Warrior's slam ground effect did not scale with sources of increased AoE. Used events and delegates for this, doesn't work since a new effect is instantiated every time and the effect has a very short duration, the receiver cannot listen to the event.	Changed it to reference instance of warrior to get AoE of slam.
Warrior's interaction with IceSnail and DustElemental (New in Milestone 3) The Warrior's slam attack did not work against IceSnail and DustElemental. The Animation of the slam plays but the ground effect is not played, nor do the enemies take damage.	Changed Ice Snail's shell and Dust Elemental's sand dome to be on the EnemyProjectile layer instead of the Enemy layer. Warrior's slam checks the enemy layer for their Health component but since the aforementioned parts of enemies do not have a Health component, there was an error.
Mage's lightning Dead Zone (New in Milestone 3) Mage's lightning has a dead zone (an area in which no enemies can be damaged) in between the lightning and the mage itself.	We extended the collider of the lightning to close the dead zone that appeared.
Mage's lightning orb skill orbit (New in Milestone 3)	We made the lightning orbs rotate around the centre of the screen instead of around

Picking the lightning orb skill over the course of the game results in the lightning orb orbiting out of control. This is because the centre of the player's position is different based on whether the player faces left or right, meaning the orbs could rotate around different centres resulting in them going out of control.	the player. This means that there is only one centre to rotate around instead of two.

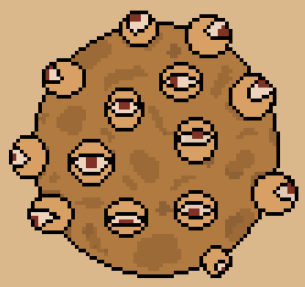
Game Mechanic	Improvements/Balance changes
Collision The box colliders were too restrictive in representing the shapes of our characters, where visually it seemed like projectiles were colliding with thin air.	Usage of polygon colliders instead, which allows us to adjust the collider to fit the shape of our characters.
Enemy Spawning Spawning based on time passed can pose issues for the player if they do not gain sufficient levels to deal with stronger enemies.	Modify spawner to spawn based on player level instead of time passed.
Enemy Movespeed Some of our enemies, especially the blood spider, were too fast based on feedback from milestone 1.	Lowered the movement speed of the faster enemies and shifted them to spawn at a higher level when the player is more equipped to deal with them.
Status Effects Movement based status effects made bosses significantly easier to fight as they were slower in reaching the player to execute melee attacks.	Made bosses immune to movement based status effects.
Getting lost in the game (New in Milestone 3) Testing during Milestone 2 and internal testing afterwards revealed that our game really needs a map.	Added landmarks to the tilemap and a large map accessible by pressing 'M' or clicking on the minimap. The map contains landmarks which players can compare with the actual environment.
Warrior's Regeneration (New in Milestone 3) During playtesting in GDC's Playtesting Session, we received feedback that Warrior's Regeneration skill was too strong. It allowed Players to recover large amounts of health every second.	We changed health regeneration to instead proc every 3 seconds. This will keep regeneration a strong source of health recovery but make it so that the Warrior can be burst down in the 3 second window without regeneration.

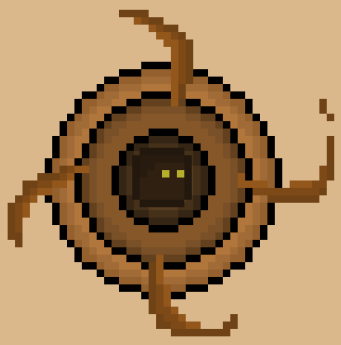
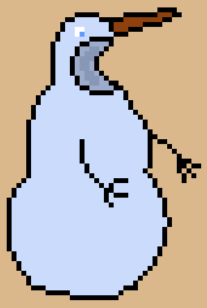
Intuitiveness of UI (New in Milestone 3) During playtesting sessions in Milestone 2, we were also given feedback that the UI may not be intuitive enough.	Hotkeys to the UI in the game are now written with the UI buttons during gameplay. We also added a button into the minimap. Players that click the minimap will open the full map.
Bomb King throwing one bomb (New in Milestone 3) When testing the skill, we realise that the bomb can be thrown past the player if the player is too far from Bomb King since we are using a vector based on the distance between the player and Bomb King.	Equip the bomb with a collider such that it slows down when near the player.

Appendix - Enemies *(Updated for Milestone 3)*

(Sprites are all self-drawn and animated using Aseprite.)

Name	Sprite	Description
Ghost		Malevolent spirit. Behaviour: Chases the player.
Bat		Annoying animal. Behaviour: Chases the player.
Ice Bat		Relative of the bat that lives in the Snowfields. Behaviour: Chases the player.
Ice Slime		Blob of coldness. Behaviour: Chases the player.

Bomber		Mad green creature armed with explosives. Behaviour: Chases the player. Explodes on-death, dealing very high damage.
Ashtoad		Ashen toad with ranged attack. Behaviour: Chases the player. When in range, fire spit attack.
Blood Spider		Vicious bug that moves at extremely high speeds. Behaviour: Chases the player. Very, very quickly.
Blood Mother		Female blood spider that survived to adulthood. Filled to the brim with its babies. Behaviour: Chases the player. Spawns (4-6) blood spiders on death.
Ash Orb		A mysterious orb with psychic powers. Fires rapidly. Behaviour: Chases the player. When in range, fire spit attack.

Dust Elemental		The dust of the Dry Sea comes to life. Behaviour: Chases the player.
Ice Snail		Snail that has adapted to harsh conditions. AI: Chases the player. Shoots icicles in 8 directions periodically. Hides in shell when player is far away.
Snow Thing		Essence of the Snowfields. AI: Hops towards the player.
Rat		A nasty rat. Likes to eat dirt. AI: Chases the player.

Giant Flybug		Disgusting bug that lives in the jungle. AI: Chases the player. Enrages when damaged.
Hail Cloud		Clouds formed from the wet and coldness of the Snowfields. AI: Chases the player. When in range, fire ice cube attacks.
Salamander		Descendants of the ancient dragonewts. AI: Chases the player. When in range, fire lava bubble attacks. lava bubbles create burning ground that lingers for 5 seconds.
Boar		Once domesticated pigs have turned wild. AI: Chases the player. When in range, charges at the player.

Jungle Bat		Annoying jungle animals. AI: Chases the player.
Jungle Spirit/Flame Spirit		Little floating rocks said to be ancestors of the land. AI: Chases the player.
Rotor Blades		Magical rocks with extendable blades. AI: Chases the player. When in range, extend blades which slice up the player.
Demon Warrior		Strange statues that look exactly like the Warrior... AI: Chases the player.
Demon Archer		Strange statues that look exactly like the Archer... AI: Chases the player. When in range, shoots arrows at the player.

Appendix - Skill list *(Updated for Milestone 3)*

Basic Skill Upgrades

Melee Upgrade	**Projectile Upgrade**	**Spell Upgrade**
Impact (1-8) Increase slam Area of effect by (0.5 - 4).	*More shots (1-10)* Adds (1-10) projectiles	*Overreach (1-5)* (5-25%) increased range
Brutality (1-10) Adds (10-100) damage	*Vicious Arrows (1-10)* Adds (10-100) damage	*Arcane Power (1-10)* Adds (10-100) increased damage
Vitality (1-10) Gain (30 - 300) maximum life. Gain (1-5) additional life per second	*Faster Shooting (1-10)* Shorter duration in between shots (0.1s - 1.0s)	*Faster Casting (1-10)* Shorter duration in between casts (0.1s - 1.0s)
Juggernaut (1-5) (1-5)% increased movement speed +(5 - 25) to armour	*Entrench (1-5)* Gain (20 - 100) maximum life. Gain (3-15%) chance to avoid enemy attacks	*Arcane Fortitude (1-5)* Gain (15 -75) maximum life. (5-25%) increased movement speed
Robust (1-5) Gain (2-10) additional life per second	*Piercing Shots (1-5)* Projectiles pierce (1-5) additional targets	*Orbs of Power (1-10)* Level 1 - 4: Gain 1 lightning orb that rotates around you. Level 5 - 10: Orbs deal additional (2-12) damage. Orbs rotate (10 - 60% faster).
	Quickstep (1-5) (5-25%) increased movement speed (5-25%) chance to avoid enemy attacks	*Equivalent Exchange (1-5)* (10-50%) increased damage taken (10-50%) increased damage with spells
	Blood Drinker (1-5) Gain (0.05-0.25) life per enemy hit by your projectiles	*Soul Drain (1-5)* *Gain (0.5-2.5) life when you kill an enemy*

		Lightning Storm (1-10) Every 10 seconds, lightning strikes the ground, creating a lightning field that damages enemies. Level 1: Gain Lightning Storm skill. Level 2-5: Gain (5-20%) lightning field range. Gain (1 - 5s) lightning field duration. Level 6 - 10: Gain (5-25%) lightning field range. Gain (1-5) lightning field damage.

Signature passive skills (Can only get it by reaching level 10, 20, 30)

Warrior Signature Upgrade	**Archer Signature Upgrade**	**Mage Signature Upgrade**
Earthquake The ground trembles after you slam, delivering an aftershock to enemies in range after 1 second. -100 Initial hit damage Aftershock base damage: 300 damage	*Frost Shots* Your arrows now reduce enemy movement speed by -40% by 2s.	*Apex Form* Press "Space" to activate /deactivate Apex Form. While in Apex Form, • Duration in between casts is shortened by 0.5s. • You gain +30 attack. • You gain 20% movement speed. • Lose 5 health per second.
Sanctuary Health regeneration is converted to a percentage of your maximum life.	*Explosive Munitions* Your Arrows can no longer pierce. Your Arrows explode in an area, dealing 150% of your attack damage to surrounding enemies.	*Perfect Storm* Your attacks have a 50% chance to be critical strikes. Critical strikes deal double damage and have +30% to range.
Demon You can no longer passively regenerate health. You now gain lifesteal based on 150% of your health regeneration.	*Unwavering Instinct* You cannot avoid attacks. Your avoidance chance is converted to armour.	*Arcane Protection* Every 4 seconds, gain a shield that nullifies the next attack you receive completely.
Executioner Execute enemies that are below 15% life	*WindDancer* Gain 30% chance to avoid enemy attacks. Restore 10 health when you avoid enemy attacks.	*Glacial Storm* Summons an aura of frost around you that slows enemies by 20% while they are inside.
Frenzy	*The Weight of the Draw*	*Second Spark*

Max and Current health is halved. Future health gains are halved. You can no longer regen health, You are no longer able to see your life bar. 50% increased movement speed. +100 to attack.	Projectiles deal triple damage. Projectiles take 2 more seconds to fire. Enemies hit by projectile attacks are stunned.	Your lightning basic attack now triggers twice. -20% lightning basic attack damage.
Steadfast Regenerate 3 additional health per second. +30 to armour. 30% reduced movement speed.		