M0 - Game Proposal: Soulless 🫀

CPSC 427 – Video Game Programming	*Version 1 - 27.09.2024*

Team Hebbs

Members	Student Numbers
Joshua Fung	91626895
Ricky Li	21973375
Luca Festa-Bianchet	80372634
Ben Vinnick	20038972
Armin Talaie	59500959

📜 Story

⏳ Background

This story takes place in another realm, in the kingdom of Eldoria. While Eldoria was not an easy place to live, its people had always taken pride in being strong and tough. They battled fiercely and sustained themselves on the spoils of their victories. But one day, the king realized that all this fighting was uncivilized; things needed to change. He began to lead quests, in the hopes of finding some other way to secure Eldoria's safety. After some time, these hopes were answered. They came upon a glowing rune, wedged firmly in the ground. The brave king reached out to grab it, and upon doing so, a magical force passed into his soul. He quickly discovered that this force enabled him to control the elements. Eldoria's soil became fertile with plants and fruits, flourishing abundantly. The people rejoiced and celebrated their newfound prosperity.

But all good things must come to an end. One day, the evil mage Zelathar, who had been drifting around from place to place, entered Eldoria. He soon learned of the king's story, but more importantly, of his power. Zelathar made it his sole objective to acquire that power. After months of planning, he was ready. He snuck into the castle at night and used his dark magic to extract the king's soul. It was nearly perfect, but at the last second, a guard spotted him. Zelathar dashed through the underground of the castle to make an escape. As he emerged into the forest, he heard the chants and stomps of a crowd approaching. The people of Eldoria were going to battle once again, determined to recover the soul of their king.

🎮 Objective

You play as Zelathar. Your motivation is to destroy everyone in your way to keep the soul for yourself. Through the soul, you have gained the ability to cast powerful elemental spells. These include fireballs, lightning bolts, and powerful gusts of wind. However, since you acquired the soul with dark magic, it is resistant. Therefore, you do not have full control over your spells; instead, they are conjured in a random cycle. You may also harness spells to restore health, but only as frequently as your body permits.

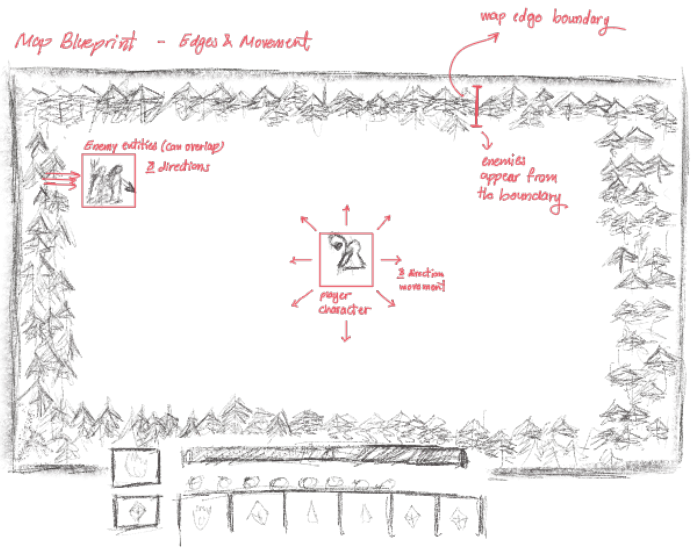
Amongst the multitude trying to stop you are average villagers armed with pitchforks, as well as stronger warriors and archers. Enemies will take damage from spells, potentially resist damage, throw ranged weapons, attack from melee range, and more. They have varying hitpoints, weapons and armour. More will emerge from the trees with time. Finally, there is the Great Knight. He is the champion of Eldoria, and is rumored to have won ten thousand battles. If you can defeat him, your victory will be secured.

Scenes

Produce basic, yet descriptive, sketches of the major game states (screens or scenes). These should be consistent with the game design elements, and help you assess the amount of work to be done. These should clearly show how players will interact with the game and what the outcomes of their interactions will be. For example, jumping onto platforms, shooting projectiles, enemy pathfinding or 'seeing' the player. This section is meant to demonstrate how the game will play and feeds into the technical and advanced technical element sections below. If taking inspiration from other games, you can include annotated screenshots that capture the game play elements you are planning to copy.

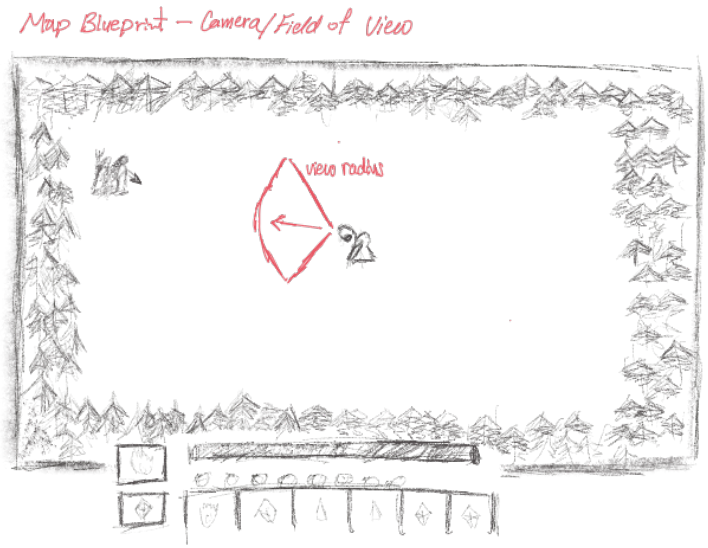
Edges and Movement

- the area the character can move, its directions as well where enemies emerge from + movement.
- Where the character starts (middle of the forest field)



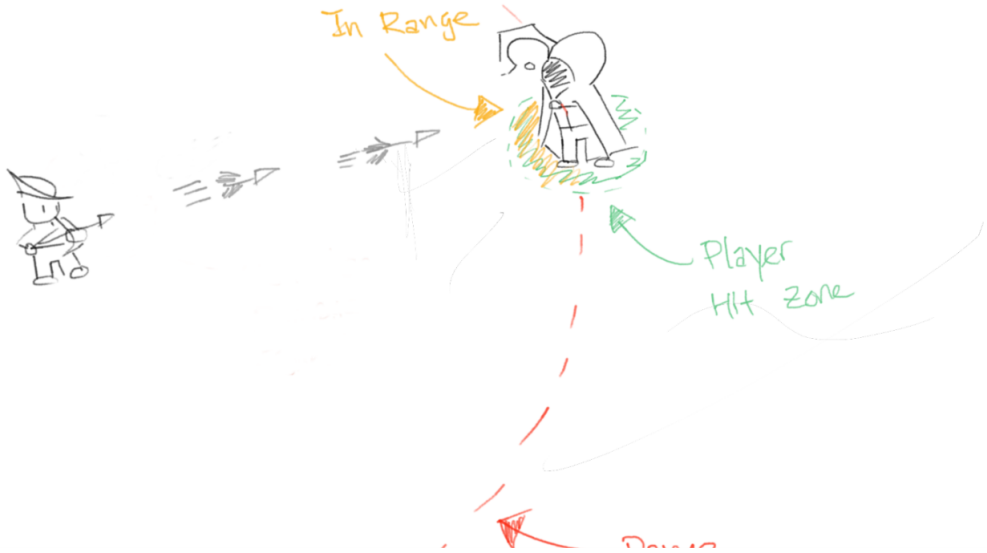
Camera and Field of View

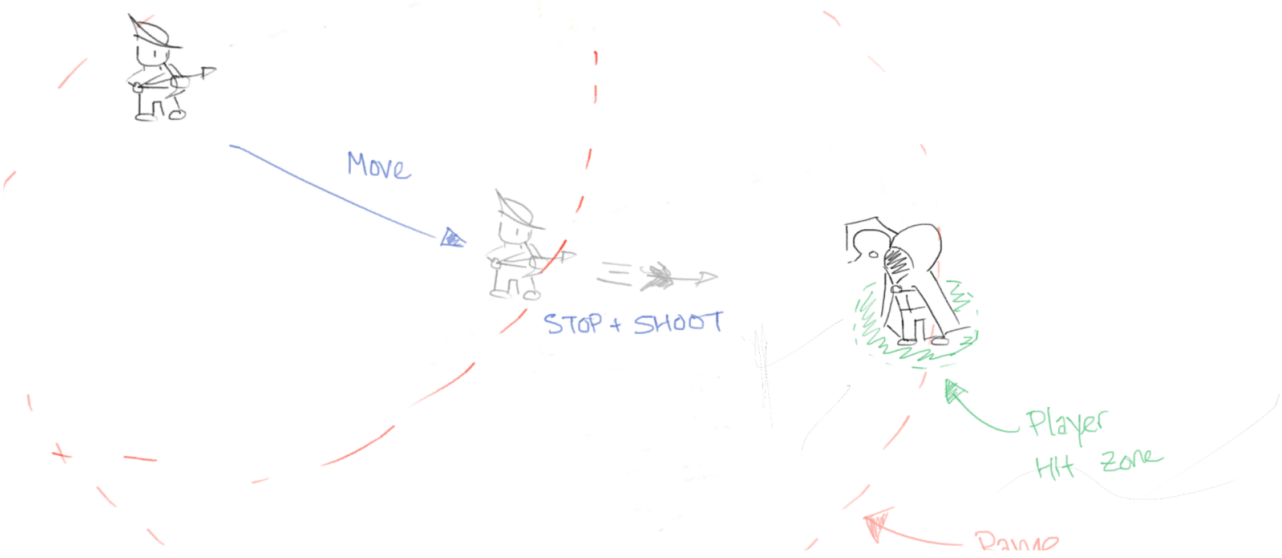
- Detection/Visibility is within a grid-format radius for all entities with possible varying ranges for entity types



Enemy Attack

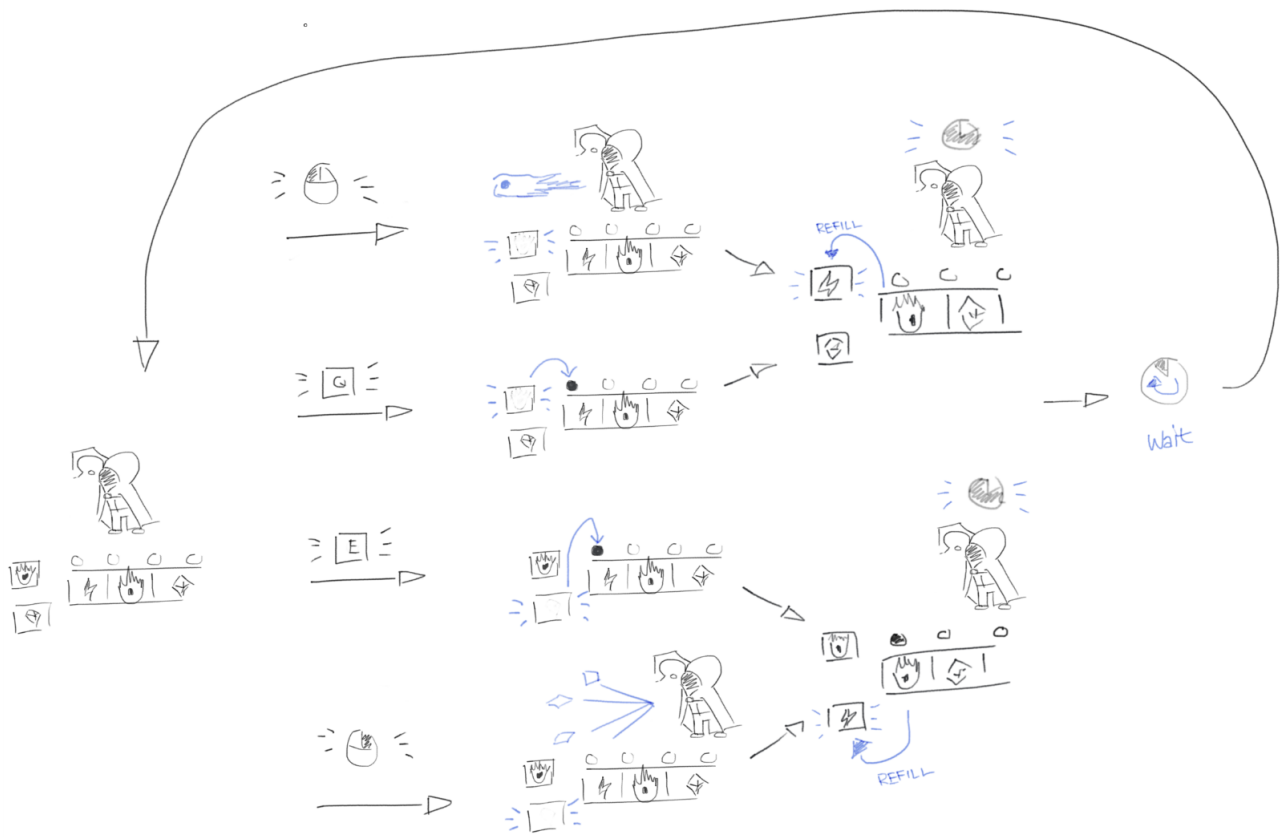
- melee enemy pathing
  - they always move towards player character in a straight line
- ranged enemy pathing
  - they have an effective range where they stop moving once the player is in that range
  - once player is in range they fire in the direction of the player
  - they also will move towards player character in a straight line





Character Spells and Combat Options

- example of what happens after a left click + right click:
  - spell is taken out of hand slot
  - new spell taken off top of queue
  - spell cast by the player character
  - cooldown is incurred



💻 Technical Elements

1. **Rendering:**
    - Spell activation and hits should generate corresponding particles to emphasize their effects
    - Enemies and player getting hit should result in a white blinking effect and red overlay to signify hit
    - Animated entities need to cycle through their animation states
    - Sprites should be rendered based on y-coordinate with entities higher in the screen being rendered behind those that are lower
2. **Assets:**
    - Sprites for main character + different enemy types
    - Sprites for spells
    - Sprites need to be made or found for each of the following:
        - Mage (player character)
        - Enemies
            - Archer (ranged enemy)
            - Villager (melee enemy)
            - Heavy unit (larger melee enemy)
            - Knight (boss enemy)
3. **2D geometry manipulation:**
    - Collision
        - Each collidable sprite will have a collision hitbox to deal with interactions of entities
        - Examples of collidable entities:
            - archer projectiles (arrows)
            - enemies + player
            - spells
            - walls
        - projectiles/spells should cause damage effects to occur between enemies and the player
    - Boundaries of map
        - player and enemy movement should be restricted by the boundaries of the map
    - Transformation of sprites
        - certain enemies will need a coloured highlight to denote element resistance
            - a coloured border will be programmatically added
    - Enemy and player sprites need to change as their direction changes
    - On death, enemies and the player entities need to play an animation and fade out
4. **Gameplay logic:**
    - Player movement with WASD
    - Player aims and casts spell with mouse
    - Player gets new abilities from picking up off ground from enemies
5. **AI:**
    - Melee enemies move in towards player to collide
    - Ranged enemies stay at a distance and attack at their max set range
    - Enemies at a range will try to dodge the player's spells
    - Can predictively aim if player is on the move

- Large melee enemies will move in a more predictive pattern
- Enemies will be programmed to find the shortest path towards the player
- Boss enemies will play towards both ranged and melee behaviours

6. **Physics:**
   - Entity interactions -> Main character can interact with spells to pick them up
   - Spells will have different behaviours and disappear differently - e.g. could knockback enemies
   - Particle effects on spells and hit effects

7. **Sound**
   - Music - speeds up as time progresses
   - Spell cast sound effects
   - Enemy sounds
   - Footsteps/ambiance
   - Hit sounds for enemy attacks and spell hits

Advanced Technical Elements

The previous section covers the core elements of our game. Below are some ideas for advanced technical elements that would be nice to have. We will likely be implementing at least 2 of these as "Creative Components" in later milestones.

- **Swarm behaviour (AI)** - e.g. groups of hounds launch coordinated attacks
- **Cooperative planning (AI)** - e.g. "healer mage" enemy that can resurrect dead enemies, other enemies protect this healer
- **Skinned motion (Graphics)** - for boss enemy
- **Particle systems (Graphics)** - implement system for smoke, fire, lightning
- **Sophisticated integrated assets (Assets)** - to give the game its own unique feel

Potential Adjustments in Scope

We believe that our targets are feasible, but in the case of unexpected difficulties, we have some fallbacks in different game areas. On the other hand, we also have some possible expansions, if we find we are capable and there is enough time.

Spells

Base plan: 3 spells to start - fire, lightning, and wind. Player can unlock a total of 3 new spells by picking them up from fallen enemies.

Fallback: No spell pickups, just starting 3

Expansion: Increase number of unlockable spells to 5-6

Enemies

Base plan: 3 types of enemies (melee, ranged, heavy) + 1 boss battle

Fallback: Simplifying or removing the boss battle, leaving out advanced AI behaviour

Expansion: Increase types of enemies to 5-6, add another boss battle so there is an intermediate and a final boss

Map

Base plan: Single map - forest

Fallback: n/a

Expansion: 3 maps in total, player moves through them as they progress in the game. New maps have different layouts.

Devices

The game’s primary input devices are a mouse and keyboard.

🖱️ Mouse

The mouse allows the player to aim, using Mouse 1 (Left Click) to ‘fire’ their first spell and Mouse 2 (Right Click) to 'fire' their second spell.

⌨️ Keyboard

With the keyboard, the player moves in cardinal and ordinal (intercardinal) directions using the ‘WASD’ keys. The ‘F’ key will be used for interaction events present in the game. The 'Q' and 'E' keys will be used to 'sacrifice' their spell in order to fill up the health gauge, 'Q' will sacrifice the first spell and 'E' will sacrifice the second.

🛠️ Tools

We intend on using the GLFW library to create our game window and the SDL library for sound.

💼 Team management

Since the game pitch video, we have developed a breakdown of what aspects of the game each of us want to work on.

The work is divided into *7* categories:

1. game mechanics
2. movement and interactions
3. game and story
4. rendering
5. visual assets (design)
6. sound design
7. Admin/Project Management
    - *(Related to CPSC 427 project management and deliverables)*

Within each category, the work is broken down into a higher-level tasks of what needs to be implemented or designed, e.g. 'player movement.'

Given many of us have overlapping interests, each higher-level task is spearheaded by one person to break down the task into more granular subtasks. They are responsible for assigning these subtasks to other team members who have interest in working on the same higher-level task. Furthermore, they are expected to create a timeline for when subtasks should be done so that the higher-level task can be completed on time.

Role Types for each area:

- Leads: in charge of the feature/task with lead 1 having more supervision over the feature

Game Mechanics	Lead 1	Lead 2
Time System	Josh	Luca
Health, terminal and start logic	Josh	Luca

Movement and Interactions	Lead 1	Lead 2
Player (mage) movement	Ben	Ricky
Character control + HUD	Luca	Ben
Map collision (Edges of the map and objects)	Ricky	Ben
Entity with Entity collision	Armin	Ricky
Enemy movement + logic (AI)	Armin	Luca
Player (mage) attack/spell mechanics	Josh	Ben

Game and Story	Lead 1	Lead 2
Story, spells, and enemy types	Armin	Ricky
Character “die” and states	Luca	Ben
Collision and spell animations	Armin	Luca
Rendering pipeline	Luca	Armin

Visual Assets (Design)	Lead 1	Lead 2
Game character, and enemies	Ricky	Josh
Map and Mesh	Ricky	Josh

Sound Design	Lead 1	Lead 2
Collision Effects	Ben	Armin
Background/Map + Foliage	Ben	Armin

Admin/Project	Lead 1	Lead 2
Document deliverables	Ben	Ricky
Project management	Josh	Josh

The following is the role distribution.

Role Map	Lead 1 Roles	Lead 2 Roles
Joshua Fung	4	3
Ricky Li	3	4
Luca Festa-Bianchet	3	4
Ben Vinnick	4	4
Armin Talaie	4	3
Median	4	4
Average	3.6	3.6
Total	18	18

📆 Tracking and Internal Deadlines

For tracking and project planning we will be using Github Projects. We will use milestones, issues, and tags with priority labels to efficiently manage and triage our progress.

After each high-level task is broken down into subtasks with a timeline, we will collaborate on putting together one full timeline and discuss adjustments. This complete timeline will align expectations and priorities between all team members.

For internal deadlines and policies, we will try for one-week sprints at the beginning of each week.

We meet three times a week.

- In person on Mondays for 2 hours before class.
- In person during Wednesday tutorials.
- Remote online meetings on Fridays via Discord.

We have **two** deadlines for each milestones:

1. A soft deadline of two days before.
2. A hard deadline of 1 day before the milestone deadline. This is when we are "happy" with the progress for submission but will use the 1 day left for any urgent fixes.

👷 Development Plan

Milestone 1 is fully detailed whereas for future milestones we have outlined the tasks to do but will plan the tickets and project accordingly.

Milestone 1

Week 1 - Completed by September 29

No.	Category	Task	Who
	Gameplay	Add one basic enemy – including its simple sprite and characteristics (e.g. movement speed, health)	Josh
[2]	Rendering	Player position movement in cardinal and intercardinal directions	Ben
[3]	Rendering	Linear interpolation for enemies moving towards the player (from enemy position to player position) using an interpolation formula	Armin
[4]	Gameplay	Keyboard control for player movement (change in player position) and mouse control to aim (change in player orientation, if time permits)	Luca
[6]	Gameplay	Implement basic map and its associated boundaries (along edges of the map)	Ricky, Josh
[7]	Gameplay	Simple collision detection & resolution (e.g. collision with bounding boxes for enemies and player should not be able to walk through map boundaries)	Ricky
[10]	Software Engineering	Progressively add to our test plan (of actions and expected outcomes) as we implement new features	Everyone

Week 2 - Completed by October 6

No.	Category	Task	Who
	Gameplay	Add one basic spell -- including its simple sprite, projectile speed and	Josh
[Creative #1]		Camera should be fixed on the player even as they move around (basic)	Josh, Ricky
[Creative #2]		Implement simple path finding for enemies moving towards the player (basic)	Armin
[1]	Rendering	Player and enemy sprites are displayed over the map When players and enemies overlap, the player's sprite is displayed above the enemy	Armin, Luca
[5]	Gameplay	Randomized spawn location of enemies at some set interval (Enemies should $\approx n$ pixels away from player, every m seconds)	Luca, Ricky

[8]	Stability	Ensure frame rate is stable with no substantial lag	Ben
[9]	Stability	Debug major crashes and glitches	Ben
[10]	Software Engineering	Progressively add to our test plan (of actions and expected outcomes) as we implement new features	Everyone
[11]	Reporting	Write down bug list into Google Sheets	Everyone
[12]	Reporting	3 min. max video of required and creative features	Everyone

• • •

Milestone 2

Week 1 - Completed by October 13

- [1] Gameplay: Game AI, enemies try to run into player and do damage to them upon hitting, enemies spawn more frequently as game progresses
- [3] Gameplay: Map assets (trees, walls, etc) + spell and enemy assets
- [6] Gameplay: Display FPS on window title
- [10] SE: Update test list as we implement additional features

Week 2 - Completed by October 20

- [Creative 1] Basic physics: spell that bounces off enemies (basic)
- [Creative 2] Audio feedback: 3 interactions + progressive background music (basic)
- [2] Gameplay: Sprite sheet animation (8 sprites for enemies, player - North, East, South, West, NorthWest, NorthEast, SouthWest, SouthEast, each spell has a unique amount of sprites but generally have cast, travel, impact)
- [4] Gameplay: Mesh based collision detection and resolution
- [10] SE: Update test list as we implement additional features

Week 3 - Completed by October 27

- [5] Gameplay: Base user tutorial/help (spellbook, movement, etc)
- [7] Playability: Make sure all features work for 2 minutes with no crashes, enemies spawn more frequently as time progresses (non-repetitive)
- [8] Stability: Ensure stable framerate and minimal lag (based on framerate counter)
- [9] Stability: Debug major crashes and glitches
- [10] SE: Update test list as we implement additional features
- [11] Reporting: Update and reprioritize bug list in Google Sheets
- [12] Reporting: 4 min. max video of required and creative features

• • •

Milestone 3

Week 1 - Completed by November 3

- [Creative 1] Reloadability: store enemies + positions, player position + health, game clock (basic)
- [Creative 2] Basic integrated assets: add some background villagers, trees, villages (basic)
- [Creative 3/4] Swarm behaviour in enemies (advanced)

- [3] Robustness: Handle all user input (tabbing out/minimizing window pauses game, hitting invalid keys does nothing or produces some 'error' sound)
- [6] SE: Update test list as we implement additional features

Week 2 - Completed by November 10

- [1] Playability: User can play for 5 minutes where more enemies spawn as well as different types of enemies as the game progresses (non-repetitive)
- [5] Stability: Implement any missing previous milestone features and fixes, game resolution and aspect ratio consistent across machines/displays, no crashes, glitches, or other unpredictable behaviour
- [6] SE: Update test list as we implement additional features

Week 3 - Completed by November 17

- [2] Robustness: Debug and ensure smooth memory management, no leaks
- [4] Robustness: Realtime performance (ensure no input lag or delays)
- [6] SE: Update test list as we implement additional features
- [7] Reporting: Update and reprioritize bug list in Google Sheets
- [8] Reporting: 5 min. max video of required and creative features

Milestone 4

Week 1 - Completed by November 24

- [Creative 1] Game balance: game should be progressive, fun to play, challenging (basic)
- [Creative 2/3] One of: skinned motion, cooperative planning, particle systems (advanced)
- Playability: 10 minutes of non-repetitive gameplay for at least 10 minutes (new enemy, map changes), should be new content for most of the time
- Stability: ensure all prior milestones are completed, all previously identified bugs are fixed, consistent resolution and aspect ratio, and supports continuous execution and graceful termination
- Robustness: proper memory management, can handle expected and unexpected user input, and ensure no input lag or stuttering

Week 2 - Completed by December 1

- Improve user tutorial (as needed based on feedback)
- Update and reprioritize bug list in Google Sheets
- Report on user testing, including user feedback and changes we made in response to feedback
- 6 min. max video about game features, new creative features, any significant bug fixes, and evolution of the game from concept to final product