Name: Ryan Costin Mark ______________________/50

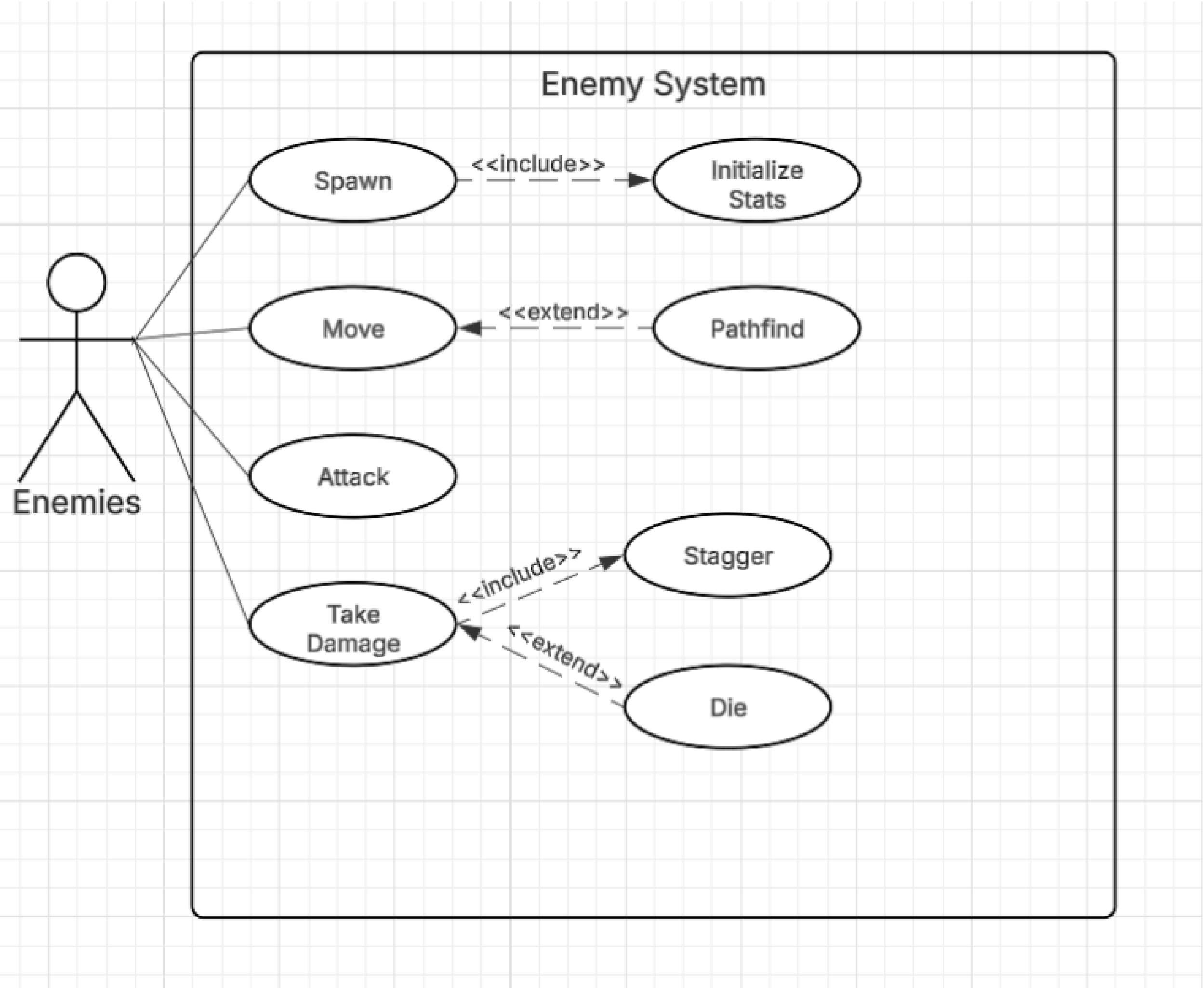
1. Brief introduction __/3

This feature for the videogame, The Crawl, involves integrating enemies that the player will face throughout the dungeon.

When a level is generated, depending on the level of difficulty, different tiers of enemies will engage the player. Enemies will consist of damaging the player or altering their movement whether through melee or range capabilities, and buffing or healing fellow enemies to challenge the player.

Additionally, the rooms within the dungeons will have walls and the enemies will need to be implemented with a pathfinding algorithm to prevent the enemies from getting stuck on walls and other objects.

2. Use case diagram with scenario __14



Scenario 1:

Name: Pathfinding

Summary: The enemy logs its current location and the player's location and computes a route to engage the player while not being obstructed by walls, objects, etc. If the path were to become invalid (i.e. encounters wall, obstacles, or player moving) the pathfinding will recompute a replan.

Actors: Enemies

Preconditions:

- Navigation data (grid/room system) is up-to-date and accurate.
- Enemy has valid start location and goal location.

Basic sequence:

Step 1: Enemy determines current location and player’s location.

Step 2: Enemy measures obstacles using mesh map.

Step 3: Pathfinding computation begins.

Step 4: Enemy receives and follows path.

Step 5: Enemy reaches end of computed path.

Step 6: Pathfinder replanning loop.

Exceptions:

Step 1: Enemy may not find player’s location.

Step 2: Pathfinding may not correctly read mesh map’s obstacles.

Step 3: Computation may not start due to not receiving coordinates.

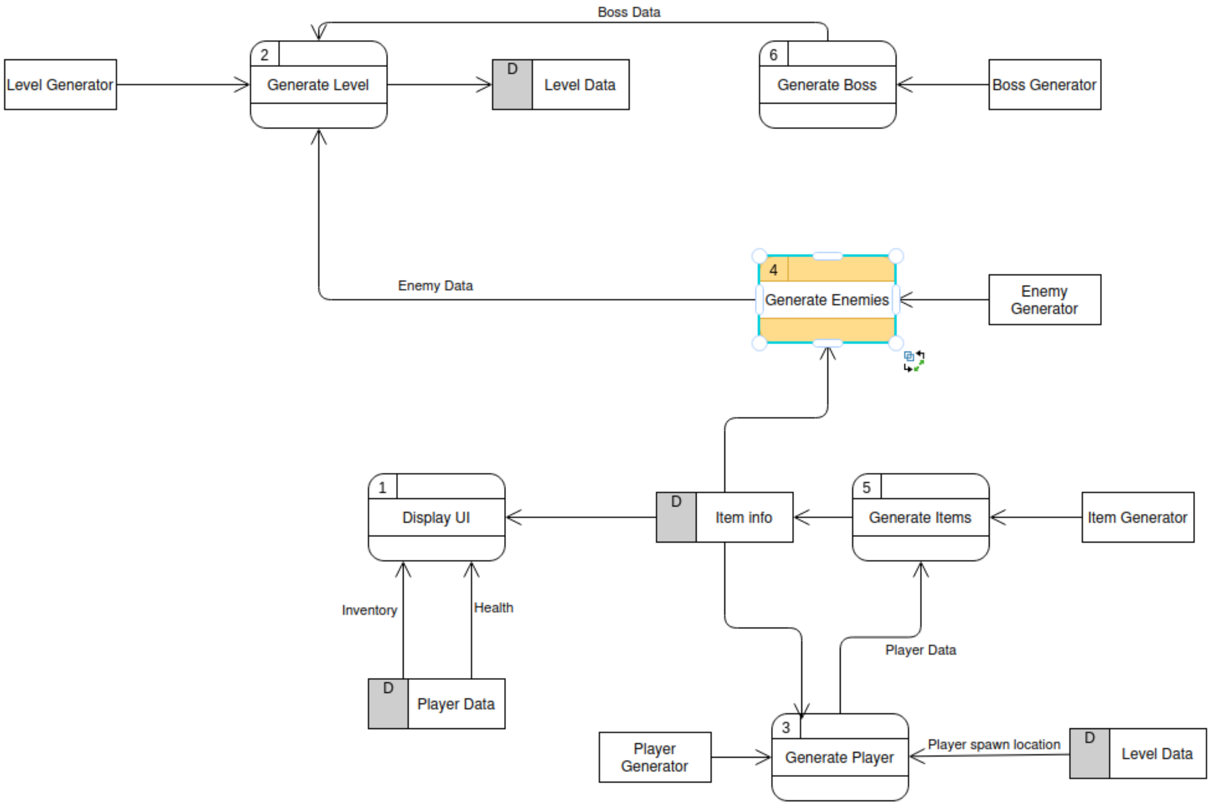
Step 4: Enemy may not receive coordinates, therefore giving it no path to follow.

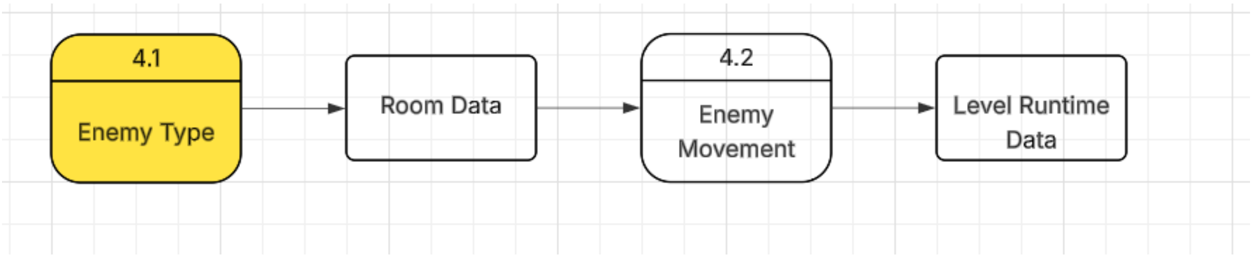
Post conditions: A valid path has formed and enemies may arrive at goal and execute further behaviors.

Priority: 1*

*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

3. Data Flow diagram(s) from Level 0 to process description for your feature ______14





Process Descriptions

Enemy Type:

```
enemyType(int roomLevel, int uniqueEnemy)
if (roomLevel < 3)
if (uniqueEnemyTypesInRoom < 2)
return "Slime";
else
return "Rat";
else if (roomLevel < 7)
if (uniqueEnemyTypesInRoom < 3)
return "Orc";
else
return "Skeleton";
else
if (uniqueEnemyTypesInRoom < 4) {
return "Orc";
else
return "Spitter";
```

4. Acceptance Tests _______9

Example for Pathfinding

Run feature 1000 times sending pass or fail to reaching player’s location if not dead.

- If alive, enemies must reach player’s location
- Sprites must be facing the correct direction of movement
- If colliding with walls/objects, enemies must correct/replan path
- Cannot move through walls/objects
- Path length must be reasonable with no obvious loops

Example for Enemy Attack

Run feature 1000 times with characteristics:

- Attack animation should be facing the direction the enemy is facing player.
- Attack hitbox should overlap with player's hitbox.
- Damage should be applied to player's health if successful.
- Damage should be reduced if player is blocking/dodging.
- Ranged enemies should only attack when inside appropriate range of player.

5. Timeline ________/10

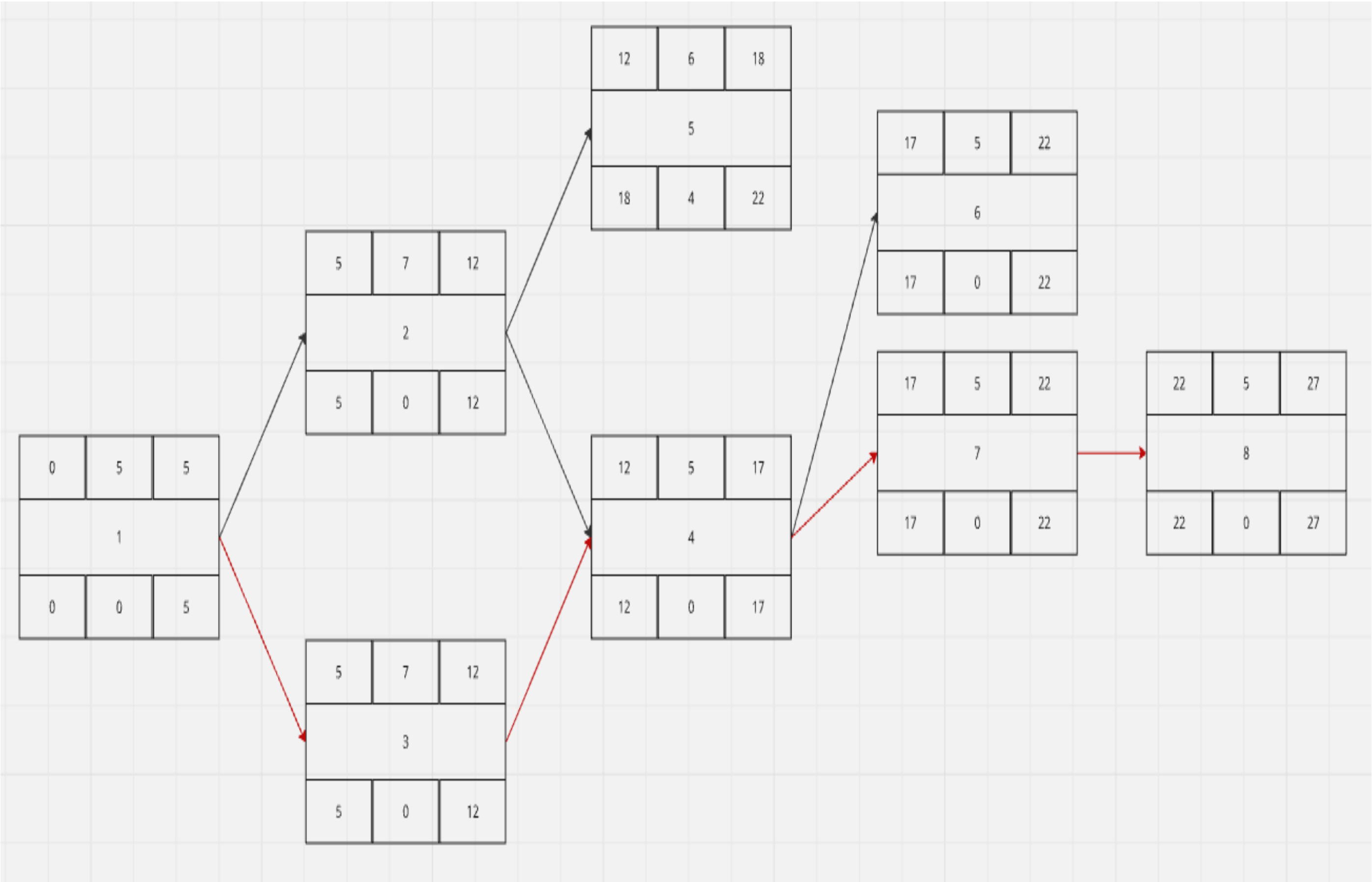
[Figure out the tasks required to complete your feature]

Example:

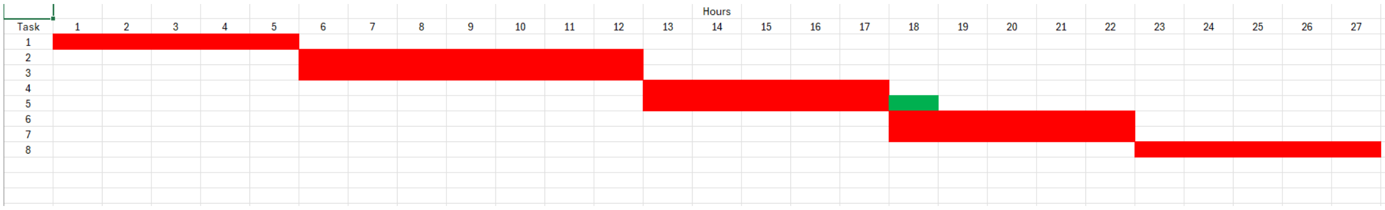
Work items

Task	Duration (Hours)	Predecessor Task(s)
1. Requirements Collection	5	-
2. Enemy Type Programming	7	1
3. Enemy Movement Program	7	1
4. Collision Programming	5	2,3
5. Sprite Animation/Sound	6	2
6. Documentation	5	4,5
7. Testing	5	4,5
8. Installation	5	7

Pert diagram



Gantt timeline


Hours
Task