Name: Nathan Hampton Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Brief introduction \_\_/3

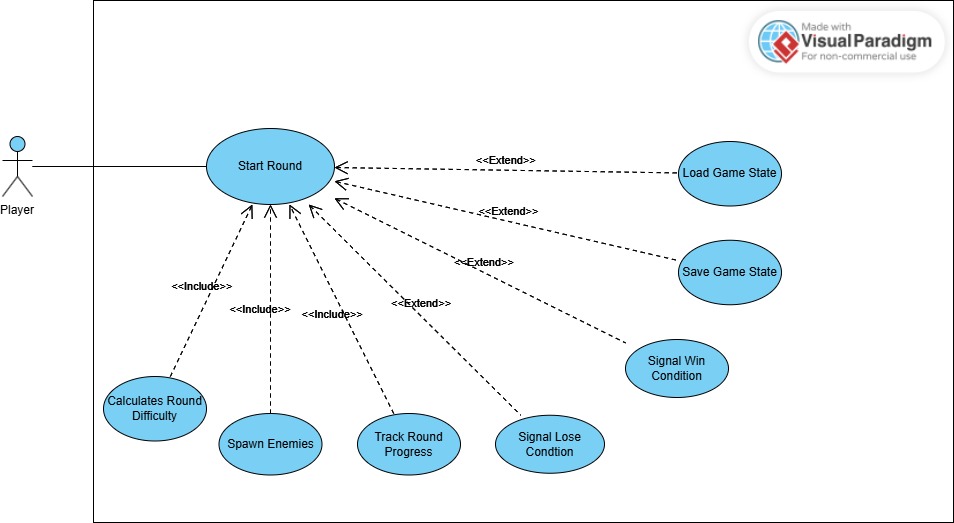
My feature for Kill Mo’ Chickens is the creation and coordination of the Round Manager.

When a Level is loaded and the user wants to start the round, the round manager needs to prepare the round difficultly, begin spawning the enemies on the level, and keep track of when the round is complete. It also needs to be able to increase the difficulty based on how many rounds have passed and what round the player is currently in and let the UI system know when a round is done.

Lastly the round manager must be able to signal to the UI system if the player has completed all the rounds, or if the player has died.

## Use case diagram with scenario \_\_14

### Use Case Diagrams



### Scenarios

**Scenario 1**

**Name:** Start Round

**Summary:** The round manager starts the round by calculating the round difficultly, then spawning the correct type and number of enemies and tracks when all the enemies have been removed to know when the round is over.

**Actors:** Player.

**Preconditions:** A level is loaded. A round is not already running. The game has not been stopped.

**Basic sequence:**

**Step 1:** Calculate the current difficultly of the round based on what round it is.

**Step 2:** Spawn the correct number and type of enemies based on the round difficulty.

**Step 3:** Keep track of how many enemies are left as the round runs.

**Step 4:** End the round when the number of enemies left is zero.

**Exceptions:**

**Step 1:** The player loads a saved game: Load a saved game state’s round number

**Step 1:** The player saves the game: Save the current game state’s round number

**Step 3:** The player’s heath drops to zero: Emit Loss condition signal

**Step 3:** The final round number is reached: Emit Win condition signal

**Post conditions:** Increment the round number by one

**Priority:** 1\*

**ID:** SR1

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

## Data Flow diagram(s) from Level 0 to process description for your feature \_\_\_\_\_\_\_14

[Get the Level 0 from your team. Highlight the path to your feature]

Example:

### Data Flow Diagrams

A screenshot of a diagram

Description automatically generated

### Process Descriptions

Calculate Round Difficulty\*:

IF Round start clicked AND Round number != last round number THEN

Collect Level Difficulty Table

FOR EACH Enemy in Level Difficulty Table[Round Number]

Send enemy class to the enemy spawn queue

END FOR

Emit Round Running Signal

END IF

Round Over\*:

IF Round Running THEN

IF Player Health <= 0 THEN

Set Game Lost State

END IF

IF length of Enemy stats = 0 THEN

Increment round number

END IF

IF round number > Level Rounds:

Set Game Won State

END IF

END IF

## Acceptance Tests \_\_\_\_\_\_\_\_9

**Round Over State Testing**

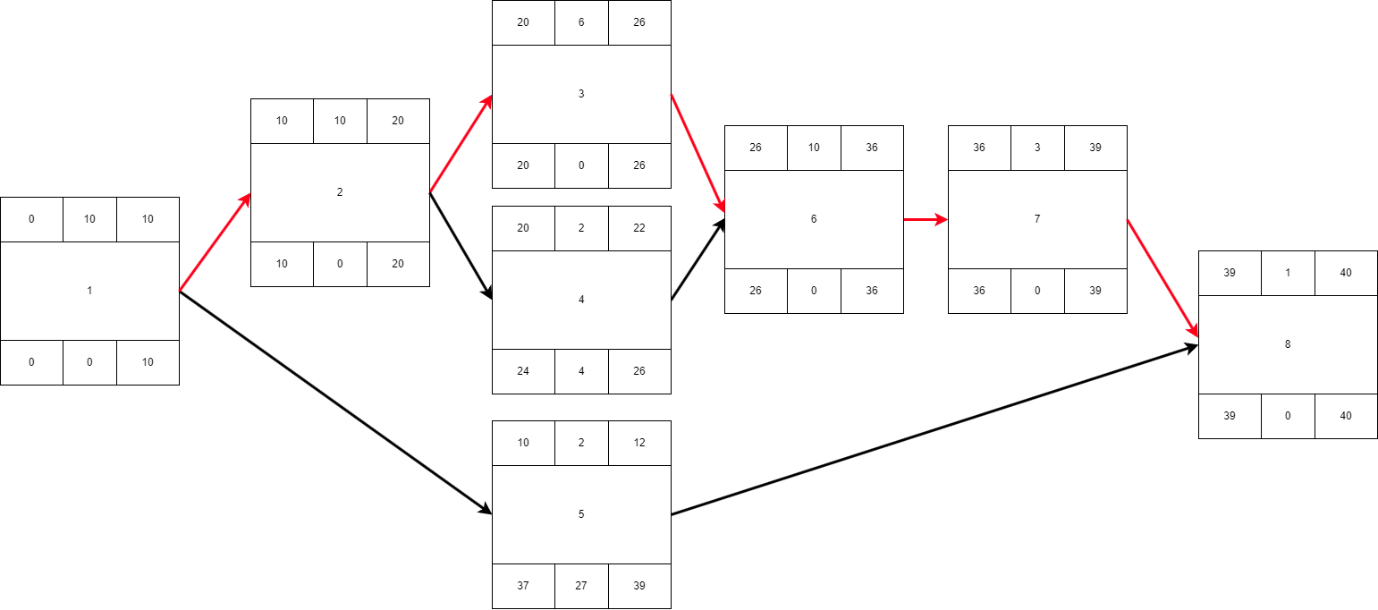
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Outputs | | Inputs | | | | Notes |
| Round # | Win/loss State | Round Number | Player Health | Number Of Enemies | Level Rounds |  |
| 1 | Null | 1 | 100 | 20 | 50 |  |
| 2 | Null | 1 | 100 | 0 | 50 | Round completed with health remaining |
| 1 | Loss | 1 | 0 | 12 | 50 | Round in progress with no health remaining |
| 50 | Win | 50 | 56 | 0 | 50 | Round number at the max value for level |
| 13 | Loss | 12 | 0 | 0 | 50 | All enemies gone and health is 0 is still a loss |
| 50 | Loss | 50 | 0 | 0 | 50 | Loss Condition always overwrites win |

## Timeline \_\_\_\_\_\_\_\_\_/10

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (PHrs) | Predecessor Task(s) |
| 1. Requirements Collection | 10 | - |
| 2. Report Design | 10 | 1 |
| 3. Level Difficultly Tables | 6 | 2 |
| 4. Level Data Object design | 2 | 2 |
| 5. User Documentation | 2 | 1 |
| 6. Programming | 10 | 3,4 |
| 7. Testing | 3 | 6 |
| 8. Installation | 1 | 5, 7 |

### Pert diagram



### Gantt timeline