Name\_\_\_Caleb Mouat\_\_ Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Brief introduction \_\_/3

My feature is the Grid Manager.

The Grid Manager (3) is in charge of moving the cursor, computing which unit was selected based on the cursor’s position, moving the selected unit to a correct space based on the player’s inputs, and finally bundling up information regarding the unit selected, enemy selected and selected action and sending these to the Action Event Manager.

## Use case diagram with scenario \_\_14

### Use Case Diagrams

A diagram of a process

Description automatically generated

### Scenarios

**Name:** Player Selects Action

**Summary:** When moving the cursor, the player can either select a unit or end their turn

**Actors:** Player

**Preconditions:** Game has been Started and Level has been initialized

**Basic sequence:**

**Step 1:** Accept the player’s directional movements and/or “confirm” button press.

**Step 2:** Show movement unit can take based on if cursor was over a unit when conform was pressed.

**Step 3:** Accept directional movements and confirm button press from the player.

**Step 4:** Display updated unit position and list of potential actions.

**Step 5:** Accept directional movements and confirm button press from the player

**Exceptions:**

**Step 1:** If confirm was pressed not over a unit, then display option to end turn.

**Step 2:** If the “back” button is pressed at any point, return to the step prior.

**Post conditions:** Action selected is initiated.

**Priority:** 1\*

**ID:** C01

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

### Data Flow Diagram 0

A diagram of a data flow

Description automatically generated

Grid Manager (3)

A diagram of a company

Description automatically generated

### Process Description

4.1 Unit Movement \*:

**Purpose**: The "Unit Movement" process calculates valid movement options for the player-controlled unit based on input and unit and map characteristics.

**Description**:

1. The process begins by receiving the current cursor position from the cursor class.
2. The cursor class updates this position based on the player's input.
3. The "Unit Movement" process then considers data related to the selected unit, the player's directional input, and the tile information surrounding the unit.
4. It calculates whether the selected unit can move to the player's chosen location. If the selected location is invalid, the process selects the nearest available grid position.
5. Once the unit's movement destination is determined, the process updates the unit's position in the Unit Selector process.
6. Additionally, the process communicates the unit's position and the player's chosen action to the Enemy Selector process.

**Inputs:**

* Cursor Position (from the cursor class)
* Player Inputs and Selection (from the Menu Manager)
* Selected Unit Data and Movement Stat (from the Unit Selector)
* Tile Information (from the Level Manager)

**Outputs:**

* Updated Unit Position (to the Unit Selector)
* Unit Position and Player Action (to the Enemy Selector)

**Dependencies**:

* The process relies on receiving player inputs for accurate calculations.
* It depends on the Level Manager for information about available movement tiles.

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

The Grid Manager is a critical component of our turn-based tactics RPG. It manages various aspects of player interaction with the game world, including cursor movement, unit selection, unit movement, and communication with the Action Event Manager. To ensure the proper functioning of the Grid Manager, a set of acceptance tests has been designed to cover key functionalities and scenarios.

**Acceptance Test Overview:**

The Grid Manager feature combines both pre-determined and dynamic elements to provide a dynamic gameplay experience. While certain aspects, such as cursor movement and player interactions, follow predetermined rules, other elements, such as unit positions and player actions, may vary based on player choices and game state.

Acceptance testing for the Grid Manager primarily focuses on verifying the following aspects:

1. **Cursor Movement Test:**
   * **Scenario:** The player moves the grid’s cursor using arrow keys.
   * **Acceptance Criteria:**
     + The cursor's position updates accordingly in the cursor class and on the screen.
     + The cursor cannot move beyond the grid boundaries.
2. **Unit Selection Test:**
   * **Scenario:** The player positions the cursor over a unit and selects it using the confirm button.
   * **Acceptance Criteria:**
     + **T**he selected unit is correctly identified based on the cursor's position and input from the Unit Manager.
     + The unit is highlighted or otherwise indicated as selected.
3. **Unit Movement Test:**
   * **Scenario:** The player selects a unit and provides valid directional input (e.g., arrow keys) to move it.
   * **Acceptance Criteria:**
     + The selected unit visually moves to the desired space on the grid.
     + Invalid movement attempts (e.g., moving through obstacles) are handled by moving the unit to the nearest legal space.
4. **Action Bundling Test:**
   * **Scenario:** The player selects a unit, moves it, and chooses an action (e.g., attack).
   * **Acceptance Criteria:**
     + Information regarding the selected unit, enemy (if applicable), and chosen action is correctly sent to the Action Event Manager and are properly displayed.
5. **Boundary Testing:**
   * **Scenario:** The player attempts to move the cursor and units to various grid edges and corners.
   * **Acceptance Criteria:**
     + The cursor is reset inside the bounds of the map.
6. **Error Handling Test:**
   * **Scenario:** The player performs actions that could result in errors (e.g., selecting an enemy unit, attempting an impossible move).
   * **Acceptance Criteria:**
     + Any non-player unit selection is ignored.
     + An impossible move is corrected to the nearest legal space. The player is informed that an illegal move was attempted.
7. **Performance Test:**
   * **Scenario:** The player performs multiple unit selections and movements within a short timeframe.
   * **Acceptance Criteria:**
     + The Grid Manager handles rapid player input without lag or performance issues.
8. **Integration Test:**
   * **Scenario:** The Grid Manager communicates with the Action Event Manager to trigger actions (e.g., combat), the Unit Manager to correctly pull unit information, the Menu Manager to correctly pull player inputs a.
   * **Acceptance Criteria:**
     + Verify that the Grid Manager effectively communicates with the Action Event Manager, triggering actions like combat.
     + Confirm that the Grid Manager correctly interfaces with the Unit Manager to retrieve unit information.
     + Ensure that the Grid Manager accurately collaborates with the Menu Manager to obtain player inputs and selections.
     + Assess the Grid Manager's coordination with the Level Manager to calculate and execute unit movements.
     + The Grid Manager handles any bad inputs by using default values.

**Example for cursor movement test**

|  |  |  |
| --- | --- | --- |
| Input | Expected Output | Notes |
| UP arrow  (or equivalent) | Cursor position transforms (x, y+1) | The cursor should move one grid unit up. |
| DOWN arrow  (or equivalent) | Cursor position transforms (x, y-1) | The cursor should move one grid unit down. |
| RIGHT arrow  (or equivalent) | Cursor position transforms (x+1, y) | The cursor should move one grid unit to the right. |
| LEFT arrow  (or equivalent) | Cursor position transforms (x-1, y) | The cursor should move one grid unit to the left. |
| UP arrow  (or equivalent) at top boundary | Cursor position remains at (x,y) | The cursor does not move |
| DOWN arrow  (or equivalent) at lower boundary | Cursor position remains at (x,y) | The cursor does not move |
| RIGHT arrow  (or equivalent) at right boundary | Cursor position remains at (x,y) | The cursor does not move |
| LEFT arrow  (or equivalent) at left boundary | Cursor position remains at (x,y) | The cursor does not move |

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

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (PWks) | Predecessor Task(s) |
| 1. Grid Class and Functions | 2 | - |
| 2. Player Input to Move Cursor | 2 | 1 |
| 3. Cursor Movement and Animation | 5 | 2 |
| 4. Unit Movement script and functions | 4 | 3 |
| 5. Combatant Selector | 6 | 4 |
| 6. Unit Selector | 4 | 4 |
| 7. Joint Selector | 2 | 5, 6 |
| 8. Testing | 8 | 7 |

### Pert diagram

A diagram of a mathematical equation

Description automatically generated with medium confidence

### Gantt timeline

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  | 2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  | 3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |  |  |  | 4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |  |  |  | 4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 5,6 |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 7 |  |  |  |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |