Name\_\_\_Ben Murray\_\_ Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Brief introduction \_\_/3

My feature for this game includes the creation of all in-game menus, buttons, and other graphical UI/UX elements, ensuring a seamless and intuitive interaction for the end user.

The purpose of my involvement is to provide the means that allow the end player to interact with the game in every way possible. This includes communicating units’ statistics, abilities, equipment, actions, and other necessary information to the player in a clean and effective manner, providing the interactable menus and buttons to perform various actions and events with units, and allowing the player to interact with the game’s natural lifecycle, such as pausing the game, modifying the game’s settings, continuing once a game has ended, and beginning the game in the first place.

In essence, I am responsible for marrying the game’s intricate codebase with the end user to allow them to interact and utilize the game’s “under the hood” logic using the device that the end user is equipped with, such as a keyboard and mouse, or a controller.

## Use case diagram with scenario \_\_14

### Use Case Diagrams (Next page)

A notebook with a diagram

Description automatically generated

### Scenarios

**Name:** Select unit

**Summary:** When a unit is selected, either by being clicked on or hovered over with the mouse, that unit’s statistics, attributes, and possible actions need to be displayed in order for that information to be conveyed to the player.

**Actors:** Player

**Preconditions:** There be a unit that belongs to the player and is selectable.

**Basic sequence:**

**Step 1:** If the unit is hovered over but not clicked on, a small tooltip will be displayed over the unit containing the most essential information about that unit that would be relevant to the player.

**Step 2:** If the unit is clicked on, there will be a graphic menu element that opens containing the full statistics for that unit (such as their health, movement speed, attack power, etc.) as well as the actions that the unit can take.

**Step 3:** Using the opened unit menu, their available actions will be able to be selected to perform their corresponding action.

**Step 4:** The opened unit menu will have an ‘X’ button in the top right corner to close the menu. Alternatively, if the player’s cursor is clicked outside of the menu except on the same unit or another unit, then the menu will close. If the cursor is clicked on a different unit, the menu will remain open, but the menu’s information will be updated to be for the newly selected unit.

**Exceptions:** None

**Post conditions:** The unit performs an action and/or the unit menu is closed.

**Priority:** 1\*

**ID:** C01

**Name:** Combat predication prior to engagement

**Summary:** When a selected unit chooses the ‘Attack’ action and selects an enemy unit to attack, before the two units engage in combat a menu will open giving a brief prediction of what the combat would theoretically result in, as well as allowing the player to choose the unit’s weapon that they will use in combat. This will help inform the player on if choosing to engage in combat would be an optimal decision.

**Actors:** Player

**Preconditions:** There be a unit that belongs to the player that is selected.

**Basic sequence:**

**Step 1:** The selected unit’s ‘Attack’ action button is clicked.

**Step 2:** The player chooses which enemy unit that the selected unit can engage in combat with, assuming there is one or more enemies that can be fought.

**Step 3:** When the enemy unit is selected, the combat predication menu will open.

Step 4: The player may choose to change the unit’s weapon that they will use in combat, potentially affecting the predicted result of the combat engagement.

**Step 5:** If the player wishes to still engage in combat upon reviewing the combat predication information, they will press the ‘Attack’ confirmation button. If they do not wish to engage in combat, they will press the ‘Cancel’ button to close the menu or click anywhere outside of the combat predication menu to also close the menu.

**Exceptions:** None

**Post conditions:** The units engage in combat or combat is cancelled before it’s initiated.

**Priority:** 1\*

**ID:** C01

**Name:** Game paused

**Summary:** At any point throughout the game, the player can pause the game, bringing up the pause menu and preventing any actions or events to happen in the game until the pause menu is closed and the game resumes.

**Actors:** Player

**Preconditions:** The game is actively being played in some capacity.

**Basic sequence:**

**Step 1:** The player presses a button on their choice of device (keyboard, controller, etc.) to pause the game and open the game’s pause menu.

**Step 2:** The pause menu will have five (potentially more) buttons that can be clicked; ‘Resume’, ‘Help’, ‘Settings’, ‘Quit to Main Menu’, and ‘Quit to Desktop’.

**Step 3:** If the ‘Resume’ button is clicked, the pause menu closes and the game resumes.

**Step 4:** If the ‘Help’ button is clicked, a new menu will be swapped to that provides helpful instructions and tips for the player to use. The player may select the ‘Go back’ button to return to the normal pause menu.

**Step 5:** If the ‘Settings’ button is clicked, a new menu will be swapped to that details settings that the player can change and save to their liking, such as game volume, key bindings, and others. The player may select the ‘Confirm’ button to apply the changes, or the ‘Cancel’ button to cancel any changes and return to the normal pause menu.

**Step 6:** If the ‘Quit to Main Menu’ button is clicked, the current game is quit prematurely, and the player is taken back to the game’s main menu.

**Step 7:** If the ‘Quit to Desktop’ button is clicked, the entire application closes.

**Exceptions:** None

**Post conditions:** The game is either resumed or quit.

**Priority:** 1\*

**ID:** C01

**Name:** Game ends

**Summary:** When the game finishes, the player will have either won or lost; so, a menu dependent on their success or failure needs to be displayed.

**Actors:** Player

**Preconditions:** The game ends.

**Basic sequence:**

**Step 1:** The game ends, with either the player winning or losing.

**Step 2:** Regardless of the result, the player can either choose to restart the game or quit to the game’s main menu.

**Exceptions:** None

**Post conditions:** The game is restarted, or the player quits to the game’s main menu.

**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

Menu Manager (1) A paper with a diagram

Description automatically generated

### Process Description

* 1. Start New Game: Initializes and starts a new instance of the game.
  2. Play Game: The general state of playing the game normally without directly interacting with UI elements.
  3. Open Unit Stats Menu: Opens a selected unit’s details menu.
  4. Choose Attack Action: The player selects the button to attack an enemy unit with the currently selected unit.
  5. Select Enemy Unit to Attack: The player selects an enemy unit within range for their currently selected unit to attack.
  6. Open Combat Prediction Menu: Opens another menu showing a predication for what combat between the selected unit and selected enemy unit would result in.
  7. Calculate Combat Predication: Calculates the results that are displayed in the Combat Predication Menu.
  8. Change Weapon: In the Combat Predication Menu, the selected unit can swap their current weapon before combat commences.
  9. Close Menu: An opened menu is closed.
  10. Open Pause Menu: The player presses a button on their keyboard to pause the game and open up the Pause Menu.
  11. Quit Game: Closes the game application.
  12. Game Ends: Occurs when the current game ends its gameplay loop, with either the player winning or losing.

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

The extensiveness of testing for my specific tasks would be fairly rudimentary, I believe. The only interaction that the player can have with my various menus and other UI elements is clicking and hovering. The logic for the actions and events that correspond to the menus’ buttons in which the player will directly interact with should already be completed and tested in isolation. Therefore, when a player presses a button or otherwise interacts with a UI element to perform some action, the only testing that will need to be certified for my work is that the click event registers and performs it’s required purpose through completion.

The only logic whose test I would be responsible for is for modifying the player’s game settings in the ‘Settings’ menu. Currently, the only settings that are planned to be included in this menu are a slider to control the game volume (perhaps separating it into multiple sliders for audio elements such as music, SFX, etc.) and the ability to remap the player’s input bindings. I’m unsure what automated tests that Unity provides for testing audio, such as some sort of decibel detector, but that’s something that can be experimented with later. The input binding test(s) would need to be able to access any other tests that utilize the player’s input and rerun the test with the remapped input bindings to verify that they are receiving and sending the correct inputs.

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

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (PWks) | Predecessor Task(s) |
| 1. Main Menu | 6 | - |
| 2. Textboxes | 2 | - |
| 3. Pause Menu | 4 | 1 |
| 4. Help Menu | 4 | 3 |
| 5. Settings Menu | 8 | 3 |
| 6. Win Screen | 4 | 1 |
| 7. Lose Screen | 4 | 1 |
| 8. Unit Menu | 6 | - |
| 9. Unit Tooltip | 2 | 8 |
| 10. Combat Predication Menu | 4 | 8 |

### Gantt Timeline & Pert diagram

A notebook with yellow squares and numbers

Description automatically generated