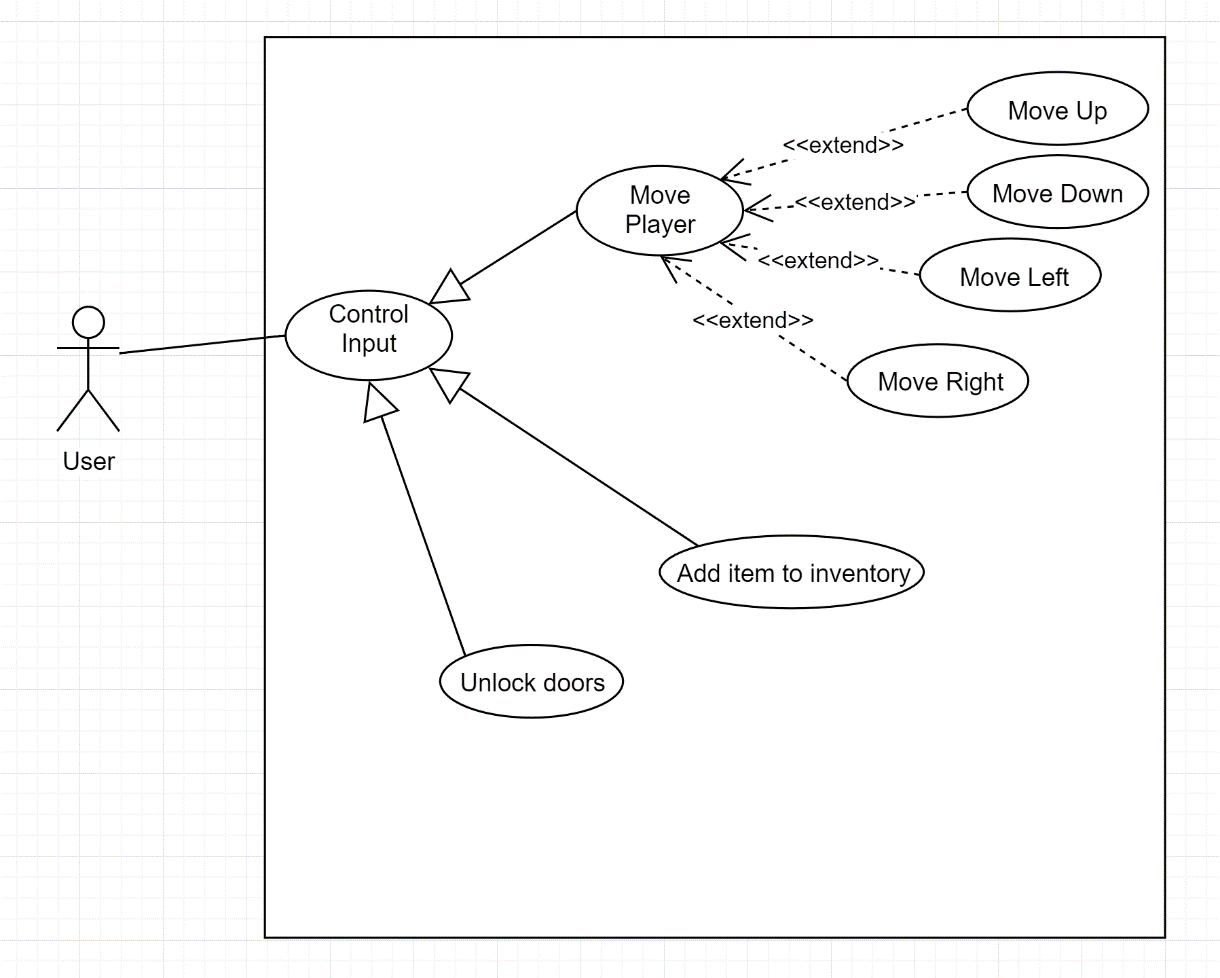
Name: Brad Whitesell Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Brief introduction \_\_/3

I will be championing the player design component of Chasing the End. This component includes the player movement and control functionality, player interaction with items, player interaction with enemy AI units, player interaction with the environment, and any graphical functionality that relates to the player.

## Use case diagram with scenario \_\_14

### Use Case Diagrams



### Scenario 1

**Name:** Move player

**Summary:** The user enters control input that moves the player.

**Actors:** User

**Preconditions:** Scene has been initialized.

**Basic sequence:**

**Step 1:** Accept control input for moving the player.

**Step 2:** Continue moving the player until control input is discontinued.

**Exceptions:**

**Step 1:** User attempts to move their player out of scene bounds: ignore input.

**Step 2:** User attempts to move their player into static scene objects: ignore input.

**Post conditions:** Player model is transitioned to a new location in the scene.

**Priority:** 1\*

**ID:** P01

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

### Scenario 2

**Name:** Add item to inventory

**Summary:** The user enters control input for adding an item to inventory.

**Actors:** User

**Preconditions:** Scene has been initialized and the player has been moved within pickup range of an item.

**Basic sequence:**

**Step 1:** Accept control input for adding an item to the player’s inventory.

**Step 2:** Remove item from the scene.

**Step 3:** Add item to the player’s inventory.

**Exceptions:**

**Step 1:** User is out of the acceptable range for picking up an item: ignore input.

**Post conditions:** Item has been removed from the scene and added to the player’s inventory.

**Priority:** 2\*

**ID:** P02

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

### Scenario 3

**Name:** Unlock doors

**Summary:** The user enters control input for unlocking a door in the scene.

**Actors:** User

**Preconditions:** Scene has been initialized, the player has been moved within interaction range of the door, and the user has the appropriate unlocking item.

**Basic sequence:**

**Step 1:** Accept control input for unlocking the door.

**Step 2:** Remove the appropriate unlocking item from the player’s inventory.

**Step 3:** Open the door and transition to the next scene.

**Exceptions:**

**Step 1:** User is out of the acceptable range for unlocking the door: ignore input.

**Step 2:** The user does not have the appropriate unlocking item: prompt the user to find the correct item and ignore input.

**Post conditions:** The associated unlocking item has been removed from the players inventory, the door has opened, and the game has transitioned to the next scene.

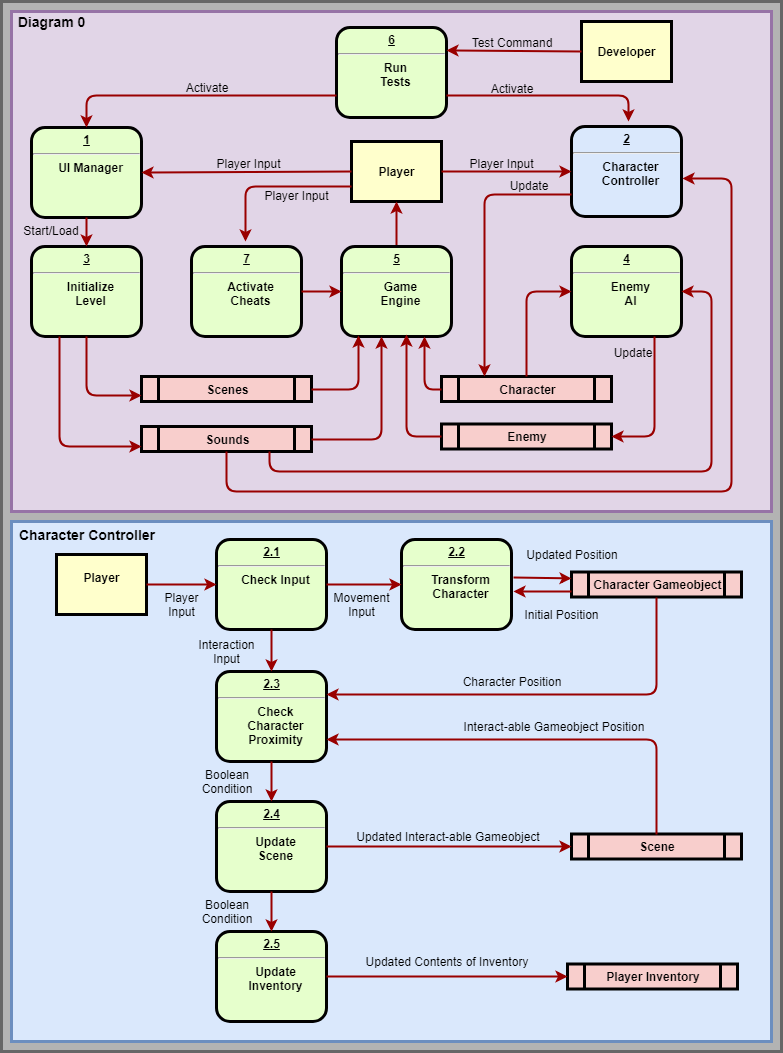
**Priority:** 2\*

**ID:** P03

\*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

### Data Flow Diagrams



### Process Descriptions

2.1 Check Input (Parameters: Player Input):

IF Player Input is movement control input

Call the Transform Character process

ELSEIF Player Input is interaction control input

Call the Check Character Proximity process

ENDIF

2.2 Transform Character (Parameters: Movement Input):

WHILE there is Movement Input

Update the position of the character gameobject

END WHILE

2.3 Check Character Proximity (Parameters: Interaction Input, Character Position, Interact-able Gameobject Position):

IF Character Position is within range of Gameobject Position

RETURN True

ELSE

RETURN False

ENDIF

2.4 Update Scene (Parameters: Boolean Condition):

IF Boolean Condition is True

Update the scene with interaction and RETURN True

ELSEIF Boolean Condition is False

Do nothing and RETURN False

ENDIF

2.5 Update Inventory (Parameters: Boolean Condition):

IF Boolean Condition is True

Update player inventory to reflect interaction

ENDIF

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

**Test:** Verify “move up” control input transforms player to a location north of their previous location.

**Parameters/Input:** Move up control input.

**Output:** Player should be transformed to a new location with an increased Y-coordinate value.

**Test:** Verify “move down” control input transforms player to a location south of their previous location.

**Parameters/Input:** Move down control input.

**Output:** Player should be transformed to a new location with a decreased Y-coordinate value.

**Test:** Verify “move right” control input transforms player to a location east of their previous location.

**Parameters/Input:** Move right control input.

**Output:** Player should be transformed to a new location with an increased X-coordinate value.

**Test:** Verify “move left” control input transforms player to a location west of their previous location.

**Parameters/Input:** Move left control input.

**Output:** Player should be transformed to a new location with a decreased X-coordinate value.

**Test:** Verify any “move” control input is ignored when the user attempts to move the player out of scene bounds.

**Parameters/Input:** Move up control input, move down control input, move right control input, move left control input.

**Output:** Player should remain in their previous location- adjacent to the scene bound.

**Test:** Verify any “move” control input is ignored when the user attempts to move the player into a static object.

**Parameters/Input:** Move up control input, move down control input, move right control input, move left control input.

**Output:** Player should remain in their previous location- adjacent to the static object.

**Test:** Verify “item pick-up” control input removes the item from the scene.

**Parameters/Input:** Item pick-up control input.

**Output:** The “picked-up” item should not exist in the current scene. There should be no graphical, audio, or physical representation left of the previously existing item in the current scene.

**Test:** Verify “item pick-up” control input adds the item to the player’s inventory.

**Parameters/Input:** Item pick-up control input.

**Output:** The “picked-up” item should now exist in the player’s inventory view.

**Test:** Verify “item pick-up” control input is ignored when the player is outside the acceptable range for picking up the item.

**Parameters/Input:** Item pick-up control input.

**Output:** The “picked-up” item should still exist in the current scene. The “picked-up” item should not exist in the player’s inventory view.

**Test:** Verify “door interaction” control input transitions to the next scene.

**Parameters/Input:** Door interaction control input.

**Output:** The user should be faced with the next game scene.

**Test:** Verify “door interaction” control input removes the appropriate unlocking item from the player’s inventory.

**Parameters/Input:** Door interaction control input.

**Output:** The unlocking item should be removed from the player’s inventory view.

**Test:** Verify “door interaction” control input is ignored when the player is outside the acceptable range for unlocking the door.

**Parameters/Input:** Door interaction control input.

**Output:** Player should remain in their previous location, within the same scene. The player’s inventory view should be unchanged.

**Test:** Verify “door interaction” control input is ignored when the player does not have the appropriate/associated unlocking item.

**Parameters/Input:** Door interaction control input.

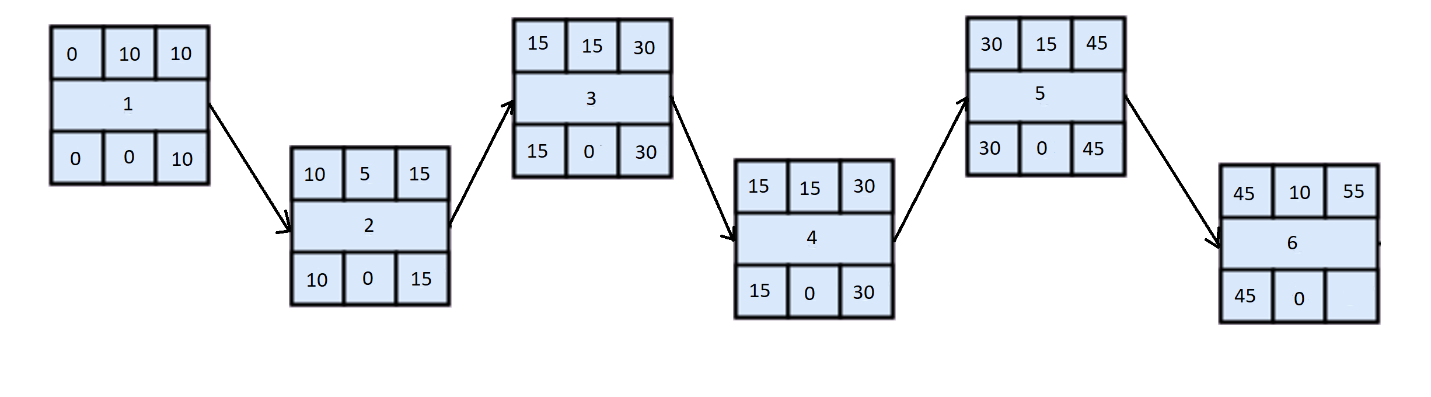
**Output:** Player should remain in their previous location, within the same scene. The player’s inventory view should be unchanged. The player should have been prompted to find the correct unlocking item.

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

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (Hrs) | Predecessor Task(s) |
| 1. Requirements Collection | 10 | - |
| 2. Setup Player Gameobject | 5 | 1 |
| 3. Create Movement Controller | 15 | 2 |
| 4. Create Interaction Controller | 15 | 2 |
| 5. Unit Testing | 15 | 3,4 |
| 6. Integration Testing | 10 | 5 |

### Pert diagram



### Gantt timeline

