Name: Triston Hardcastle Peck Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Brief introduction \_\_/3

My feature is the Zombie (the player). My feature involves player movement, taking damage, a healing system around consuming corpses, and an ability or power up system and how it affects the player.

## Use case diagram with scenario \_\_14

[Use the lecture notes in class.

### Use Case Diagrams

Diagram

Description automatically generated

### Scenarios

**Name:** Interact with NPC store

**Summary:** The player uses the interact button to interact with the NPC Store

**Actors:** Player

**Preconditions:** Player and NPC store have been initialized.

**Basic sequence:**

**Step 1:** Player Moves in the direction of the NPC store.

**Step 2:** Player Uses the interaction button to interact with NPC Store

**Step 3:** NPC store opens.

**Step 4:** Player chooses what to do at NPC store.

**Exceptions:**

**Step 1:** Wrong input is given.

**Step 2:** No interactions take place until player is in range of the NPC Store.

**Post conditions:** NPC Store menu is opened and player presented with options.

**Priority:** 1\*

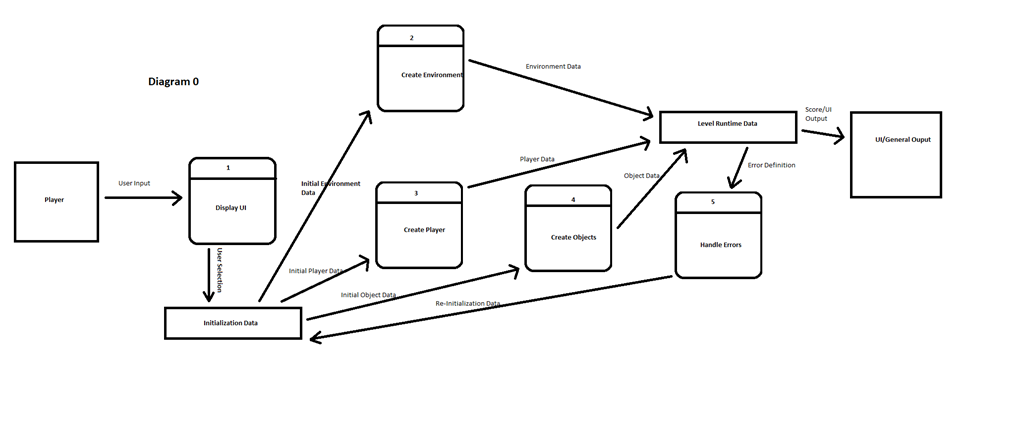
**ID:** THP 1

\*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

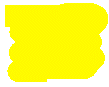
[Get the Level 0 from your team. Highlight the path to your feature]

### Data Flow Diagrams





### Diagram Description automatically generated



### Process Descriptions

Initialize Player:

Initial player Stats are sent to the Player Stat Data storage. Data Storage sends initial health info to the “Check player health” process, the initial upgrades to the “check player upgrades” process and initial attack to the “Check player attack power” process. The “Check player health process returns the health info to the data storage. “Check player Upgrades” returns the Player upgrades to the data storage. “Check Player Attack Power returns the attack number to the data storage.

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

[Describe the inputs and outputs of the tests you will run. Ensure you cover all the boundary cases.]

My feature includes monitoring the status of the player as well as the interactions of the player between the environment and enemies.

For my first test I will be testing the damage calculation between an enemy and the player. I will be simulating the player receiving an attack from an enemy and ensuring that the lost health from the player’s health pool is equivalent to the attack stat of the enemy attacking. The player will be positioned inside of the enemy’s attack range where the player character will then receive an automated attack. There will be a function that automatically compares the predetermined attack stat of the enemy and the change in player health from maximum. After each iteration the Players health will be restored to full.

For the second test I will be checking if the game properly ends once the Player’s health reaches 0. For this test I will be subtracting an incrementing number starting at 1 from the players total health pool, until the number is 3 times that of the players maximum health. This is to ensure that once the Players health goes below 0 and into the negative numbers that the game handles it as a game over.

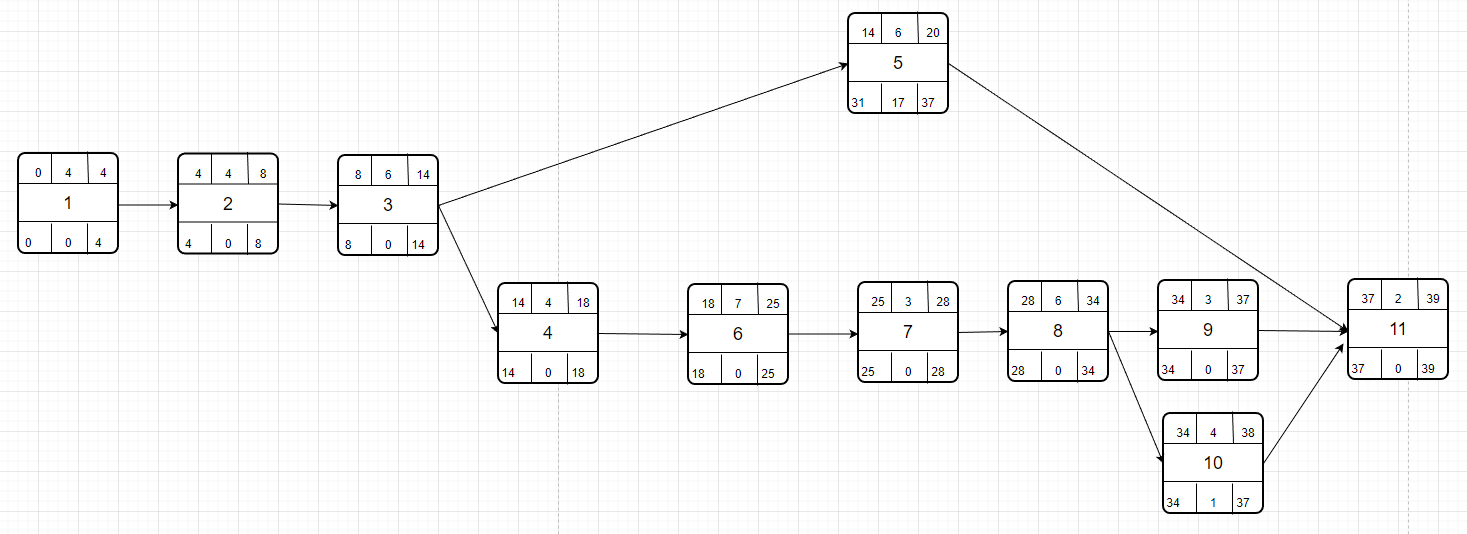
## Timeline \_\_\_\_\_\_\_\_\_/10

[Figure out the tasks required to complete your feature]

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (Hours) | Predecessor Task(s) |
| 1. Requirements Collection | 4 | - |
| 2. Player Creation | 4 | 1 |
| 3. Movement programming | 6 | 2 |
| 4. Stat data programming | 4 | 3 |
| 5. Collision System | 6 | 3 |
| 6. Attack System | 7 | 4 |
| 7. Damage calculation | 3 | 6 |
| 8. Player Upgrade System | 6 | 7 |
| 9. Documentation | 3 | 8 |
| 10. Testing | 4 | 8 |
| 11. Installation | 2 | 9,5 |

### Pert diagram



### Gantt timeline

