**Safal Shrestha**

**1. Brief Introduction (3 pts)**

This feature implements the Player (Orange Ninja) and Enemy system within the colorful rogue-lite game. The Player must be able to move, attack, interact with objects, upgrade stats, and equip weapons. Enemies will scale in difficulty, attack when the player is in range, and drop items when defeated. This system forms the core gameplay loop, enabling combat, progression, and interaction between the player and game world.

**Use Case Diagram with Scenario (14 pts)**

**Use Cases:**

* Move Player
* Attack Enemy
* Pick Up Item <<include>> “Update Stats”
* Enemy Attacks Player <<extend>> “Player Dies”
* Enemy Drops Item <<include>> “Spawn Loot”

A diagram of a game

AI-generated content may be incorrect.

1. **Scenario: Move Player**

* Name: Move Player
* Summary: The Player moves the Orange Ninja around the dungeon map.
* Actors: Player
* Preconditions:
* Game world is loaded.
* Player character is spawned.
* Basic Sequence:

1. Player presses movement keys (WASD/arrow keys).
2. System updates Player position on the map.
3. Camera follows Player position.

* Exceptions:
  + Step 2: Player tries to move into a blocked tile → Movement is stopped.
  + Step 2: Player collides with interactable → Movement stops and interaction is possible.
* Postconditions: Player’s position is updated.
* Priority: 1 (must have)

1. **Scenario: Attack Enemy**

* Name: Attack Enemy
* Summary: Player damages an enemy until it is defeated and drops loot.
* Actors: Player, Enemy
* Preconditions:
* Player and Enemy are within the same level.
* Player has at least a default attack.
* Basic Sequence:

1. Player moves into range of Enemy.
2. Player presses attack button.
3. System calculates damage using equipped weapon stats.
4. Enemy health decreases.
5. If Enemy health > 0, return to Step 2.
6. If Enemy health ≤ 0, Enemy despawns.
7. Enemy drops item(s).

* Exceptions:
  + Step 2: Player out of range → No damage applied.
  + Step 2: No weapon equipped → Default punch/kick animation plays.
* Postconditions: Enemy removed, loot available.
* Priority: 1 (must have)

1. **Scenario: Pick Up Item**

* Name: Pick Up Item
* Summary: Player collects items dropped by enemies or placed in the level.
* Actors: Player, Game System
* Preconditions: Item exists in the level.
* Basic Sequence:

1. Player moves near item.
2. Player presses interact key.
3. Item is removed from map and added to inventory.
4. Stats are updated if the item grants bonuses.

* Exceptions:
  + Step 2: Player inventory is full → Item not collected.
  + Step 2: Item despawns before collection → Nothing happens.
* Postconditions: Player inventory updated.
* Priority: 2 (essential)

**4. Scenario: Update Stats**

* Name: Update Stats
* Summary: Updates the Player’s health, attack power, defense, or abilities after collecting an item.
* Actors: Player, Game System
* Preconditions: Player picks up an item or gains an upgrade.
* Basic Sequence:

1. System checks item type.
2. If it’s a consumable, update health/energy.
3. If it’s a weapon, equip and adjust damage values.
4. If it’s armor, update defense/resistance values.

* Exceptions:
  + Item data invalid → No update applied.
  + Stats at maximum → Item has no effect.
* Postconditions: Player stats updated.
* Priority: 2 (essential)

1. **Scenario: Enemy Attacks Player**

* Name: Enemy Attacks Player
* Summary: Enemy damages the Player when inside its battle radius.
* Actors: Enemy, Player
* Preconditions:

1. Player is within enemy’s detection range.
2. Enemy is alive and active.
3. Basic Sequence:
4. Enemy detects Player entering battle radius.
5. Enemy moves towards Player.
6. Enemy performs attack.
7. Player health decreases.

* Exceptions:
  + Player dodges → Attack misses.
  + Player has shield → Damage reduced/blocked.
* Postconditions: Player health updated.
* Priority: 1 (must have)

1. **Scenario: Player Dies**

* Name: Player Dies
* Summary: Player health reaches 0 after taking enemy damage.
* Actors: Player, Game System
* Preconditions: Player health > 0 before attack.
* Basic Sequence:

1. Enemy attack reduces Player health.
2. If Player health ≤ 0, trigger death sequence.
3. Show death animation and game over screen.

* Exceptions:
  + Player has revive item → Health restored, death canceled.
* Postconditions: Player cannot continue unless revived or new game started.
* Priority: 1 (must have)

**7. Scenario: Enemy Drops Item (Included Use Case)**

* Name: Enemy Drops Item
* Summary: When defeated, the Enemy drops loot that can be picked up.
* Actors: Enemy, Game System
* Preconditions: Enemy health ≤ 0.
* Basic Sequence:

1. System selects item from drop table.
2. Item spawns in level at enemy’s death location.

* Exceptions:
  + Drop chance is 0% → No item spawned.
  + Enemy flagged as “boss” → Guaranteed drop.
* Postconditions: Item is now collectible by Player.
* Priority: 2 (essential)

**Data Flow Diagram**

A diagram of a game

AI-generated content may be incorrect.

**Process Description:**

*Create Player(){*

*If (level data is accessed && level is loaded properly){*

*Spawn the Player character (Orange Ninja);*

*Load Player stats (HP, attack, defense);*

*Initialize Player inventory;*

*}*

*}*

*Access Movement System(){*

*If (Player character is loaded properly && Player gives valid input){*

*Execute movement in the given direction;*

*Update Player position on the map;*

*}*

*}*

*Player Attack Enemy(){*

*If (attack input is given && Enemy is in range){*

*Calculate Damage = Weapon Damage + Buffs - Enemy Defense;*

*Apply Damage to Enemy HP;*

*If (Enemy HP > 0){*

*Continue combat loop;*

*}*

*Else{*

*Trigger Enemy Death();*

*}*

*}*

*}*

*Enemy Attack Player(){*

*If (Player is within Enemy battle radius){*

*Execute Enemy attack animation;*

*If (Player is inside hitbox){*

*Calculate Damage = Enemy Attack - Player Defense;*

*Apply Damage to Player HP;*

*If (Player HP <= 0){*

*Trigger Player Death();*

*}*

*}*

*}*

*}*

**Acceptance Tests**

* Test 1 – Player Attacks Enemy
  + Input: Attack button pressed in range.
  + Expected Output: Enemy health decreases by weapon damage.
* Test 2 – Enemy Dies
  + Input: Enemy health reduced to 0.
  + Expected Output: Enemy despawns, loot spawns.
* Test 3 – Enemy Attacks Player
  + Input: Player enters radius.
  + Expected Output: Player health decreases.
* Test 4 – Boundary Case: Attack Out of Range
  + Input: Attack button pressed outside range.
  + Expected Output: No damage applied.
* Test 5 – Multiple Enemies
  + Input: Player attacked by two enemies at once.
  + Expected Output: Both enemies can deal damage independently.

**TimeLine**

|  |  |  |
| --- | --- | --- |
| Task | Duration | Predecessor |
| 1. Define Player/Enemy Requirement | 1 | - |
| 1. Implement Player Movement and Stats | 2 | 1 |
| 1. Implement Attack System | 4 | 2 |
| 1. Implement Enemy AI and Behavior | 4 | 2 |
| 1. Intergrade Combat Loop | 3 | 3, 4 |
| 1. Add Item Drop and Stats Upgrade | 2 | 5 |
| 1. Test Edge case (multiple input..) | 1 | 6 |
| 1. Final Debugging and Documentation | 2 | 7 |

**Pert Chart**

**A diagram of a number

AI-generated content may be incorrect.**

**Gantt Chart**