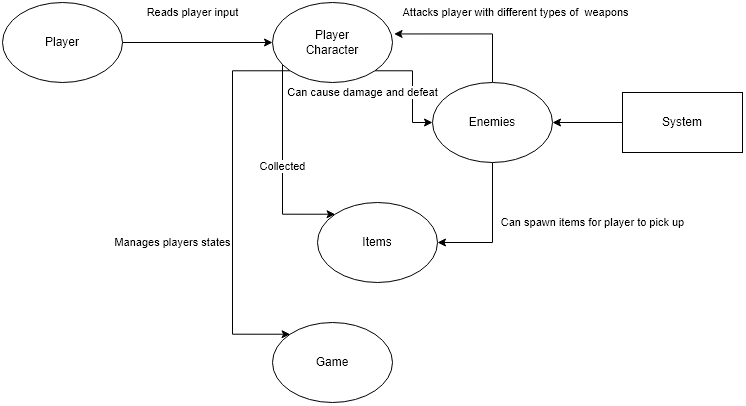
Jakob Guadagno

Software Design Document

**Requirements table**

|  |  |
| --- | --- |
| Requirement Number | Requirement |
| R1[1] | When user pushes “right” character moves right with respect to camera. |
| R2[1] | When user pushes “up” character moves forward with respect to camera. |
| R3[1] | When user pushes “left” character moves left with respect to camera. |
| R4[1] | When user pushes “down” character moves towards the camera. |
| R5[1] | When user presses “fire” weapon in hand with be used |
| R6[1] | When user presses “inventory” inventory UI opens on screen |
| R7[1] | Inventory items that the player collects can be used |
| R8[1] | Enemies will fight player when in range of player |
| R9[1] | Inventory items can be “traded” or “broken down” for different items |
| R10[2] | Multiple weapons to equip for user to use (pistol, axe, shotgun, machine gun, etc.) |
| R11[2] | Enemies can move with AI elements using the Nav-Mesh in unity |
| R12[1] | Lootable chests in the world for the player to interact with. |
| R13[2] | Lootable chests with inventory items that the player can use |
| R14[2] | Doors that are interactable for the user to open and close |
| R15[3] | Enemies will be able to interact with doors (opening/closing) |
| R16[1] | Exits will be marked for user to progress to the next stage |
| R17[2] | Interactable world map for the player to use to progress |
| R18[3] | Enemies will be able to be looted by player after being defeated |
| R19[3] | Item will be able to interact with other items in the player inventory |

**Use Case Diagram**



**Use Case Scenarios (At least for each arrow in the Use Case Diagram)**

|  |
| --- |
| UCS-1 Player Movement |
| Use Case: Player Character |
| Preconditions:   1. Player must be alive 2. Player must be free to move without collisions |
| Actions:   1. Player will input movement directions 2. Left and right will move character left and right from player perspective 3. Up will move player forward from camera perspective 4. Down will move player towards camera |
| Postconditions:   1. Player will move in the direction relative to camera with constant speed |

|  |
| --- |
| UCS-2 Player Attack |
| Use Case: Player Character |
| Preconditions:   1. Player must be alive 2. Player must be using a weapon 3. Player must be in range 4. Enemy must be alive |
| Actions:   1. Player will use the fire button to attack towards mouse direction |
| Postconditions:   1. Player will fire an attack in the direction of the mouse pointer 2. If player hits enemy, enemy health decreases with corresponding weapon damage value |

|  |
| --- |
| UCS-3 Enemy Attack |
| Use Case: Enemy |
| Preconditions:   1. Player must be alive 2. Enemy must see player 3. Enemy must be in range 4. Enemy must be alive |
| Actions:   1. Enemy will attack player if in range |
| Postconditions:   1. Enemy will deal damage to player dependent on what type of enemy |

|  |
| --- |
| UCS-4 Enemy Drop Item |
| Use Case: Enemy Death |
| Preconditions:   1. Enemy must be dead |
| Actions:   1. Enemy will drop interactable chest when dead |
| Postconditions:   1. Enemy chest will spawn items corresponding to enemy type |

|  |
| --- |
| UCS-5 Player Pick-Up |
| Use Case: Player Character |
| Preconditions:   1. Player must be alive 2. Player must be in range of interactable 3. Player must have inventory room |
| Actions:   1. Player will choose what item to take from a menu to keep |
| Postconditions:   1. Player will then store item in their inventory |

|  |
| --- |
| UCS-6 Enemy Movement |
| Use Case: System/Enemy |
| Preconditions:   1. Enemy must be alive 2. System must be running |
| Actions:   1. Enemy will be assigned a type of AI 2. AI will be then run by the system |
| Postconditions:   1. Enemy will be moved accordingly by the system |

|  |
| --- |
| UCS-7 Player/Game State |
| Use Case: Player Character/Game |
| Preconditions:   1. Player must be alive 2. Player must be in the level |
| Actions:   1. Player actions will affect the game variables of the player and other entities |
| Postconditions:   1. Player variables will be changed as the game progresses |

**Requirement Fulfillment Chart (Modified for readability)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **UCS-1** | **UCS-2** | **UCS-3** | **UCS-4** | **UCS-5** | **UCS-6** | **UCS-7** |
| **R1-4** | **R5** | **R8**  **R15** | **R18** | **R12**  **R13**  **R18**  **R19**  **R10**  **R7**  **R9**  **R6** | **R11** | **R14**  **R20** |

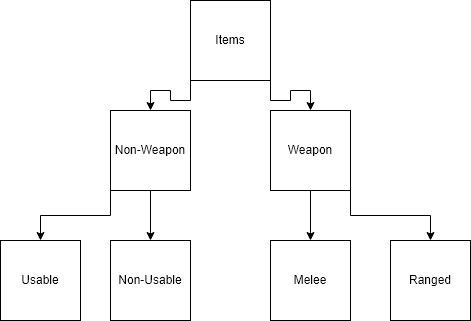
**Module Relation Diagram**

**Enemies:**

Diagram

Description automatically generated

**Items:**



**Class Diagram**

|  |
| --- |
| Player |
| private PlayerInput MyInput;  private CharacterController controller;  private LineRenderer aimLine;  private Vector3 playerVelocity, playerPos;  private bool groundedPlayer  private bool isDragging  private Vector2 dir, cDir, sDir, mLinePos, pLinePos, mousePos2D  private Ray rayLine  private float gravityValue  private Vector3 move, mousePos3D, clickPosStart, clickPosEnd, forwardCamera, rightCamera, fmov, rmov, camRelMov  private float playerSpeed  private float smoothSpeed  private int pHealth  private int pScore  public Camera cameraPlayer;  public float cameraP  public float ySpeed  private Vector2 cameraYV2temp, cameraYRef, cameraYV2  private float zSpeed  private float maxZ  private float minZ  private float cameraZ  private float cameraY  private float scroll, mouseMovementX  public string weaponInHand |
| private Vector3 movePlayerfromCam(Vector3 move)  private void aimLineFunc()  private void dragCameraFunc()  private void cameraPosFunc()  private void playerMovement()  public void Firer(InputAction.CallbackContext context)  public void Dragger(InputAction.CallbackContext context)  public void Scroller(InputAction.CallbackContext context)  public void Mover(InputAction.CallbackContext context)  public void OpenInventory(InputAction.CallbackContext context)  public void Inventory()  void OnTriggerEnter(Collider collision)  void FixedUpdate()  void Update()  void Start()  void Awake() |

|  |
| --- |
| Enemy |
| Public float health  Public string weapon  Public bool enemyAI  Public string enemyAIType |
| void FixedUpdate()  void Update()  void Start()  void Awake()  void OnCollisionEnter(Collision collision)  public void enemySee()  public void enemyAttack()  public void enemyAI() |

|  |
| --- |
| Chest |
| public string itemName  public string itemType  public string itemDescription  public bool isUsableItem  public Sprite itemIcon  public int amountOfItem  public Transform interactionSpot;  public float radiusOfInteraction = 3f; |
| void Update()  void Start()  void Awake()  void chestInventory() |