**Software Requirements and Design Document**

**For**

**Group 8**

Version 2.0

**Authors**:

Zak K.

Taj B.

Ash C.

Taylor R.

John M.

# Overview (5 points)

* Our group is creating a video game that falls into the genre of a metroidvania. There will be a map the player can explore, find coins, and interact with NPC’s. The goal for the player is to ascend through the various levels and reach the very top.

# Functional Requirements (10 points)

* ***Door Requirements***

1. Door Collision: The system should be able to recognize when the Player has walked through the door.
2. Room Transition: The system should be able to correctly transition from room to room according to the specifications on that specific door node.

* ***Enemy Requirements***

1. ENEMY Collision: The system should be able to recognize that the enemy character has entered the user’s hitbox. (HIGH)
2. ENEMY State Machine: The system should be able to accurately switch the enemy’s state correctly and in a sensible manner (i.e., idle state, patrol state, action state, etc.) (HIGH)
3. ENEMY Enhanced Mechanics: The system should allow the user to interact with enemy in a unique way (i.e., ride an enemy across a gap, turn an enemy into a weapon) (LOW)

* ***UI Requirements***

1. The Main Menu must allow the player to start and exit the game by clicking the respective buttons. (HIGH)
2. The Pause Button must pause the game and bring up the Pause Menu. (HIGH)
3. The Pause Meny must allow the player to continue from where they paused or return to the Main Menu. (HIGH)
4. The menus must adapt to different screen resolutions and aspect rations, maintaining consistent layout and usability. (LOW)
5. Health Bar shows the amount of health the player currently has. (HIGH)
6. Coin counter keeps track of the number of coins the player has picked up. (HIGH)
7. Implement an interactable inventory for the player to keep track of their items. (LOW)

* ***Player Requirements***

1. PLAYER Movement: User interacts with the player by using appropriate keys (L & R arrows plus spacebar for jump). (HIGH)
2. PLAYER Collision: Player plays appropriate death animation once collided with enemy. (HIGH)

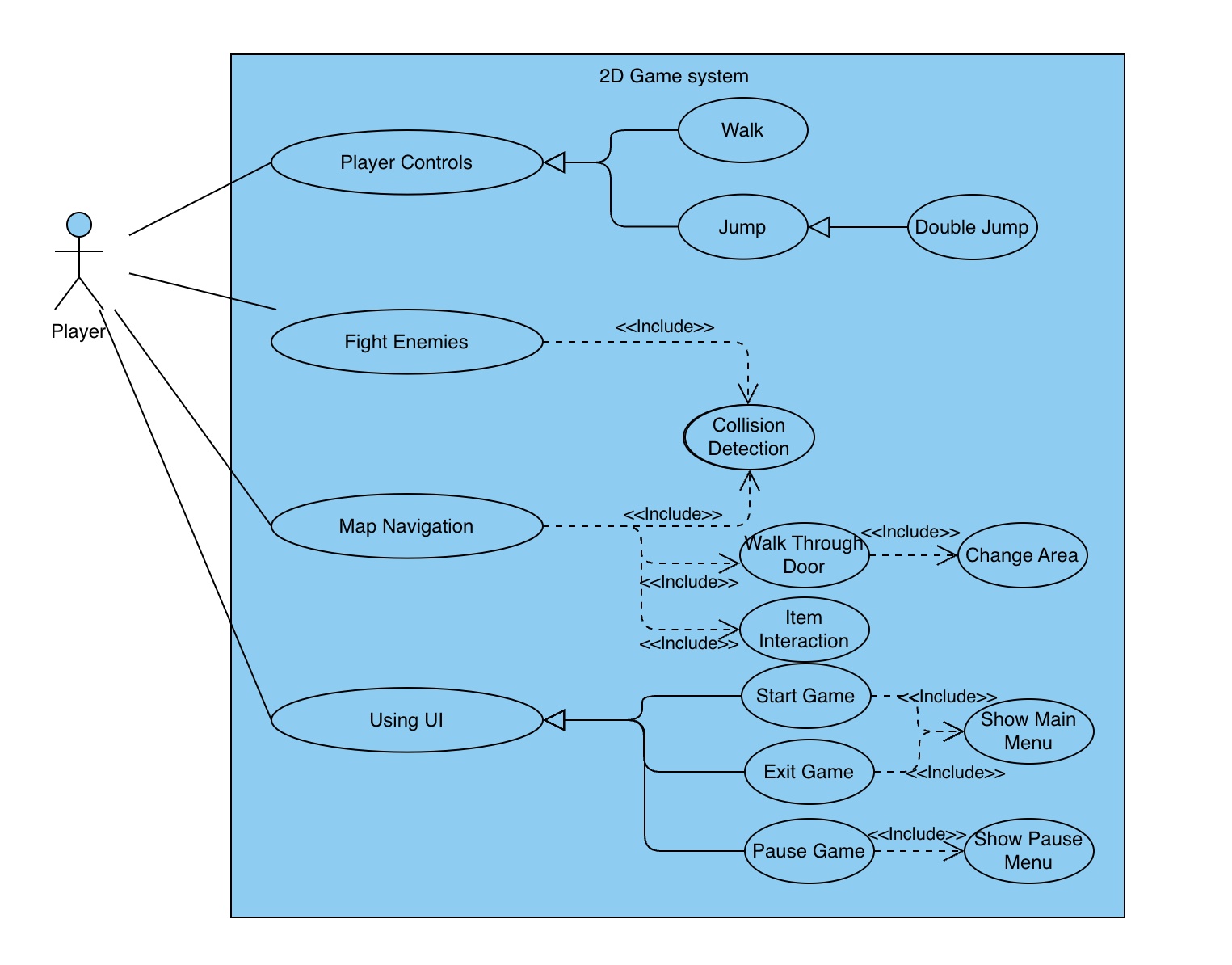
* ***Map Requirements***

1. MAP Navigation: The player should be able to walk around the map, with map collision properly set to handle player movement and interactions with items. (HIGH)
2. MAP Tileset: A map tileset has been set up, allowing for future additions such as obstacles and other interactive elements. (HIGH)

# Non-functional Requirements (10 points)

1. The game should maintain a frame rate of at least 30 FPS
2. The game should load the entire map within 5 seconds
3. The game should be responsive to user input with little to no delay
4. The camera should follow the player smoothly with no sudden jumps
5. The menus should load and display all elements within 1 second
6. Menu options should be clearly labeled and intuitive for new players
7. Scene transitions between levels should occur within a few seconds
8. Dialogue boxes should appear instantly and quickly display the necessary text

# Use Case Diagram (10 points)



Actor

• The player is the primary actor in this diagram. This represents the user who interacts with the 2D game system, controlling the character, fighting enemies, navigating the map and using the UI.

Player Controls

Description: This use case allows the player to control the character within the game. It includes basic movements such as walking, jumping, and double jumping.

Fight Enemies

Description: This use case enables the player to engage in combat with enemies within the game. It includes collision detection, where the system checks for collisions between the player and enemies or their attacks. This is essential for detecting hits and triggering combat events.

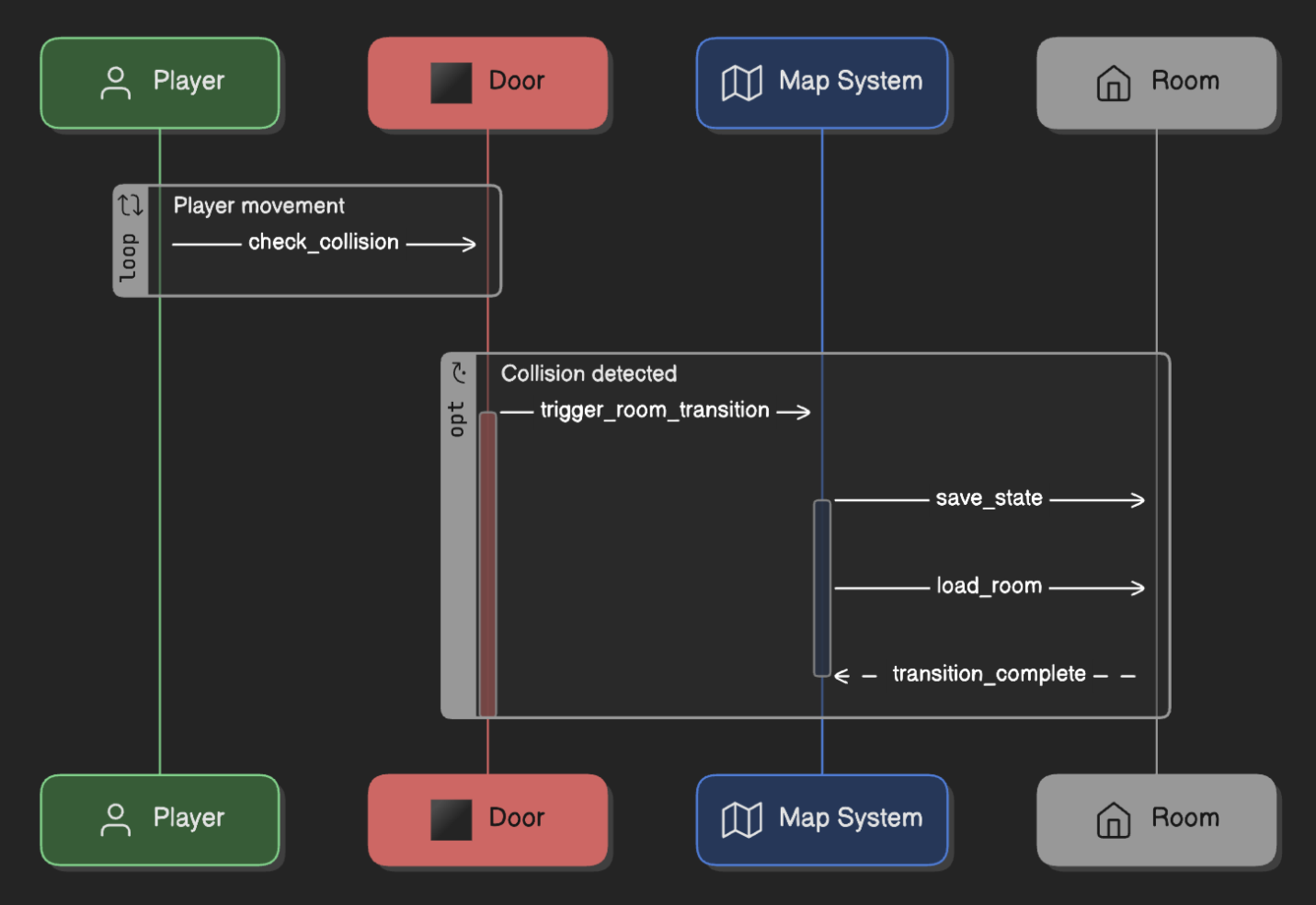
Map Navigation

Description: This use case involves the player navigating through different areas of the game map. It includes collision detection which is used to identify interactions with obstacles or walls during navigation. The player can enter different areas by walking through doors. When the player walks through a door, the game transitions to a new area or scene. The player can interact with items found on the map, such as picking up objects like coins.

Using UI

Description: This use case covers the interaction of the player with the game's user interface (UI). This includes starting the game through the main menu which allows the player to exit the current game session. The player can pause the game during gameplay through the pause menu, which displays options such as continuing the game or returning to the main menu. If the player dies, a menu allowing the player to retry the level will appear.

Class Diagram and/or Sequence Diagrams (15 points)



# Operating Environment (5 points)

* The software will operate only in a Windows 10/11 environment.

# Assumptions and Dependencies (5 points)

* It is assumed that the player is using a keyboard to play the game. We are relying on Godot’s engine and features.