Software Requirements Design and Document

For

Group 11

Version 3.0

Authors:
Maddy Burns
Carly Sweeney
Miranda Arnold

1. Overview

In Lost Wizard’s Labyrinth, you play as a young wizard stranded in a mysterious land after losing your home. To return to your world, you'll need to navigate through enchanting landscapes, interact with quirky characters, and solve intricate puzzles. Each puzzle brings you closer to uncovering the secret magic potion that holds the key to your escape. As you unravel the land's mysteries, you'll gain magical skills and form lasting bonds with its inhabitants. Can you harness the power of magic and find your way home?

Lost Wizard's Labyrinth is an adventure/puzzle game where you are placed in a world and you must solve puzzles to win the game. The player uses clues from wizards to find puzzles and knowledge to solve the puzzles.

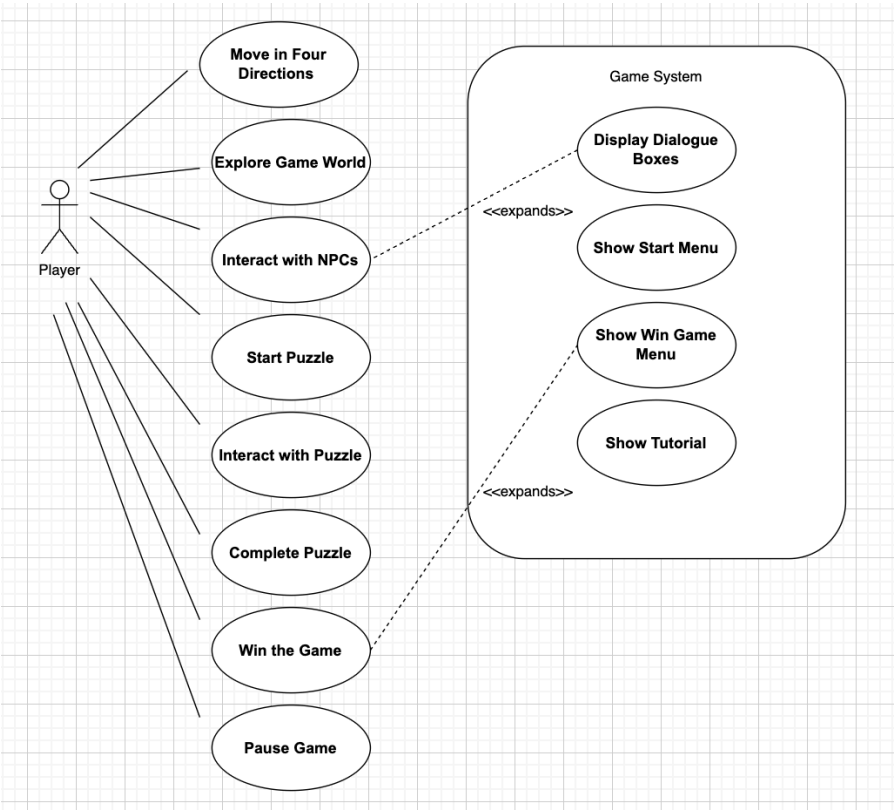
2. Functional Requirements

1. The player must be able to move in four directions (up, down, left, right) using keyboard arrow keys or WASD (HIGH)
2. The player can jump using space bar and sprint using shift (HIGH)
3. The player must be able to explore the game world (HIGH)
4. The player must be able to press a key and interact with NPCs (HIGH)
5. The NPCs must have dialogue boxes (HIGH)
6. The player must be able to start a puzzle (HIGH)
7. The player must be able to interact with the puzzle and complete it (HIGH)
8. The player must be able to win the entire game (HIGH)
9. The user must be given a start menu to begin the game (MEDIUM)
10. The user must be shown a win game menu to exit the game (MEDIUM)
11. The user must be able to pause the game (LOW)
12. The user must see a tutorial before entering the game (LOW)
13. The user must have access to control menu to view controls (LOW)

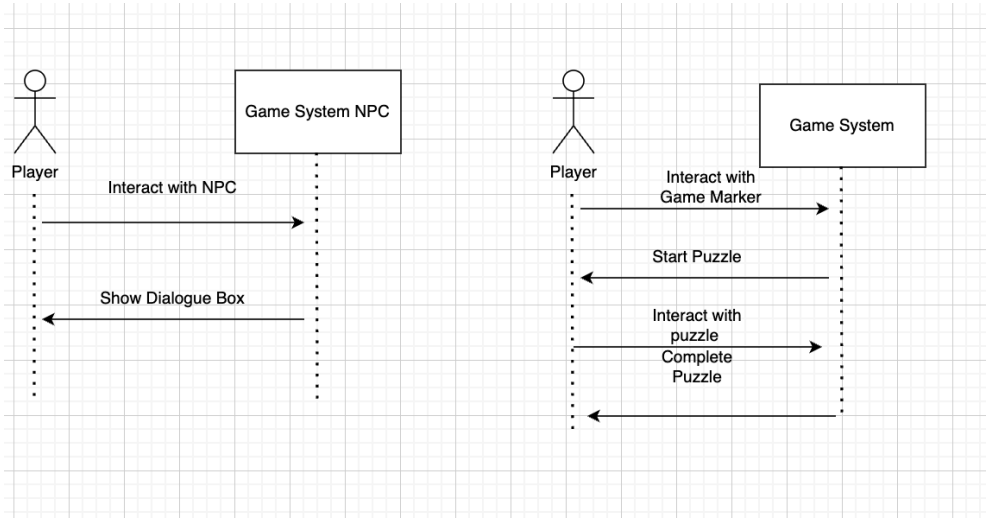
3. Non-functional Requirements

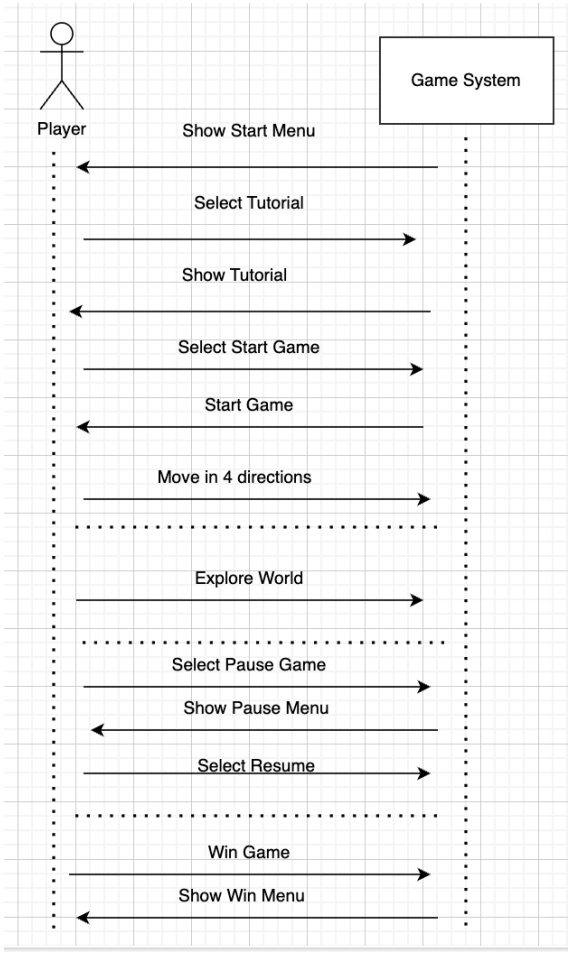
1. The game must have minimal lag for the player
2. The game has a consistent art style throughout
3. The player is not able to access anything that affects the way the game runs
4. The code is easily adaptable for future changes
5. The game is playable on multiple platforms (macOS, Windows, etc.)
6. The game does not crash

4. Use Case Diagram



5. Class Diagram and/or Sequence Diagrams





6. Operating Environment

Hardware: PCs with Intel or AMD processors, at least 4GB of RAM, NVIDIA/AMD/Integrated graphics, and basic input peripherals (keyboard and mouse).

Operating System: Windows 10/11, macOS 10.14+, and popular Linux distributions.

Software Components: Unity 2022 or later, C# scripting, physics, audio, and graphics APIs like DirectX, OpenGL, Vulkan, and Metal.

7. Assumptions and Dependencies

1. Unity removes feature we use in our game
    a. we will have to change our game to work without the feature
2. Players use hardware that is not compatible with our game
    a. players may experience lagging or crashes
3. If assets used in our game are discontinued
    a. we will have to recreate the models and feature of that asset pack
4. Different hardware or driver issues
    a. the keyboard and mouse may be unresponsive

1. Our project relies on Unity's functionality, if Unity makes updates that break our game we will have to change and adapt
2. If Unity becomes non compatible with Blender we will have to find a new way to make models for our game
3. Different versions of Git may cause issues will pushing and pulling our code