Escapism

Project by:
Michael Sclafani, Emelia Hajdarovic, Natalie McGovern, Rachel Asante

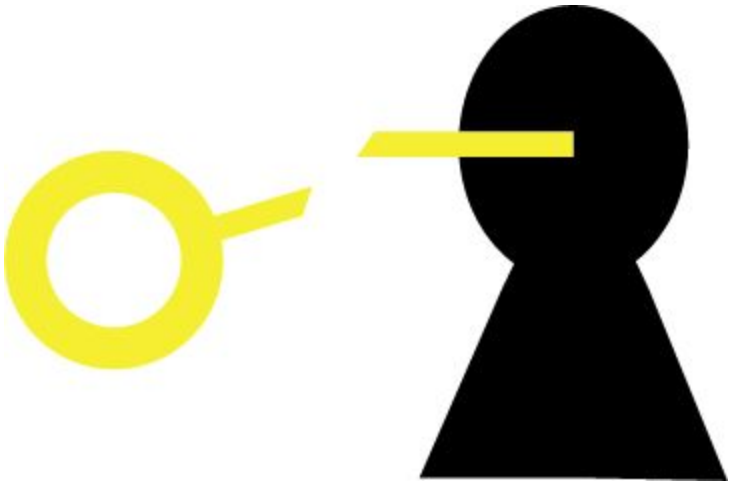

Table of Contents

1. Introduction

This document will go over the fundamentals and overall design our project. We plan to make a game using the Unity game engine named Escapism. The user will be able to download this relatively light game onto their pc. Because of its size, the system the user is playing on does not have to be robust. In addition to this, a version of the game will also be uploaded to the internet so users can play online. This is done in order to reach as many people as possible without the worry of lag and processing power. There is also no need for players to register an account to play making the utilization of the game even easier. In this game, The user will be challenged with logical puzzles and force them to think under the pressure of time constraint. Each puzzle is unique and players will need to solve each one in order to escape the room they spawn in. From this, we hope to make a fun and enjoyable game for everyone while also forcing them to think hard about the tasks at hand.

2. Requirements

2.1 Functional User and System Requirements

2.1.1 *Functional User Requirements*

A user should be able to

a. move freely among the bounds of the room
b. know their level of progress through visual checkpoints
c. see how much time they have left to complete puzzles
d. adjust volume, difficulty of game, etc...

2.1.2 *Functional System Requirements*

A system should be able to

a. allow users to use a keyboard clicking arrangement to move around the bounds of the room
b. give users visual feedback on which puzzles in the room are solved and which ones that are not
c. give the option to change difficulty, volume, etc…

2.2 Non-Functional User and System Requirements

2.2.1 *Non-Functional User Requirements*

A user should be able to

a. play at anytime at any location in the world that allows games from the US
b. participate in game privately without worrying about hackers watching their game

2.2.2 *Non-Functional System Requirements*

A system should be able to

a. accessible 24/7
b. provide quick feedback (confirming correctness) to user's inputs within the puzzles
c. incorporate clear graphics for a good user experience

3. Diagrams

The images below display how the user will be able to interact with our game system. Our goal is to make the interaction between the player and our game as seamless as possible. In **Image 1**, this case diagram is meant to indicate what options the user can take when they are greeted at the main menu of our game, as well as what facilities they have available to them during the game. Features such as starting or stopping a game and adjusting the options will be accessible during the game via the menu in **Image 8**. The other cases include movement both for the character and the camera, and these controls will be instructed through a note during the game as well as an option in the main menu in case the user needs a reminder. By the same token, **Image 2** illustrates the general process each puzzle in the game will go through upon user interaction. Every puzzle will follow a very similar track regardless of any additional details and complications, so factors such as time remaining and incorrect inputs will affect all of them in the same way.

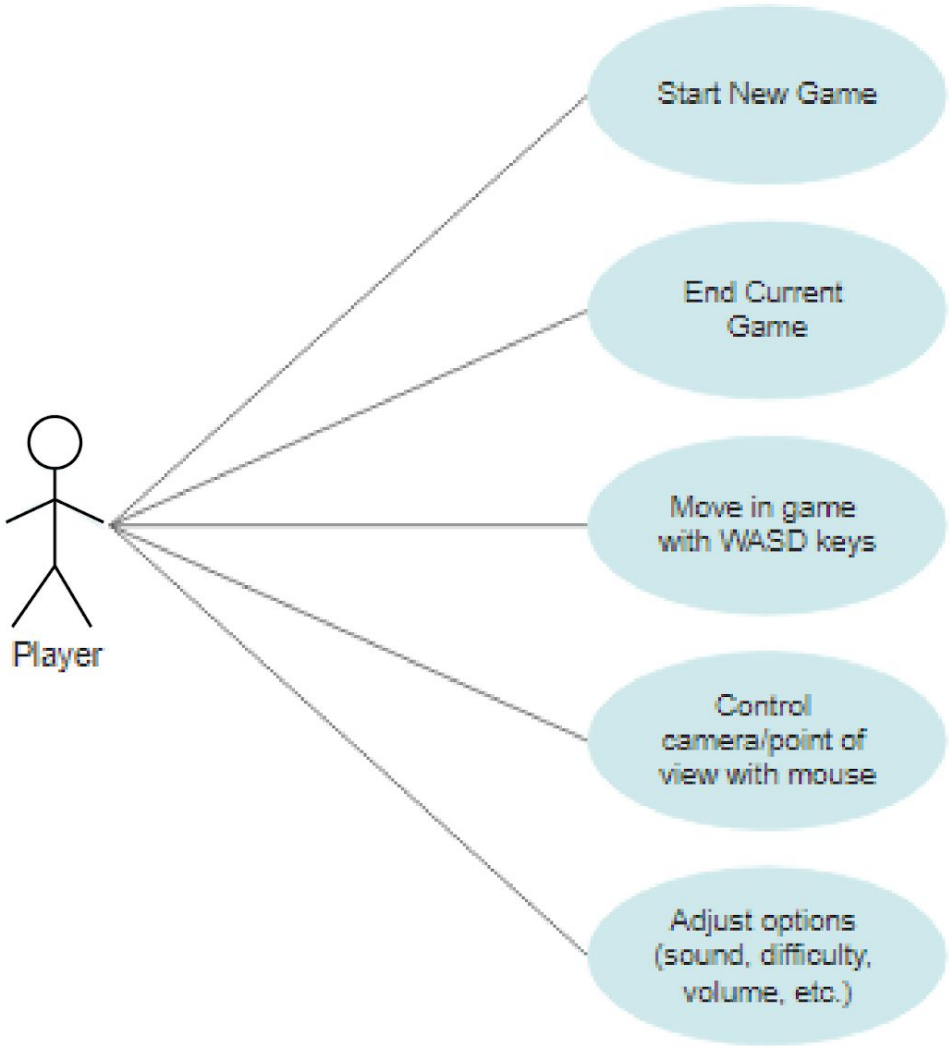

Image 1: User Case Diagram

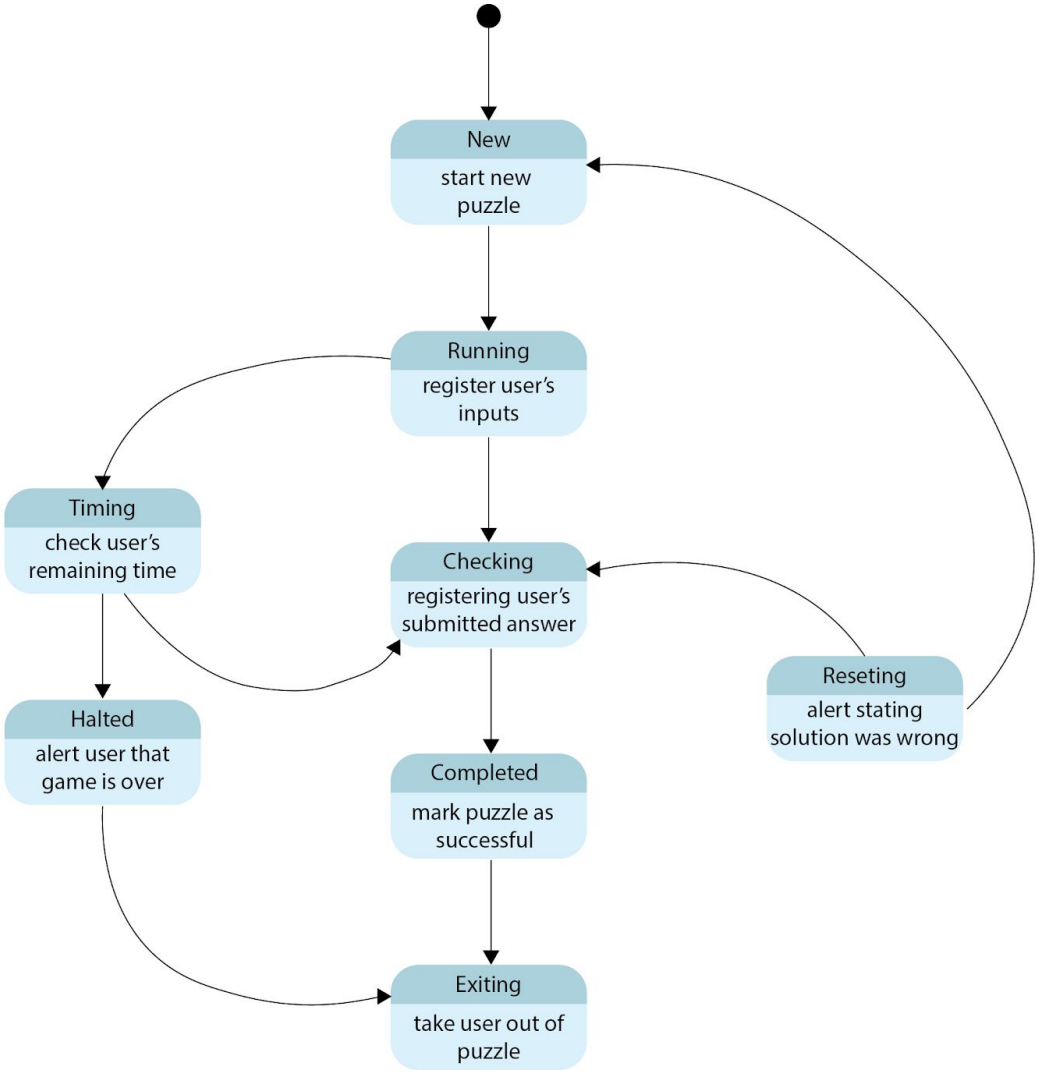

Image 2: Generic New Puzzle State Diagram

4. Interface and Dynamics

As mentioned before, the player will interact with a 3-dimensional room, which is shown in **Image 3**, in which they need to escape by solving multiple puzzles. When a player approaches one of many puzzles in the room, they will be able to focus on it. One such puzzle is shown in **Image 4**, the player is focusing on the sliders puzzle. The sliders puzzle consists of tiles, which are randomized at the start of each game, in which the player must move the "activator" tile to the correct position to complete the puzzle. Instead of the player dynamically moving tiles one at a time as many other sliding puzzle games work, the player will have to input the sequence in order to move the tiles as shown in **Images 5 and 6**. If the player gives the puzzle invalid movement (for example: the player tries to move a tile left but there is no tile there) or if the activator tile is not at the power node by the time the sequence is done, the puzzle resets. All puzzles in the game will follow a similar format in which the player must solve it and messing up will result in a reset and losing time. In addition to this, the user will also be able to change options in game such as volume and difficulty as shown in **Image 7**.

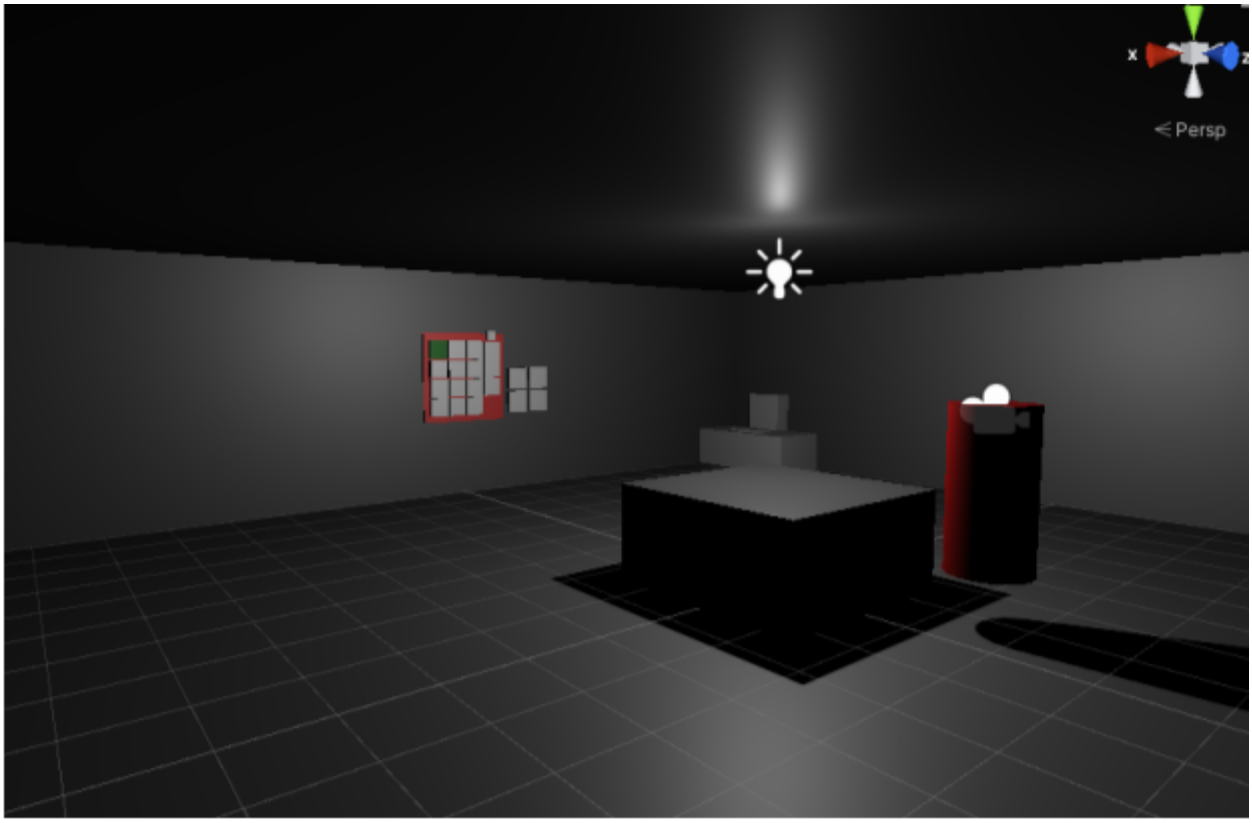

Image 3: Room Layout

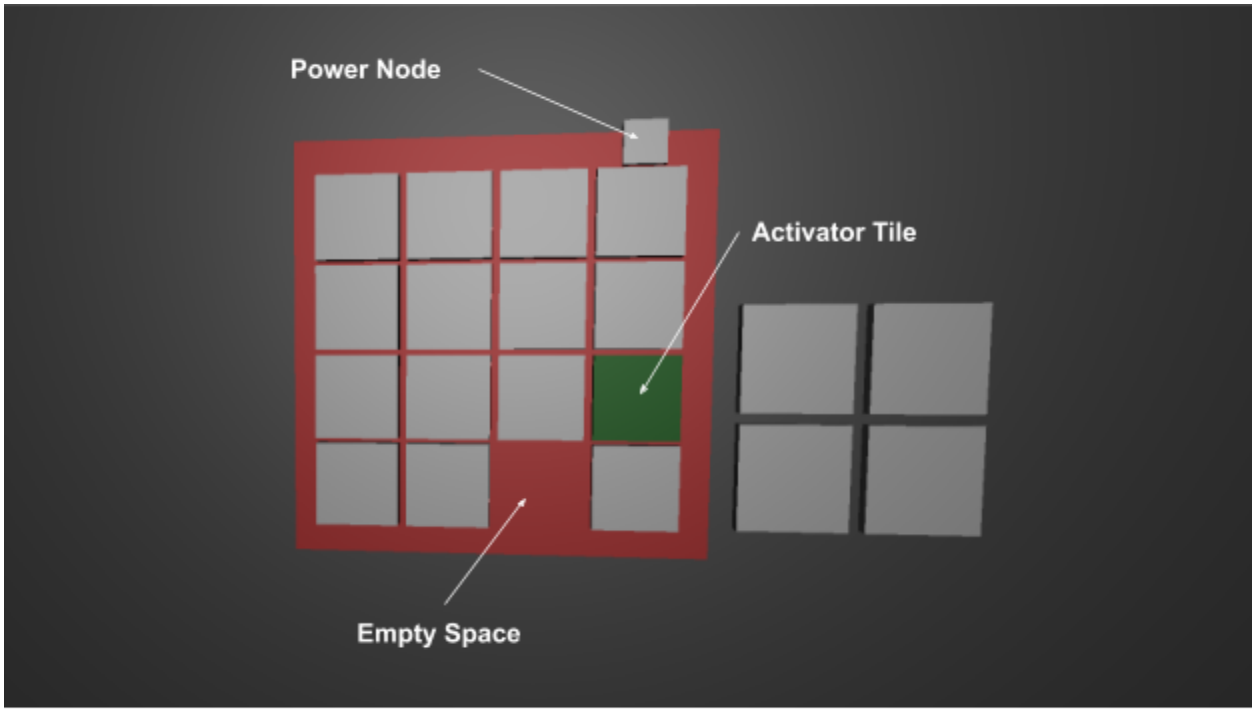

Image 4: Sliders Puzzle Layout

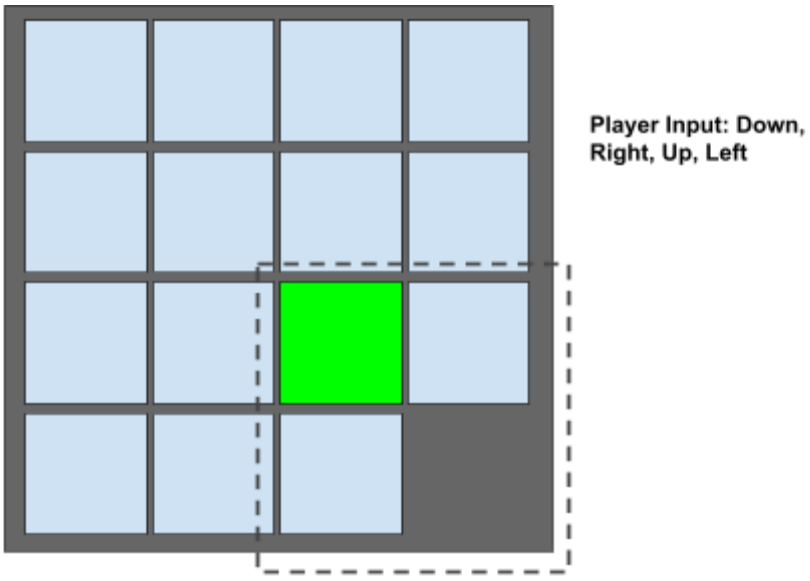

Image 5: Example Starting Puzzle

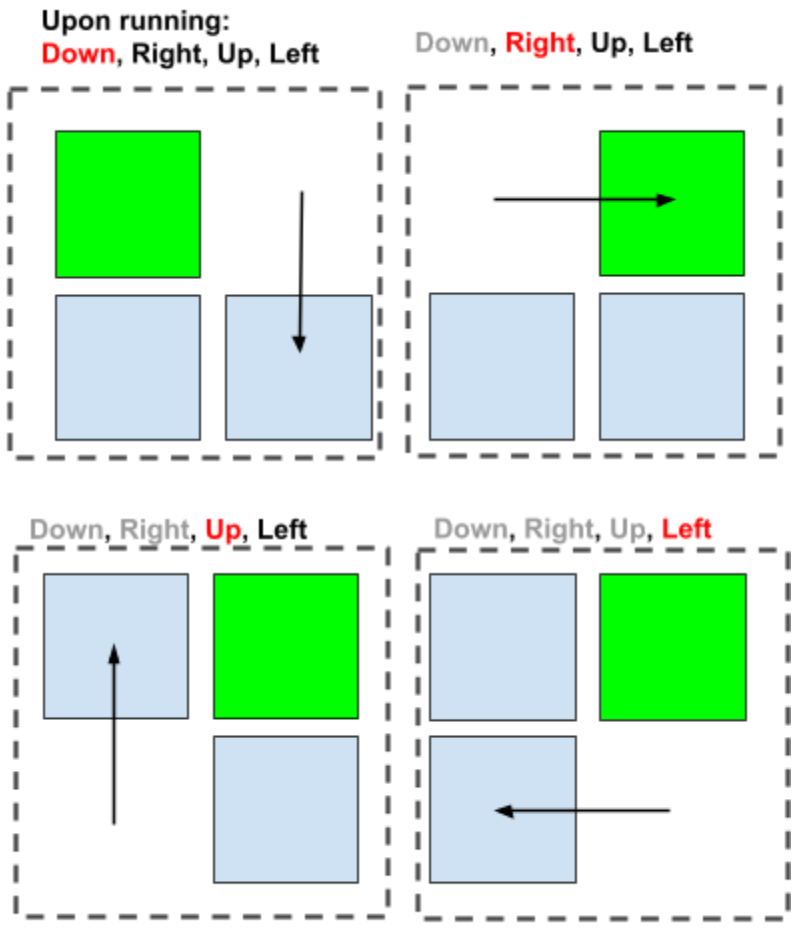

Image 6: Process and Dynamics of User Input Movement

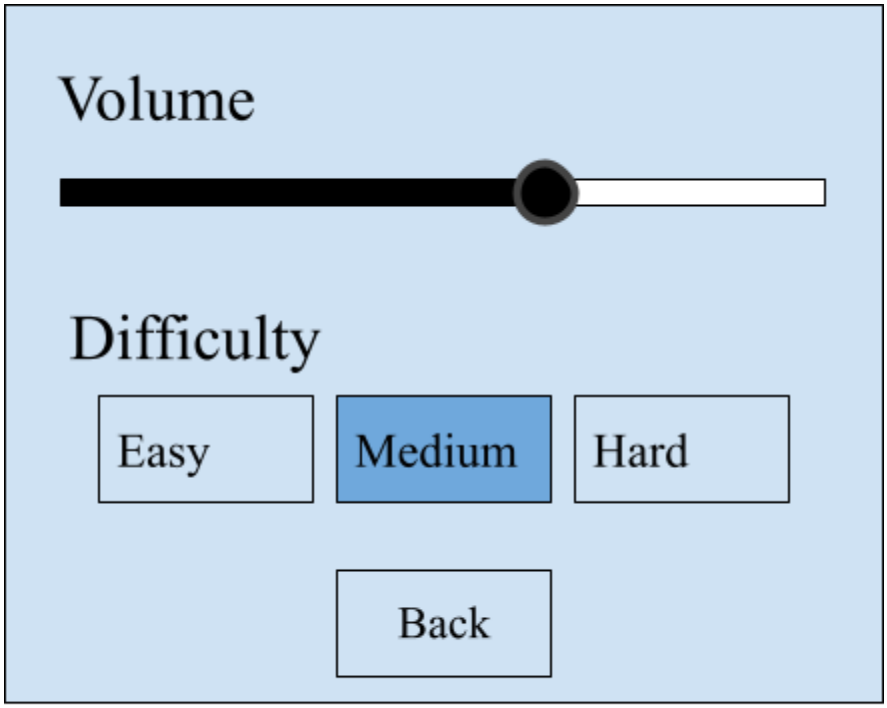

Image 8: Example of Settings UI