By: Chen Ostrovski

Software Requirements Specification

User Stories

The intended audience of the game is mainly young children aged 7-13. The main goal of the game users is to complete all the stages and reach victory, while the desires can be divided into several types of users. Some users will be striving for victory at any cost compared to other users who want to spend time having fun with friends or separately.

User Cases

The users can use the keyboard and mouse in this game. Below you will find a list of the various actions that the user can perform with them:

- Create account/sign in with email and password.
- Choose the difficulty of the game.
- Choose the procedural generation algorithm they want to play in.
- Take on missions by clicking on NPC's.
- Kill enemies with different attacks using the keys 1,2,3,4.
- Move a player with a mouse click on the desired end location.

System Features and Requirements

Functional Requirements

The game should be able to create different types of level with these algorithms:

Room Generation

- **Random Room Placement (RRP) –** Algorithms that starts by randomly generating a room of a random width and length, the randomness of the size of the room is controlled by maximum and minimum allowed lengths.
- **Binary Space Partitioning (BPS Rooms) –** Works by dividing the map into a series of rectangles, and then placing a room in each one on the map. The rectangle can be split either vertically or horizontally, and the minimum size of the split is determined based on the minimum size of the rooms.

Corridor Generation

- **Random Point Connect (RPC) –** Algorithm that works by first iterating through the list of rooms created by the room generation algorithm. For each corridor it draws, it selects a random point at the edge of each of the rooms and connects. If the points are parallel, the algorithm will draw a straight corridor in the appropriate vertical or horizontal direction. If the points are not

parallel, the algorithm will draw a forked corridor made up of three straight corridors in order to connect the points.

- **Drunkard's Walk (DW) –** Similar to RPC it also selects vertex between the rooms and starts drawing a line the first vertex incrementing a randomly chosen amount of vertices in the direction of the second, When the corridor is being made by DW, the algorithm do it in "steps". Each step adds a corridor of a random length that goes in the direction of the end point.
- **Binary Space Partitioning (BPS Corridors) –** Using the tree structure used by BSP Rooms algorithm. It sorts through the sub leaves, connecting each bottom leaf to its partner that shares a parent. When all sub leafs are paired in connects one out of each pair to another pair, until all sub-leafs containing rooms are connected.

User Interfaces

- Front-end software: Unity, C#.
- Back-end software: Aws, Python.
- Database software: Aws, Dynamo DB.

Hardware Interfaces

- Both Mac and Windows operating systems through their default web browser.
- Keyboard.
- Mouse.

Performance Requirements

- The game should load and be usable within 3 seconds.
- The game should update the interface on interaction within 2 seconds

Security Requirements

- Any keys used for the REST API should be stored securely.
- Only the REST API should be able to connect to the database and Lambda.
- Database should be behind a firewall.

Drawings

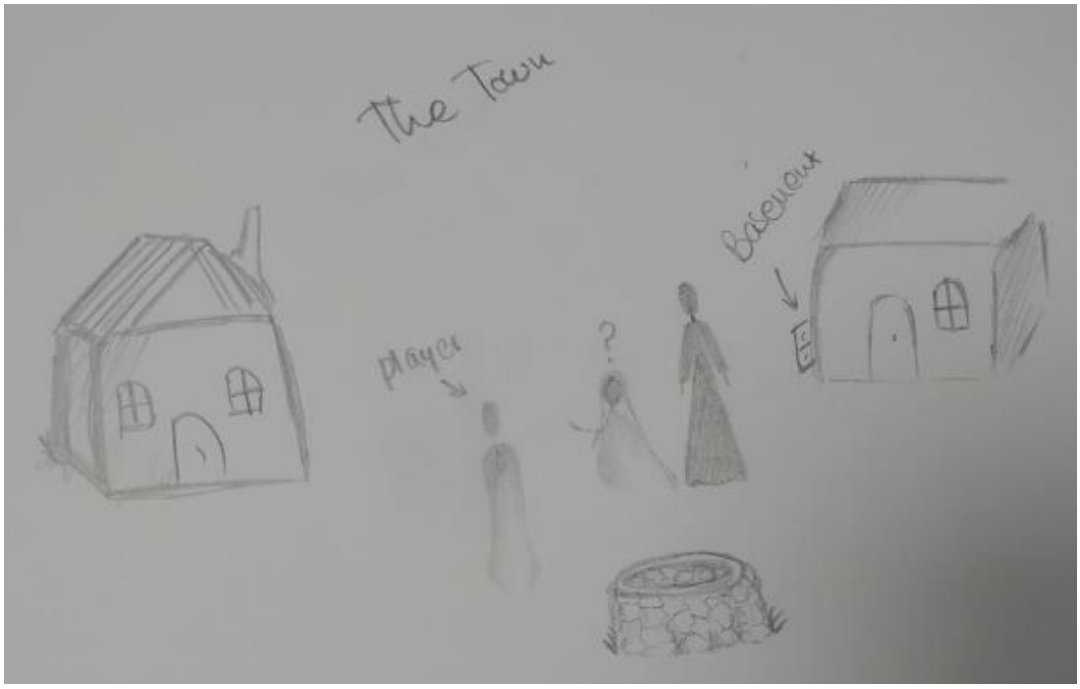

Figure 1: The Player arrives to town and speaks with mayor and the wizard.

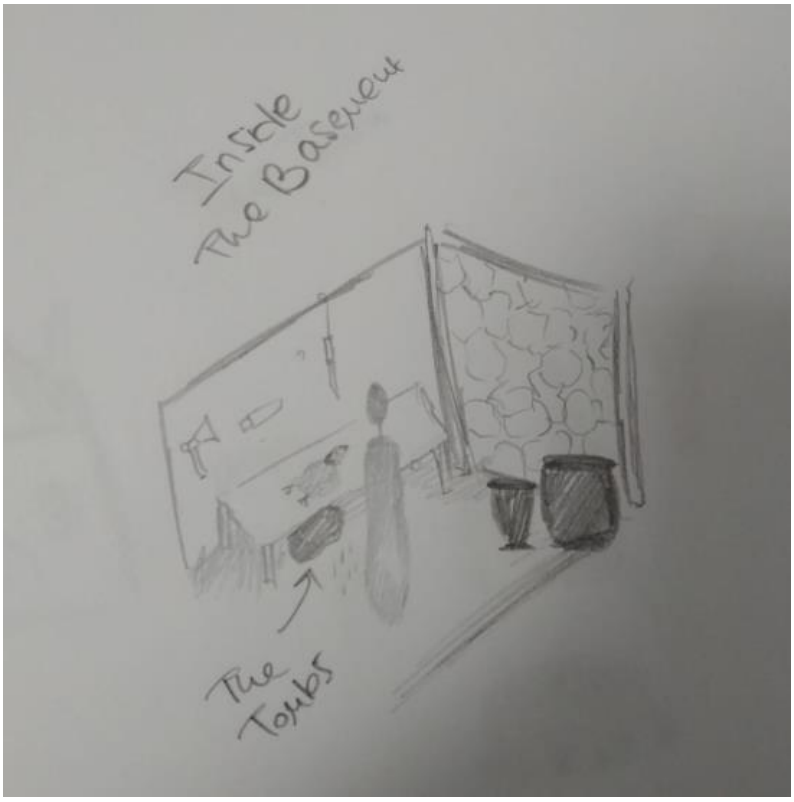

Figure 2: The Player search for clues in Alf's basement.

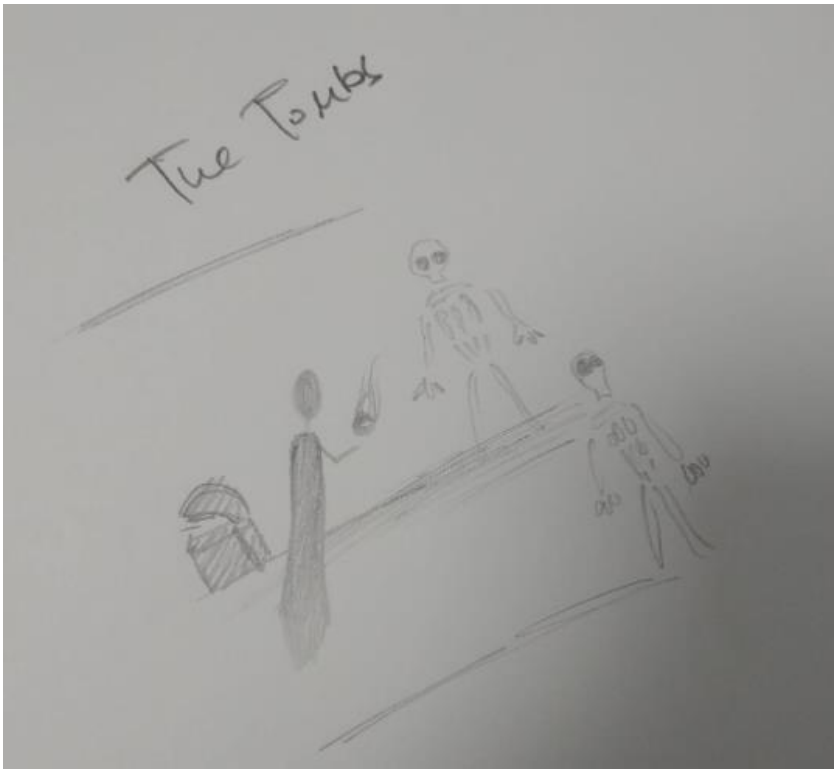

Figure 3: The Player fights enemies in the tombs.