Technical Design Document

Section 1- List of Features Captured from DGG

* 1. List of Features Based on the Game Design Document:
* First Person
* Game world, including
  + 3D Objects
  + Characters
  + Rooms with walls, doors, and windows
* Windows platform deployed
* Opening scene
* Single player
* Multiple endings
* Intractable puzzles
* Sound effects
* NPCs in the background
* In-Level HUD
* Menus
* Lighting effects
* Reflection effects
* Interactive objects
  + Puzzle components
  + Easter eggs
  + Extra objects for realism

Section 2- Choice of Game Engine

The game engine selected is Unity Real-Time Development Platform. Unity is a cross-platform engine the engine itself currently supports building games for more than 25 different platforms, including mobile, desktop, consoles, and virtual reality. Some of the reasons the engine was selected are:

* Unity has a broad reach that makes recruitment easier
* Unity is a good engine for beginners
* Unity is fast and agile
* Unity makes portability easier
* Unity has a large and varied asset store

As well as the author’s prior experience with Unity, its good compatibility with the 3D modeling software of choice, and programming language of choice.   
Ref: https://www.gamesindustry.biz/articles/2020-01-16-what-is-the-best-game-engine-is-unity-the-right-game-engine-for-you

Section 3- Schedule

The schedule for the development of the game is shown below:

//TDB

Section 4- High-level diagrams to illustrate software design

4.1 Layout Diagrams:

The layout diagram illustrates the layouts of the In-Game world areas

The first area is the main space the game will take place in, solving the different puzzles inside to ultimately achieve the end task of escaping that virtual world.

Other sections can be reached through puzzle solving and will lead to one of the endings.

The Main area is shown in the figure below:

//TBD

The Elevator Ending Area shown in the figure below:

//TBD

4.2 Technology Diagram:

\* The technology diagram indicates the technology that is going to be used for different aspects of the game and is displayed in the figure below:

//TBD

4.3 Implementation Diagram:

\* The implementation diagram illustrates the flow of the different stages of the creation of the game

\* It also includes what weeks each stage will take place at

\* This diagram is displayed in Figure 5

4.5 Gameplay Diagram:

\* This gameplay diagram is a simple flowchar of how to play the game, which is shown in fugure 6

//TBD

Section 5- Art Tools

5.1 Jasc PaintShop Pro X7:

5.1.1 Version:

\* 7.0

5.1.2 About PaintShop Pro X7:

\* It includes content-aware features such as "Magic Fill"

\* "Smart Edge"

\* as well as support for [XMP](https://en.wikipedia.org/wiki/Extensible_Metadata_Platform) sidecar files that preserve edit settings for [raw formats](https://en.wikipedia.org/wiki/Raw_image_format).

5.1.3 What it will be used for:

\* Textures e.g. notes

\* General appearance of building and objects

5.2 Blender:

5.2.1 Version:

\* 2.8x

5.2.2 about Blender:

Modeling 3D Objects with lighting and cameras

Animation and simulations for 2D and 3D

Texture and shading objects including scripted and complex costume shaders

5.2.3 What it will be used for:

Modeling the assets and characters used in the game

Most of the animations for characters, objects, and UIs

Texturing and shading the environment

5.3 Audacity:

5.3.1 Version

\* 2.0.x

5.3.2 Features:

recording audio from multiple sources

post-processing of all types of audio

adding effects such as normalization, trimming, and fading in and out

5.3.3 What it will be used for:

Editing, mixing, and recording sound effects.

Section 6- 3D Objects, Terrain, and Scene Management

6.1 Objects:

The objects will be modeled and animated in Blender and exported in the FBX format. The format was established to provided interoperability between digital content creation applications. There are many tools available for conversion from/to FBX file format.

FBX is standard for data exchange between DCC apps and game engines like Unity and UE4. FBX was originally a native format for Motion Builder scenes, so in theory it can hold whole scene data in a node-based structure.

The main materials and textures are to be applied in Unity’s editor, with the exception of textures that need special mapping will be done in Blender and exported in the FBX file.

The scaling, positioning, and rendering of objects and their animations will be done in Unity’s native editor.

6.2 Environment:

The environment will be created and controlled during the development phase in Unity’s editor. It will primarily utilize the models exported from the modeling software. In addition, the lighting, reflections, and manipulation will utilize the tools provided by unity’s development environment.

6.3. Scene Management:

Loading, rendering, and control of the scene and its object will be handled by the Unity’s editor.

Section 7 –Collision Detection, Physics & Interaction

7.1 Collision Detection: