### Technical Design Document Template

1.0 Revision History

<As you revise the document, list what was changed and when it was changed>

|  |  |
| --- | --- |
| Version | Description |
| 1.0 | Initial document |

2.0 Development Environment

2.1 Game Engine

<Proprietary/Unreal/Unity and version>

2.2 IDE

2.3 Source Control procedures

2.4 Third Party Libraries

2.5 Other Software

<2d art assets, audio, 3d modelling etc.>

3.0 Game Overview

> Temp game idea:

You are a ghost and your goal is to scare humans to death with point click movement, when you scare the humans they flee away from you, and hopefully they fall into a pit and die, which gives you one point. Every time a human dies, a ghost buster spawns and will wander around the death zone, and when it sees you or hears a scream it will seek you/the scream position, if it gets too close you lose, you can’t fall down holes but everyone else can, so your goal would be to lead a ghost buster to its death when being chased.

3.1 Technical Goals

<3d graphics, 60fps, Challenging AI etc.>

3.2 Game Objects and Logic

<A list of logical elements in the game, i.e. door, button, pistol, ammo, light, bullet, wall, character etc. and description of their behaviour and purpose>

Seek, Flee, Wander, A\*

3.3 Game Flow

<description of what the player can do (actions) from the start menu to playing the game, through to hitting quit. Include how to win, how to lose, how the player is moved, and what programmer things might need to be considered>

4.0 Mechanics

<A list of the core game mechanics. I.e., what the player can do and how they achieve this, and what this triggers in the game. For example, shooting enemies is a core mechanic in an FPS>

5.0 Graphics

<Describe graphics features here. I.e., is your game top-down 2D? >

6.0 Artificial Intelligence

<Describe how AI works, i.e. state machine, fuzzy logic, GOAP. Describe the various behaviours and how they change behaviour, how do the ‘creatures’ in the game evaluate the world> <include diagrams/flowcharts showing decision making processes>

7.0 Physics

<if needed>

<What engine are you using, what features from it (spring? Colliders?) how will physics be handled for objects? (box or sphere collider for objects, capsule for player) need to record specific locations for any reason? Potential slowdowns and how to mitigate.>

8.0 Items

<List of items you can pick up that can affect the player, and what they will affect, like ‘picking up the hammer (refer collisions above) adds 5 to the players attack attribute’. Include details on how items influence gameplay or AI logic.>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Item | Parameter | Parameter | Parameter | Description |
| Default | 6 | 6 | 6 |  |
| Weapon | 5 | 7 | 7 |  |
| Weapon | 8 | 5 | 5 |  |
| Weapon | 5 | Possible 10 | NA | text |
| Weapon |  |  |  |  |

9.0 Game Flow

9.1 ‘Mission’ / ‘Level’ structure

<Are all levels stored in memory? what data is saved across levels, are levels loaded synchronously to prevent pauses?>

9.2 Objectives

<What does the player try to accomplish on each level/mission? How is the players progress evaluated?>

10.0 Levels

<If any of the Levels require specific behaviours, describe those here>

11.0 Interface

11.1 Menu

<What are the menu options and what do they do?>

11.2 Camera

<Describe the camera, how it moves, perspective/orthographic, can it switch? How? Does it need to render-to-texture? does it prevent itself going through walls, use flowcharts to document behaviour>

11.3 Controls

<Keyboard, tablet touch/swipe/tilt, joystick, mouse etc. record double taps, multi touch, use mouse smoothing/ scale mouse for aiming etc.>

14.0 Asset List

<List all files needed, along with known attributes >

16.0 Technical Risks

<if you want your game to be a 1000 player pvp battle royale with 4k 120fps graphics, you need to say if this is doable and how you intend to do it>