**Technical Design Document || Nature Survives || Team Throwaway Games**

## Section 1 – List of Features Captured from GDD

### Section 1.1 List of Features based on the GDD:

- Organization: Team Throwaway
- Game Title: Nature Survives
- Team Members:
    - Software Engineers
        - Blair Corban
        - Ben Pointer
        - Charmaine Lim
    - Design Artists
        - Ben Llewellyn
        - Hei Huo
        - Shannell Ulunga
        - Elredeano Olofai
- Camera/Game Style:
    - RTS Based Gameplay
- Large interactive Game-World, utilizing;
    - A variety of 3D Objects
        - Trees, Ores, Wongle Workers, Wonders, Settlements, Farms, and a whole lot more
- Built for the Windows 10 operating system
    - We have the capability to pursue other operating systems such as Apple's macOS, but preferably not console-based operating systems such as Sony's PlayStation 4.

- Place able structures to recreate your localized environment and power your civilization
- Audio and sound effects
- Variety of NPC's
    - BroodShroom, which lives in the corruption biome and spawns Fiends at night and when player units are in range.
        - Mushroom Fiend's spawn as a result of the BroodShroom, which means for each night there's a range of Mushroom Fiends attempting to attack your civilization.
- Multiple AI systems
- Resource gathering with incorporated UI.
    - Spend resources to purchase new buildings and/or workers/warriors.
- Comprehensive/informative HUD
- Basic Menu System
- Lighting effects
- Task-able workers
    - Set your Wongle Workers' to complete tasks for you, such as clearing a forest or obtaining ores, all of which will reward the player with some future benefit such as the ability to purchase more workers.

## Section 2 – Choice of Game Engine

The game engine that we chose to use was the Unity Engine, largely due to the fact that we have completed all of our previous prototypes using this engine. Unity offers us a wide range of easy-to-use features such as its animation system, and the NavMesh capabilities, and there is so much more like;

- Audio management, with inbuilt mixers
- Editing of the game environment
    - The simplicity of editing terrain, placing game objects and everything else is fluid in Unity

- Physics support utilizing Unity's inbuilt Rigidbody functions etc.
- Portability
    - Unity offers a wide-range of ports for use to utilize, meaning we could market this game on a large majority of devices provided we can create a character controller best fit for that respective device.

We have been using the Unity engine largely due to the fact that C# does a lot of memory management on its own accord, ultimately meaning we don't have to deal with misallocation of data/memory issues like you might encounter developing in C++ using the Unreal engine say.

**Section 3 – Schedule**

Monday - Friday

8:30am - 5:30pm

**Section 4 – Collision Detection, Physics & Interaction**

We utilized Unity's inbuilt physics system using Rigidbody's, meaning you can apply forces and alter gravity etc., on that specific rigidbody object.

In terms of the collision detection, we utilized Unity's inbuilt RayCast system. Sending out raycast's from a variety of positions dependent on the current action, actions such as;

- Placing an object, such as a farm or building
    - This takes a variety of different positional data, as the raycast has to be manipulated from Click-Point mouse position data (2D) and converted into 3D and then projected in the correct direction at an the correct angle to hit the grid that the player is actually trying to click
    - Attacking Wongles utilize the RayCastHit.sendmessage function, which allows you to create a function on that specifically hit gameobject which you can then manipulate set variables such as health by damage on that say NPC. It's as simple as checking for whether that gameobject that was hit is the tag of the NPC with 'x' function and the rest is simple.

## Section 5 – Game logic & Artificial Intelligence

The Game Logic, and AI utilized within the game is extremely simple.

### Section 5.1 Game Logic:

- The game logic is very simple
    - Utilize resources to improve your civilizations capabilities
        - Defend your home-base from Mushroom Fiends until your wonder has been created
        - Defend your wonder until time's up
            - If you defended, then well done, Nature Survives.

### Section 5.2 Artificial Intelligence:

**Wongle AI:**

- If the unit is selected and the player right clicks a resource it will begin to start gathering it autonomously.
- When the unit has gathered enough resources, it will look for the closest storage building and go and deliver it.

**Military units**

- Will auto attack any enemies in range
- Attack a specific unit if the user right clicks an enemy if selected.

**BroodShoom AI:**

- The BroodShroom will only spawn in the corrupted grid-zone and only provided that its night-time
- The BroodShroom NPC won't spawn Mushroom Fiends unless agitated at least until its night time. If a wongle worker were to go within the region of a BroodShroom during a day it will spawn a Mushroom Fiend.

**Mushroom Fiend AI:**

- The Mushroom Fiend spawns provided there is an active BroodShroom spawned and that it’s night-time, or unless a Wongle Worker goes within range of the BroodShroom.
- The Mushroom has simplistic AI, meaning its entire goal after spawning is purely to destroy the home-base of your civilization. Although it will attack any Wongles or buildings on its way to the home-base.

## Section 6 – Audio & Visual Effects

### Section 6.1 Audio Effects:

- Utilizing free audio effects sourced from the internet.
- Using the recording room on floor 18 to create sounds

### Section 6.2 Visual Effects:

- Utilizing particle systems created by our artist Hei Huo.

## Section 7 – Delivery Platform & Hardware/Software Requirements

### Section 7.1 Delivery Platform:

- PC Windows platform
    - Windows 10

### Section 7.2 Hardware Requirements:

- Dual core processor
- 3 GB RAM
- Keyboard
- Mouse
- Monitor
- Speakers
- Graphics

- SSD/HDD/USB Drive

## Section 7.3 Software Requirements:

- Windows 10
- Direct X version 9, 10 or 11