GAME NAME
PRODUCTION PLANNING
TECHNICAL DESIGN DOCUMENT

Team Name: Endless Geeks

Abstract

This document is made to record technical information for the Project

Contents

About

A modern 3D remake of the classic game Pong with new mechanics to entertain a modern audience

Describe the purpose of this document (1 paragraph)

Change Log

Updates made to the document should be described below.

Version	Author	Date of change	Description
0.0.0	AIE	22/09/2020	Initial Template created
0.0.1	Craig	15/10/2021	Started Filling Out Tech Requirements
0.0.2	Jasper	17/10/2021	Fixed Grammer and Spelling
0.0.3	Jasper	18/10/2021	Rewrote Stuff to make it clear
0.0.4	Craig	19/10/2021	Finalized Doc finishing the Coding Standards
0.0.5	Jasper/Craig	20/10/2021	Finished the risks Being the end of the doc

Team Members

Name	Role
Craig Lovell	Programmer
Daniel Leonards	Artist
Rachel Espinosa	Artist
Sasha Smirnova	Artist
Jasper Eyers	Designer
Joel Cefai	Designer

Development Environment

This section outlines the required software and systems required for development of this project.

Software Requirements

The below table outlines the software requirements for development of this project. Developers contributing to the project are required to use the approved software outlined below.

Any software that contributes to the direct development including planning and communication tools should be outlined below. A developer contributing to the project should have the below software available to them for use.

This includes tools like Microsoft Teams, Trello, HackNPlan, Git, Image / Audio editing tools, modeling tools etc. A discussion should be had with the team to ensure all areas are identified appropriately.

Software	Version	License	Used By	Used For
Unity 3D	2020.3.15f2	Education	Programmers, Designers, Artists (On Campus / Off)	Development of Game
Microsoft Teams	1.4.00.26376	Student	Programmers, Designers, Artists (On Campus / Off)	
Photoshop	2021	Student	Artists, Designers	Texturing
Git Kraken		Student	Programmers, Designers, Artists (On Campus / Off)	
Hack N Plan	-	Student	Programmers, Designers, Artists (On Campus / Off)	Organizing jobs and time management from day to day
Maya	2020	Education	Artists (On Campus / Off)	3D Modelling
Substance Painter	2020.7.1.1	Student	Artists (On Campus / Off)	Texturing

Accounts

The below table outlines any accounts that may be needed for the development of the project. An account is usually identified by 2 areas:

- **Individual**: Each developer of the project may need an individual user account, for various software or services. This includes software like Trello, HackNPlan or Git.

- **Organization**: A project or organization account is often developed for the software to integrate with other services, this includes things like Advertising / AdSense / git organizations / repos, Facebook developer account etc. An organization account is usually managed by 1 or more team members. Ownership of the account should be able to change between members.

Account/Service	License	Used By	Used For	Owner
Github	Free	Programmer, Art, Design	Contributing to projects hosted on github	NA
HackNPlan	Free	Programmer, Art, Design		NA
Photoshop				NA
Autodesk		Art, Design	Modelling, Rigging and animating	NA
Unity	Free	Programmer, Art, Design		NA

Third Party Libraries

Unity/Unreal comes with a default collection of plugins, tools and assets. Its plausible, and often encouraged to pull in additional assets, tools, plugins or scripts etc. developed by a 3rd party. Any additional library or assets developed by the third party should be listed below.

Asset/Library/Package name	License	Used For

Version Control

Repository

https://github.com/AIESydneyStudents/ENDLESS_GEEKS

https://github.com/AIESydneyStudents/ENDLESS_GEEKS.git

Contributors

- Craigjlovell
- AmazingSmasha
- JoelCefai
- Dan-Leonards
- Jsniper19
- rachelespinosa

Commit Message Format:

Within GIT, a commit message can contain any arbitrary text, no format is enforced. However, it is useful for commit messages on a project to follow a consistent format that includes pointed details about the changes that have been made in the commit.

A Good commit message will include the following information

- **Type**: Represents the type of change, often the "Type" can be inferred based on the associated ticket in your project management tool, which may include:
  - **Feature:** This commit implements a new feature, makes progress, or improves a feature.
  - **Fix:** A bug has been identified, this commit relates to changes that resolve the issue
  - **Refactor:** The code, folder structure, or other parts of the project needs some adjustments to better support maintainability and addition of new features.
  - **Performance**: This commit has changes that improve the projects performance
  - **Docs:** This commit has made changes to documentation

- **Scope:** Refers to the area of the project being changed, could refer to things like (menu) (inventory) (save system) (level) (controls) etc… Scope's may change throughout development, but can broadly identified. Outline the scopes below that seem suitable for your project

- **TaskId:** Id of the associated ticket representing the change
  Your commit will be automatically linked to a github issue when the message contains the GitHub issue ID.

- **Summary:** A short description of what has been changed.

Commit Message Format: `Type (scope): #TaskId : Summary`
Examples:

Feature (menu) : #1302 : Added Exit button to main menu
Fix (menu) : #1395 : Updated button prefab with so that hover works on web builds
Feature (sandbox) : #1129 : Added rock asset to test scene, Created Rock prefab

Docs (readme) : #1111 : Updated project summary and title
Performance (level) : #4913 : Improved level generation code
Performance (player) : #4912 : reduced texture resolution and model vertex count for the player to ensure performance on web builds

Target Platform

This project will be deployed to the following platforms:

- Windows / PC
- WebGL / Browser
- Android Mobile

Windows/PC

Windows/PC Limitations

Outline Windows/PC limitations, provide short description of the limitation. could include:

- Keyboard/mouse (standard PC hardware)

Minimum Windows/PC Specs

Outline the expected minimum system requirements required to run the project in release build.
The minimum/maximum specs should consider target audience system specs and drive both technical and non-technical design decisions to ensure project runs on specified devices.

- Processor: intel® Core™ i7 @ 1.80GHz
- Ram: 8GB
- System Type: 64-bit OS

Release Build Instructions

Detail the steps needed to build for the desired platform – Are there any manual steps required to prepare your project for release?

WebGL/Browser

Duplicate this section for each desired platform

WebGL/Browser Limitations

Outline WebGL/Browser limitations, provide short description of the limitation. could include:

- Keyboard/mouse (standard hardware)
- Website Framerate
- Website stability

Minimum WebGL/Browser Specs

Outline the expected minimum system requirements required to run the project in release build.

The minimum/maximum specs should consider target audience system specs and drive both technical and non-technical design decisions to ensure project runs on specified devices.

- N/A

Release Build Instructions

Detail the steps needed to build for the desired platform – Are there any manual steps required to prepare your project for release?

Android Mobile

Duplicate this section for each desired platform

Android Mobile Limitations

Outline Android Mobile limitations, provide short description of the limitation. could include:

- Joystick optimization
- Performance constraints (max number of particles, game objects etc)

Minimum Android Mobile Specs

Outline the expected minimum system requirements required to run the project in release build.
The minimum/maximum specs should consider target audience system specs and drive both technical and non-technical design decisions to ensure project runs on specified devices.

- android 7

Release Build Instructions

Detail the steps needed to build for the desired platform – Are there any manual steps required to prepare your project for release?

Deliverables

A Build of the project should be generated every Day and placed in the following location:

```
path to network drive folder / <BuildID> / <platform> / *
```

Build ID:
When uploading the build, a sequential naming strategy should be used.
Outline the naming convention that should be followed when updating the daily build.
Some examples include:

- Semantic versioning: Min Max Mid
  - For bulk builds v1.0, v2.0 and so on
    https://www.geeksforgeeks.org/introduction-semantic-versioning/
- Date Time versioning: DD_MM_YYYY

Custom Game Systems

This game will feature a unique dynamic camera system linked to gameplay

There will be set locations for a camera to pan to throughout a game; when a point is scored the camera will move (via animation) to the corresponding location.

The duel victory system will be synchronized to the dynamic camera (so the game will end at the final camera position).

Gameplay will also be swapped from 2D gameplay to 3D gameplay at a certain angle of rotation.

Coding Standards

Outline the coding conventions followed during the development of the project.
You may link to existing coding standard documentation

- Functions will have braces beginning on the next line and braces will end on their own line.
- Variables will not have any prefixes or suffixes.
- Variable names must be relevant to the function of the variable.
- Global variables must be placed at the top of the script and local variables must be placed at the top of the function.
- Links to other scripts must be placed at the top of the script.

Coding Standards Enforcement

How should coding standards be enforced, code reviews? Linting tools? Any special setup or processes to follow to aid in this process?

We will use code reviews to maintain consistency in all scripts.

Technical Goals and Challenges

- Bullet point any Technical Goals that can be identified for the development of this project. A goal statement should identify how it is measured for success.

Technical Goals and Challenges:

- Camera moving with scores
- The moving 3D environment
- Complete random ball bounces off walls / realistic bounces

Technical Risks:

Bullet point technical risks (skill gaps, or potential over scoped features, areas of project needing additional research)

- Smooth movement with the cameras
- Moving Environment
- Shaders

Risk Avoidance

For each goal / risk outline potential approaches that can be taken to minimize the risk. Area there features that could be cut / redesigned?

These risks are necessary for the project and must be researched well and given time to polish