‘The Seas Will Bury Us’ TDD

Lathraia Games

Justin Green, Ethan Dawkins

# Contents

[1 Contents 1](#_Toc177122577)

[2 Team Members & Roles 2](#_Toc177122578)

[2.1 Design 2](#_Toc177122579)

[2.2 Programing 2](#_Toc177122580)

[2.3 Art 2](#_Toc177122581)

[3 Revision History 2](#_Toc177122582)

[4 Development Environment 2](#_Toc177122583)

[4.1 Game Engine 2](#_Toc177122584)

[4.2 IDE 2](#_Toc177122585)

[4.3 Plugins 2](#_Toc177122586)

[4.4 Other Software 2](#_Toc177122587)

[4.5 Source Control 3](#_Toc177122588)

[4.5.1 Source Control Style Guide 3](#_Toc177122589)

[4.5.2 Source Control Procedures 3](#_Toc177122590)

# Team Members & Roles

|  |  |  |
| --- | --- | --- |
| Design | Programing | Art |
| Yash Verma | Justin Green  Ethan Dawkins | Valentina Sequeira  Micki Huang |

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Editor** | **Additions, Changes and Removals** |
| 0.0 | 13/09/2024 | Justin | Initial Document Setup – 1.0, 2.0, 3.0, 4.0 |

# Development Environment

## Game Engine

This Game will be created in the Unreal Engine (**Unreal 5.4)**. This engine is an industry standard 3D game development software and was chose because of the team’s prior experience with the engine and its free upfront cost. It is important to note that **royalties will apply after $1 million USD (*$148,590.00 AUD*) in annual gross product revenue** as stated on the [Unreal’s download page](https://www.unrealengine.com/en-US/download).

## IDE

The IDE chosen for this product is Visual Studio Community 2022 (**VSC22 17.11.3)**. This is because it is Unreal’s default and preferred IDE and provides Benefits such as quicker symbol loading.

## Plugins

The only plugin used in this project at this version is the **Visual Studio Integration Tools.** This enables better integration with the IDE.

## Other Software

|  |  |
| --- | --- |
| **Software** | **Purpose** |
| Word | Documentation |
|  |  |

## Source Control

The version control system used for this project is Git and the version control client is GitHub

### Source Control Style Guide

To promote clarity and collaboration, a source control style guide will be put in place. I will be using a modified version of an existing [GitHub Style guide by git user Shinjith](https://gist.github.com/shinjith-dev/1c95763fe3aa0df8e2af233ae4f36686#commit-message-conventions). Commit messages should go as follows:

type(s)**(**scope**):** short description

optional body

#### **Type Scope and Message Body**

A type prefix in the subject line can be used to represent type of the changes included in the commit. Some of the commonly used types are:

* Add: To summarize a new addition in the project.
* Mod: To summerize a modification made to a previous addition.
* Fix: To address a fix to the project.

**Mod/Fix(**TutorialLevel**):** Changed level layout and fixed clipping issues

types can also be combined using the / symbol to provide further context. For example:

A scope is added to a commit's type to offer additional contextual information, and it is enclosed in parenthesis, for example:

Add(**BP\_Player**): Created Player BP.

Mod(**BP\_Player & Locomotion**): Using blueprints player now has locomotion.

A body can be added to the message to include detailed explanations in the commit.

### Source Control Procedures

The Git repository will be hosted through GitHub and will consist of 2 major branches being an origin/main branch, and a development branch, and any feature branches. Any big features will be developed separately within their own feature branch which is a snapshot of the development branch and will then be safely merged into development branch when completed, the branch will also be deleted once a feature has been finished and merged into development branch.

In the diagram below the development branch is a sub-branch of the origin / main meaning it is a modified version of the origin. Other feature branches will then be a snapshot of the development branch.

Having our Git architecture setup like this means that development will always be playable in the editor, and the origin / main will always be a playable build of the game. This will also reduce work being overwritten as ideally there should be minimal to no overlap of actively worked on assets.

Also as a sidenote, working builds will be made every 1-2 weeks to allow for a mostly up-to-date build to show off at all times.

#### Source Control Procedure Diagram