|  |
| --- |
| The University of the West of Scotland |
| Games Project Report |
| 1819\_COMP09097\_01 |

|  |
| --- |
| B00330023 Alasdair Hendry  23/01/2019 |

Contents

[1 Technical Design Document 1](#_Toc536218284)

[1.1 Responsibilities 1](#_Toc536218285)

[1.2 Overview 1](#_Toc536218286)

[1.3 Design & Implementation 1](#_Toc536218287)

[2 Project Plan 1](#_Toc536218288)

[2.1 Gantt Chart 1](#_Toc536218289)

[2.2 Risk Analysis 2](#_Toc536218290)

[2.3 Modifications 2](#_Toc536218291)

[2.4 Approach 2](#_Toc536218292)

[2.5 Version Control 2](#_Toc536218293)

[3 Implementation 2](#_Toc536218294)

[3.1 Development Methodologies 2](#_Toc536218295)

[3.2 Production 3](#_Toc536218296)

[3.2.1 Assets 3](#_Toc536218297)

[3.2.2 Source Code 3](#_Toc536218298)

[3.3 Coding Practices 3](#_Toc536218299)

[3.4 Level Design 3](#_Toc536218300)

[3.4.1 Terrain 3](#_Toc536218301)

[3.4.2 Space 3](#_Toc536218302)

[3.4.3 Texturing 3](#_Toc536218303)

[3.4.4 Light & Shadow 3](#_Toc536218304)

[3.5 Bugs Encountered 3](#_Toc536218305)

[3.6 Code Examples 3](#_Toc536218306)

[3.7 Art Production 3](#_Toc536218307)

[3.8 Audio Production 3](#_Toc536218308)

[3.9 User Interface 4](#_Toc536218309)

[3.10 Immersion & Engagement 4](#_Toc536218310)

[3.11 Mechanics 4](#_Toc536218311)

[3.12 Gameplay 4](#_Toc536218312)

[3.13 AI 4](#_Toc536218313)

[3.14 Flow & Balance 4](#_Toc536218314)

[4 Quality Assurance Plan 4](#_Toc536218315)

[4.1 Project Plan Iterations 4](#_Toc536218316)

[4.2 Risk Assessment 4](#_Toc536218317)

[4.3 Quality Assurance Audit 4](#_Toc536218318)

[5 Playtesting 4](#_Toc536218319)

[5.1 Testing Methodology 4](#_Toc536218320)

[5.2 Compatibility Testing 4](#_Toc536218321)

[5.3 Regression Testing 4](#_Toc536218322)

[5.4 Acceptance Testing 4](#_Toc536218323)

[5.5 Alpha Testing 4](#_Toc536218324)

[5.6 Beta Testing 4](#_Toc536218325)

[5.7 Functional Testing 4](#_Toc536218326)

[5.8 White & Black Box Testing 5](#_Toc536218327)

[6 User Experience Analysis 5](#_Toc536218328)

[6.1 Questionnaire Analysis 5](#_Toc536218329)

[6.2 Feedback Analysis 5](#_Toc536218330)

[6.3 Tracked Player Issues 5](#_Toc536218331)

[7 Critical Appraisal 5](#_Toc536218332)

# Technical Design Document

## Responsibilities

As with the first trimester, I will be continuing to work on this project as a solo endeavour. All responsibilities land on me and I will undertake all roles for the remainder of this project.

On a weekly basis, I will record the progress of the previous week and document the aims and objectives for the following week. This will allow me to keep track of the current state of the project and will serve as an early warning sign in the event that I need to request assistance from a peer or lecturer.

## Overview

In trimester 2, the current stage of this project, the focus shifts from planning and designing, to implementation, testing and user feedback. The implementation process must follow the plan outlined in trimester 1, unless the potential changes are documented and justified.

The solution will be implemented following the development methodology outlined in the design document. The process of implementation will consume the largest portion of time and the desired outcome is to have the game in a finished state with two to three weeks remaining before the deadline. This ensures enough time is left to carry out the post-development tasks such as testing and collecting feedback.

## Design & Implementation

From a functionality and mechanic standpoint, the idea is remaining relatively the same as the original one proposed in trimester 1. As the prototype was created in the previous stage of this project, it can be used as a foundation in which further functionality can be adapted and implemented.

Of course, there are multiple areas of the final game which will be different from the initial prototype. This is natural, as more time is now available to be spent implementing the game.

Currently, the most notable change is the artwork. Previously, artwork consisted of primitive shapes and non-uniform assets sourced from the internet. However, whilst waiting on the commencement of trimester 2, I was able to source high quality art assets that fit the style of the game perfectly. Three separate asset packs were purchased, all of which were created by the same studio. After implementing these graphical updates, the game’s aesthetic improved drastically. This graphical update will also provide an enhanced feeling of immersion to the player, as the world seems to come alive when all of the art assets are uniform in theme.

Aside from the major graphical update, there are no other notable changes to the proposed design that can be documented.

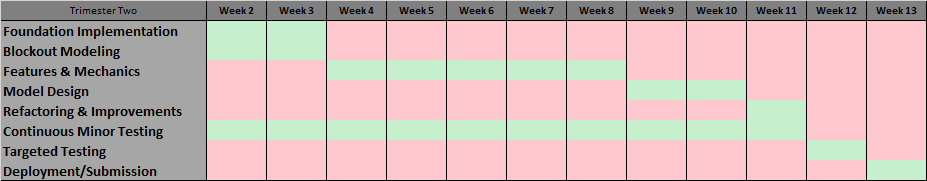
DETAIL PLANNED IMPLEMENTATIONS

# Project Plan

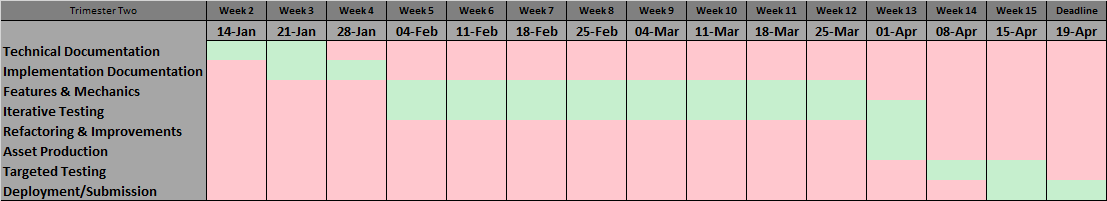
## Gantt Chart

A Gantt Chart for trimester 2 was created in the initial planning stage of the project. However, this chart has been updated in a large way to reflect the changes in the plan.

The initial chart was created under presumptions and estimations of timescales and expectations. However, as new information has become available regarding submission deadlines and coursework requirements, a new version of the Gantt Chart has been created. As can be seen below the new chart has changed in a variety of ways.



Previous Gantt Chart created in trimester 1



Updated Gantt Chart reflecting changes to project (Full size version can be found in Documentation/Report/Data)

Firstly, foundation implementations have been removed as a large portion of this work was completed as part of the prototype in trimester 1.

Secondly, sections dedicated to implementing basic models and creating custom models have been removed. This is due to the fact that, with the purchase of the previously mentioned 3D asset packages, there is now no need to create custom models. As a replacement to this, a section for generic asset production has been created. This is precautionary, in case any last-minute pieces of art or audio need to be created.

On top of this, as the time period available to work on this project was wildly underestimated in trimester 1, extra date slots have been added and assigned to the Features & Mechanics sections. This should allow for a greater overall quality of the final product.

Lastly, as the coursework has specified further documentation is required, these sections have been added to the start of the project.

## Risk Analysis

Analysing risks to the project remain much the same as the last stage of this project, with some slight modifications. Sickness and attendance issues remain untouched as this is something that is not only out of the control of the project members, but also poses the same amount of risks as the previous stage.

The major risks changes are related to personal capabilities and data corruption. These two risks become even more important in this final stage of the project, as the implementation of the game starts to begin. One instance of data corruption in this stage could potentially set the project back multiple days or weeks, however, version control services will be utilised to prevent this as much as possible.

Further to this, understanding personal capabilities, limitations and sticking to the project plan becomes crucial in this stage. Deviating from this plan, or under-estimating how much work needs to be completed, can have a serious effect on the outcome of the project. This result of this could be missing a submission deadline, or key features being scrapped from the implementation as cosmetic features are explored. Most of these risks can be avoided with thorough planning and competent implementation.

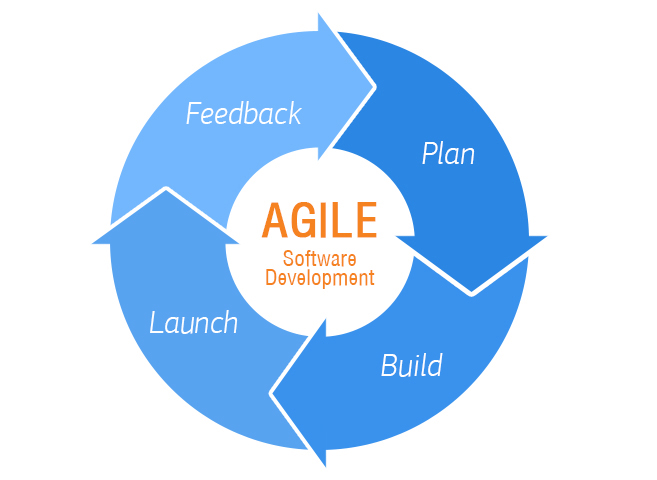
## Modifications

Although the timeline of the project remains mostly unchanged, some changes have been made to adapt to the new circumstances surrounding this stage of the project that were unknown during the first stage. There are no changes to the planned outcome of the implementation of the game. Changes to the project plan for this trimester have been detailed in sections 2.1 & 2.2.

## Development Methodologies

Many development methodologies are available to implement for a game development project. These methodologies are simply guidelines which you are advised to follow, as they can greatly increase development efficiency, allow you to detect bugs earlier and quickly iterate through development phases using a structured approach.

With no doubt, the Agile methodology is the most popular for games development projects, and as such, will be the approach taken throughout this project. Agile is an iterative approach which encourages layered implementation and a test-as-you-go strategy. This is an extremely useful approach for short-life projects as at the end of any given iteration, the product could potentially be built and released, providing freedom to remove features if time constraints become an issue.



https://www.nascenia.com/necessary-factors-to-make-agile-software-development-a-success/

Although this project spans 24 weeks, double the average university trimester length, it is still a relatively small timescale to plan, develop, test and deploy a fully featured 3D game. This is the main reason the Agile approach has been chosen for this project.

In contrast to the Agile approach, there are others available, such as Waterfall and V-Model. These approaches are preferable for projects which have access to a greater time frame, for example in a professional game studio.

Waterfall suggests a lineal approach to development, such that planning is complete before design, and design is complete before implementation. This is a solid foundation to follow during lengthy projects, as it almost guarantees a flawless development cycle. The downside to this is that it takes an enormous amount of time and effort to fully carry out each stage before continuing to the next and restricts the ability to add new features after design stages are complete.

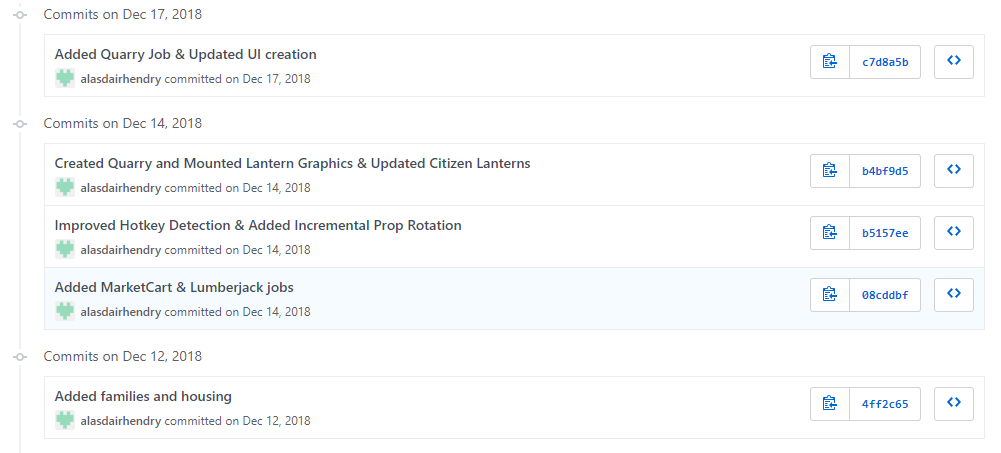
V-Model suggests a similar approach to Waterfall, as it uses the same base foundation, but adds iterative testing stages. This means each stage is tested against every previous stage before continuing. This requires even more time than traditional Waterfall, and as such, is unsuitable for such a small project.

## Version Control

Version Control is a service, provided by websites such as GitHub and GitKraken, which provide a cloud-based platform in which data can be stored. These services are targeted specifically towards project-based data such as software, games and website projects.

GitHub will be used throughout the duration of this project and has been used since the initial stage. This service provides an easy-to-use desktop application that streamlines storing and retrieving data. A repository has been created on GitHub for this project, which has been linked to a folder on project hardware. This means that any local changes to source code will be detected, and GitHub will prompt a backup when it is next opened. The repository, along with some data commits can be viewed below.





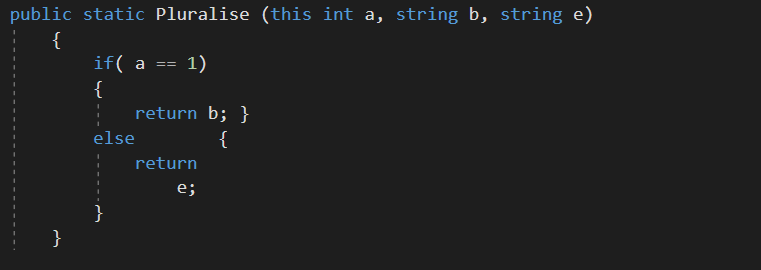
# Implementation

## Coding Practices

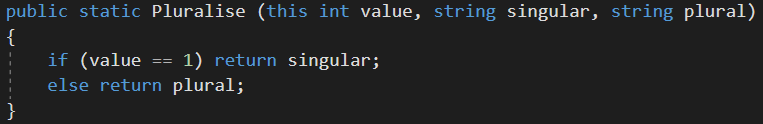
Basic coding practices are easy to implement and maintain and should serve to make life easier in the long run. These practices include:

1. Concise & Uniform Naming
2. Documented Source Code
3. Object-Oriented Implementation
4. One Line Per Statement
5. Simplify & Optimise

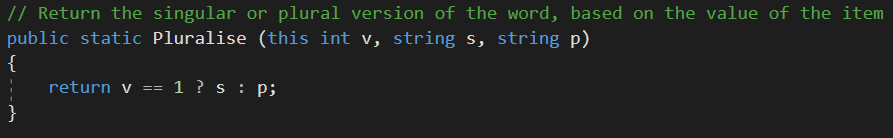
These coding practices are extremely simple to work by and support clear, concise and expandable source code. The following images compare examples of good and bad coding practices.



Undocumented, unformatted and long-winded version of the function.



Formatted, and simplified version of the function.



Simplified, optimised version of the function that has been documented

## Production

### Assets

As mentioned in the plan in Trimester One, asset production will be kept to an absolute minimum in order to maintain focus on features and mechanics of the gameplay. Environment, character and architectural assets have been sourced from Unity’s Asset Store. Content Packs, created by Synty Studios, were purchased for use in this project, and will be integrated with the final graphics of the product.

Further to this, character animations have been sourced from the free resource, Mixamo.com. This platform offers hundreds of motion-capture quality character animations, with pluggable values and Unity support. Sourcing animation assets was of utmost priority for the project’s assets, as this is the most time-consuming task.

Further assets that may not be able to be sourced will be created using 3D modelling applications in the event that these assets are needed. Finally, simple ambient animations and graphical effects will be simply created with tools that are native to Unity.

Examples of these assets are displayed below.



https://assetstore.unity.com/packages/3d/environments/fantasy/polygon-knights-pack-83694

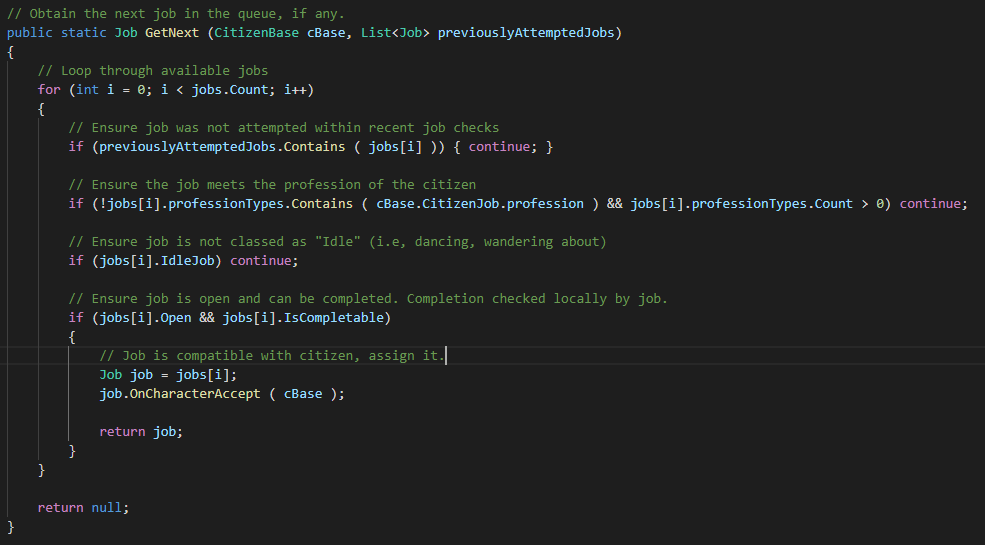


https://assetstore.unity.com/packages/3d/environments/fantasy/polygon-adventure-pack-80585

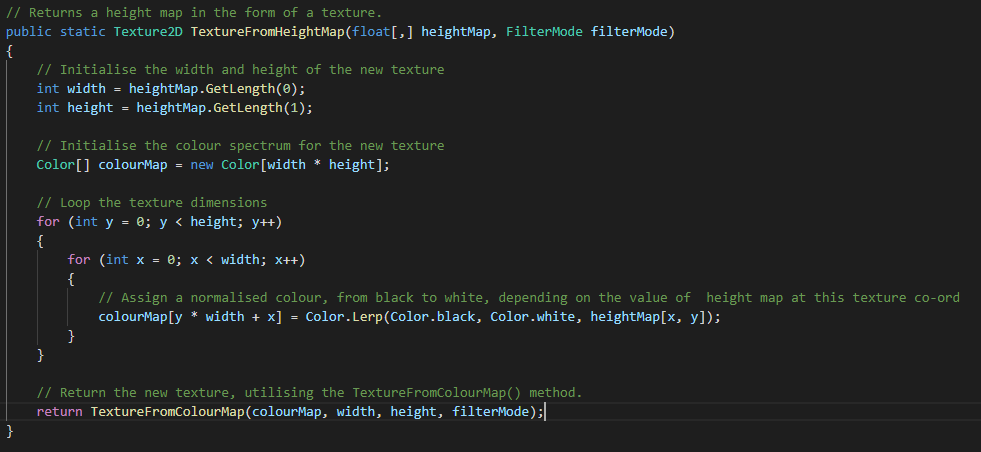
### Source Code

Source code created during the implementation stage of Township will use C# as the programming language. Visual Studio 2017 will be used as an IDE to utilise syntax highlighting, refactoring and debugging. On top of this, Visual Studio 2017 provides a Tools for Unity plugin.

Source Code for Township will follow many, if not all, of the standard coding practices detailed in section 3.2. On top of this, further practices will be used that may be more advanced than what has been mentioned. These techniques may include things such as Namespacing, Polymorphism, Inheritance and Interfacing. The following images show examples of source code that is currently implemented in Township.



Method that can be called by any Citizen who is looking for a job. The method finds and assigns the most suitable job.



Used in Procedural Terrain Generation. This method returns a texture that can be used to draw the terrain height.

## Level Design

### Terrain

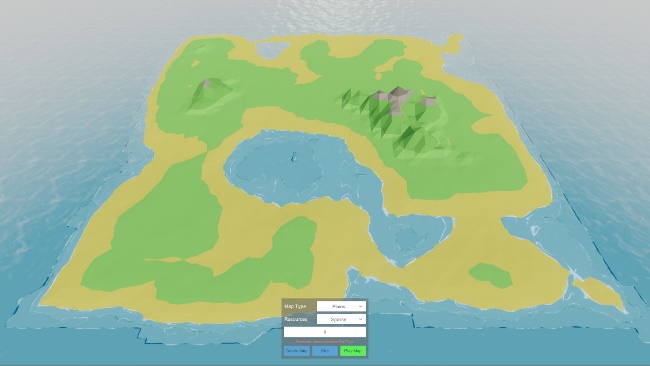
In Township, terrain is procedurally generated. From a gameplay perspective, this choice was made to ensure unique environments throughout each playthrough the player has. On top of this, it could be argued that, whilst procedural terrain is more programmatically advanced, it may be quicker to implement than designing each map or level by hand. In this case, I believe this to be true, it would have taken longer and been more tedious to design individual maps for Township. Finally, using procedural terrain provides a more natural looking terrain if the terrain designer is not very skilled.

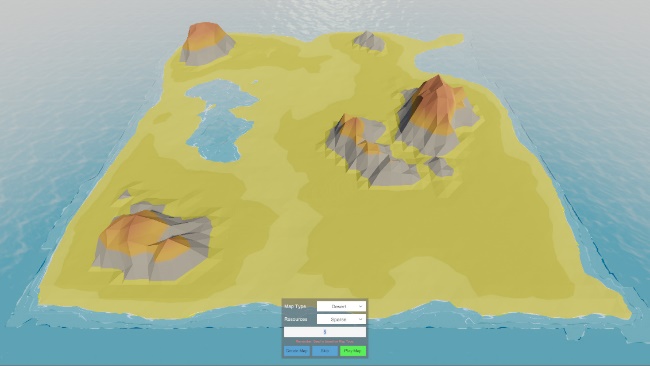
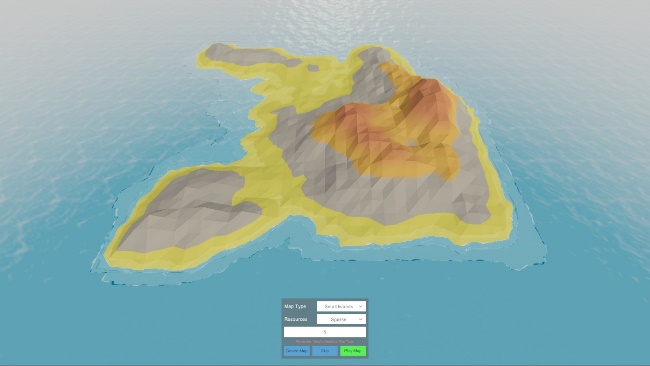
On the technical side of Township’s terrain, a simple noise map approach was employed. Other, multi-layered terrain generation approaches were tested; however, the same quality could be achieved compared to simple noise.

Three variables control the frequency, amplitude fall-off of the noise generated, with a third variable defining the octaves of the noise. Octaves indicate the iterations of noise generated, divided by the fall-off, with each iteration being layered on the previous. Using this in combination with scaling, height multipliers and custom curves, unique terrains can be achieved each playthrough.

Finally, each terrain is based off a “seed” that can be provided by the player. In doing this, the player is also able to play the same map twice if they happened to enjoy that playthrough.

Screenshots of in-game terrain are displayed below, each using different map-type presets and seeds.





### Space

### Texturing

### Light & Shadow

## Bugs Encountered

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Date** | **Resolution** | **Priority** |
| Test\_001 | 25/01/19 | Test\_001 | Low |
| Test\_002 | 25/01/19 | Test\_002 | Medium |
| Test\_003 | 25/01/19 | Test\_003 | High |

## Code Examples

## Art Production

The source column indicates where the artwork was sourced from, otherwise will state Custom to indicate it was created from scratch.

|  |  |  |
| --- | --- | --- |
| **Name / Description** | **Type** | **Source** |
|  |  |  |

## Audio Production

The source column indicates where the artwork was sourced from, otherwise will state Custom to indicate it was created from scratch.

|  |  |  |
| --- | --- | --- |
| **Name / Description** | **Type** | **Source** |
|  |  |  |

## User Interface

## Immersion & Engagement

## Mechanics & Features

## Gameplay

## AI

## Flow & Balance

# Quality Assurance Plan

## Project Plan Iterations

## Risk Assessment

## Quality Assurance Audit

# Playtesting

## Testing Methodology

A typical software testing methodology is split into two parts; Functional and Non-Functional. These are then divided further into specific testing scenarios.

Functional testing involves testing the software directly, whether it be an individual component of the system or the final build. Non-Functional testing is centred around the technical aspect of a product, such as performance & stress testing, security & vulnerability testing and hardware & software compatibility testing.[[1]](#endnote-1)

In regard to this particular project, Functional testing seems to be the most important. Non-Functional testing will be important too, however, not all of these parts will be required. For example, security & vulnerability is not an issue for a project of this scope.

Specific stages of Functional and Non-Functional testing are detailed in the following sections.

## Compatibility Testing

Compatibility Testing, a sub-stage of Non-Functional Testing, is a software testing strategy designed to ensure a piece of software is capable of running on multiple different platforms. These platforms can include specific hardware, operating systems, networks or applications.[[2]](#endnote-2)

There are two types of Compatibility Testing; Forward and Backward Compatibility Testing. Forward Compatibility Testing is used to verify that the software is functional on newer platforms, whilst Backward Compatibility Testing is used to verify that the software is functional on older platforms.

## Regression Testing

Regression Testing is a testing strategy that is used to verify the validity of recent minor and major changes to the source code of a piece of software. Regardless of whether the change was large or small, Regression Testing requires the change be verified.[[3]](#endnote-3)

In contrast to Unit Testing, Regression Testing dictates that many scenarios be re-tested, not just the component that was changed, as a small change in source code can have a ripple effect on other modules and components.

## Acceptance Testing

## Alpha & Beta Testing

## Functional Testing

Functional testing can be broken down into 4 specific parts. These parts are performed in order and should be performed both throughout the project, and at the end of the project. The four parts of functional testing are detailed below.

|  |  |  |
| --- | --- | --- |
| **Functional Testing** | | |
| **Stage** | **Description** | **Performed** |
| Unit Testing | Testing each individual component or module that makes up the overall application or system | Throughout |
| Integration (Scenario) Testing | Testing combined modules/components that have already been Unit Tested. | Throughout |
| System Testing | Testing of the entire system which has been fully Unit/Integration Tested and build as a final solution | End |
| Acceptance Testing | Performed at the end of the project, ensuring that all original requirements of the project have been met | End |

## White & Black Box Testing

# User Experience Analysis

## Questionnaire Analysis

## Feedback Analysis

## Tracked Player Issues

# Critical Appraisal

1. Inflectra.com. (2018). *Software Testing Methodologies - Learn The Methods & Tools*. [online] Available at: https://www.inflectra.com/ideas/topic/testing-methodologies.aspx [Accessed 1 Feb. 2019]. [↑](#endnote-ref-1)
2. Guru99.com. (2019). *What is Compatibility Testing? Forward & Backward Testing (Example)*. [online] Available at: https://www.guru99.com/compatibility-testing.html [Accessed 1 Feb. 2019]. [↑](#endnote-ref-2)
3. Smartbear.com. (2019). *What is Regression Testing? | SmartBear Software*. [online] Available at: https://smartbear.com/learn/automated-testing/what-is-regression-testing/ [Accessed 1 Feb. 2019]. [↑](#endnote-ref-3)