Prepared by: (Conor O’ Donoghue) (11/05/2020)

TABLE OF CONTENTS

1.0 INTRODUCTION --------------------------------------------------------------------------------------------------------- 2

2.0 OBJECTIVES AND TASKS -----------------------------------------------------------------------------------------------2

2.1 Objectives

2.2 Tasks

3.0 SCOPE ---------------------------------------------------------------------------------------------------------------------3

4.0 Testing Strategy---------------------------------------------------------------------------------------------------------3

4.1 Unit Testing

4.2 System and Integration Testing

4.3 Performance and Stress Testing

4.4 User Acceptance Testing

4.5 Batch Testing

4.6 Automated Regression Testing

4.7 Beta Testing

5.0 Test Schedule --------------------------------------------------------------------------------------------------------------

6.0 Control Procedures ------------------------------------------------------------------------------------------------------

6.1 Problem Reporting --------------------------------------------------------------------------------------------

6.2 Change Requests-----------------------------------------------------------------------------------------------

7.0 Features to Be Tested --------------------------------------------------------------------------------------------------

8.0 Features Not to Be Tested ---------------------------------------------------------------------------------------------

9.0 Resources/Roles & Responsibilities --------------------------------------------------------------------------------

10.0 Schedules ----------------------------------------------------------------------------------------------------------------

11.0 Risks/Assumptions -----------------------------------------------------------------------------------------------------

12.0 Tools------------------------------------------------------------------------------------------------------------------------

**Introduction**

The following test plan is for Pixel Wizard which is a 2D PC and mobile device game aimed at all age groups. The aim of this test plan is to insure there are no glitches/bugs in the game which could negatively impact our players. In order to insure the testing follows testing standards there is an independent development team carrying out the testing process. This improves the quality and accuracy of our test plan.

**Objectives/Tasks**

Objectives

* Find any possible defects which may have been created during the development process.
* Prevent further defects.
* Insure the game meets user standards.
* Gain confidence in the finished product.
* Insure the game meets the Business requirement specification and System requirement Specifications.

Tasks

Main Menu Test

1. On load the game must present 3 options, ‘Play’, ‘Settings’, and ‘Exit Game’.
2. Clicking Play from the main menu will start the game at level 1

2.1. When Game starts on level one controller screen will display.

3. Clicking Settings on the main menu will display sound and music level options.

3.1 Sound Option can be adjusted.

3.2 Music Option can be adjusted.

4. Clicking Exit Game will close the game completely.

In Game Test

1. At the start of level one the controller options description screen must pop up.
   1. Move forward = Right Arrow key/D
   2. Move Backwards = Left Arrow Key/A
   3. Jump = Up arrow key/W
   4. Crouch = C
   5. Attack = Left Mouse click/R
   6. Pause/Resume = Spacebar
2. At the end of level 1 the player should advance to level 2.
3. Level 2 should have increased difficulty.
4. Level 3 should also have higher difficulty.

**Scope**

The tests being completed are aimed at insuring the overall satisfaction of any players on the game, this means checking that controls are configured correctly, and menus lead to the correct places and there are no graphical or major bugs.

In order to complete the testing to the highest testing standards we will be following all major testing guidelines and also completing a range of tests, this includes unit testing, System and integration testing, Performance and stress testing, User acceptance testing, there is then a range of beta test which must be complete, this range from hardware requirements to control procedures and testing schedules.

Tactics

To achieve the best results the team overlooking the testing process are all independent developers from different departments to the game development team. This insures no bias or inaccurate testing.

In order to insure the game runs smooth under stress we will be running a variety of stress and load tests on the game, these will involve running the game with a high amount of players at once, running the game on various graphics levels and allocating low amounts of graphic card and ram processing power to see if the game will crash or lag.

During the beta test we will also be re testing all game levels and enemy spawns/health, we want to insure that if the game level is restarted, it will reset the AI to their original locations and also move the player to the starting point.

**Testing Strategy**

A Testing strategy is vital for a test plan to work. The main objective of a testing strategy is to outline what approach will be taken towards the testing. We also need to adhere to a strict timeline and insure we achieve the desired goals. This is to insure any and all glitches or defects are removed from the game before being exploited or discovered. We will also be developing a Testing Timeline which will serve as a guide for the team as to how much time they have to complete each task.

Participants: Conor, Liam, Ryan, Robert

4.1 Unit Testing

Definition:

Unit testing is when individual pieces of the software are tested. These components are often functions and methods, ranging from character movement, damage, abilities, menu items, etc. Unit testing should have one or more inputs and one output. By testing components like this we achieve an extremely specific and accurate result.

Participants:

Unit testing can be done manually however automating the testing process can speed things up further. Unit tests must be completed before any other form of testing may be applied. Ryan and Conor took the lead for the unit testing process, this involved creating test scripts for small units of code.

Methodology:

The scripts for this test will be written by Conor, they include several functions and methods which will be tested individually, and Ryan then ran each script on the desired component and recorded the result.

4.2 System and Integration Testing

Definition:

System and integration testing is defined as a type of software testing which is carried out in an integrated hardware and software environment to verify the behaviour of the entire system. Also used to evaluate the systems compliance with its specified requirements.

Participants:

Taking part in the system and integration testing is Robert and Liam who will be testing the controls and hardware/software interaction components in the game.

Methodology:

A data driven method will be used by the team as it minimises the usage of software testing tools, first a data exchange will take place between the system components. The behaviour of each data field will then be examined and recorded in order to give an overview of system performance.

4.3 Performance and Stress Testing

Definition:

Performance and Stress Testing is used to see how the game will handle under various levels of stress or load. It is a testing method which can determine the speed of a computer, network or devices, Load testing simulates real-world load on any application or website.

Participants:

Robert will be working on the performance testing for the game. Conor will handle the load testing which will test the client/server application capabilities.

Methodology:

This will involve determining how much load the game can handle, it will also check the general performance of the game and reduce the processing power continuously until achieving a fail result. This will provide a clear indication of minimum system requirements.

4.4 User Acceptance Testing

Definition: This is the last phase of the software testing process. During UAT, actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications listed.

Participants:

Liam and Ryan will be working on the User acceptance testing.

Methodology:

This will be completed as an end user through the client to verify the system before moving it on to the production environment.

4.5 Batch Testing

Definition:

Batch testing is where a group of tests are executing sequentially one by one, every test batch consists of multiple dependent test cases. In those batches every end state is base state to next case. Test batch is also known as test suit or test belt. Generally test engineers are executing test programs as batches because ‘End state of one test is base state to another test’. The result of one script failure or pass, fails or passes the whole batch test.

Participants:

Methodology:

4.6 Automated Regression Testing

Definition:

When regression testing is automated it allows for checks into a variety of changes and frees up testers to conduct manual exploration into more unusual cases in the production environment. Not all regressions are caused by new features or bug fixes.

Participants: Operating the software for this testing method will be Liam who has previous experience with these software’s.

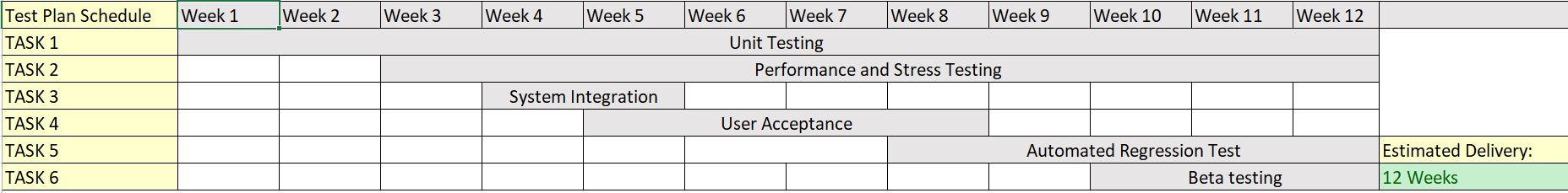
Methodology:

Regression testing will begin on week 8 and run for approx. 4 weeks.

4.7 Beta Testing

A beta test refers to the distribution of the pre-release version of the game to a select group of people who can test the game. At this stage of testing there should not be many bugs or glitches left in the game.

5.0 Test Schedule



6.0 Control Procedures

Control Procedures will be very important in this process as they provide a guideline for documenting and reporting bugs/glitches.

6.1 Problem Reporting:

Problem reporting is vital during the testing process, all results must be recorded so their required patches can be released by the development team.

6.2 Change Requests:

This is another document which will record the process of modifications to the software, it will also contain information regarding who conducted the test and signed off on the patch. If the patch will cause problems for any other modules then it must be listed in the document also as these must be identified so further patches can be done to them.  
  
7.0 Features to be tested

Menu:

The game load screen must be tested along with the 3 options it contains. These options range from ‘Play Game’, ‘Settings’ and ‘Exit Game’.

‘Play Game’ on click must place the player in level 1 with the controls screen displaying also.

‘Settings’ in the settings menu the player will be able to adjust the sound and music levels.

‘Exit Game’ this will close the game window.

From inside the game there is several tests which must be completed.

Level tests:

At the end of level one it is expected we will progress to level 2 which will be more difficult to level 1.

At the end of level 2 we expect to progress to level 3 with a higher difficulty again.

Game controls:

From inside the game we will use a keyboard and mouse to test the controls.

* 1. Move forward = Right Arrow key/D
  2. Move Backwards = Left Arrow Key/A
  3. Jump = Up arrow key/W
  4. Crouch = C
  5. Attack = Left Mouse click/R
  6. Pause/Resume = Spacebar

In game menu:

The in game menu must also be tested to insure all functionality works correctly and does not cause any incorrect redirects or bugs.

The in game menu will have 4 options. ‘Resume Game’, ‘Settings’, ‘Restart level’ and ‘Exit game’

Resume game should resume the level as normally.

Settings should show the player a menu where they can adjust sound and music levels.

Restart level should restart the game to whatever level the player is currently on.

Quit game should exit the game.

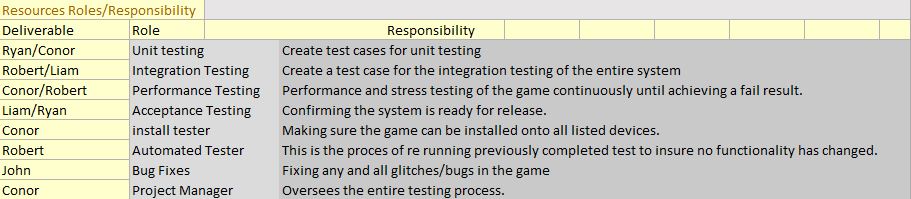
Components to be tested Breakdown:

* Front end game menu
* Game Controls Screen
* In game Menu
* Audio Bar settings
* Music Bar settings
* Jump
* Move Forward
* Move Backwards
* Move Right
* Move Left
* Attack
* Crouch
* Enemy Health
* Player Health
* Player Respawn
* Enemy Respawn
* Exit Game

8.0 Features not to be tested

* Save and load level system
* File Detection System
* Network speed
* Battery Consumption
* Loading time

9.0 Resources/Roles & Responsibilities



10.0 Schedules

11.0 Risks/Assumptions

12.0 Tools

In Order to test any system correctly it is vital to use the correct testing tools, the market has a variety of viable options for the testing process, below is a list of the chosen set of tools which were used in the testing process for the game.

Test Management Tools:

Practi Test: This is a cloud based innovative test management tool. A technology and methodology leader in the field of application life management. This service provides the tester with the best in class End to End system to meet their testing and QA needs.

Functional testing tools:

Selenium: This is a testing framework which is used to perform web applications testing across various browsers and platforms such as Linux, windows and Mac, Selenium helps its users create tests in various programming languages. These include, Java, PHP, C#, Python, Perl, Groovy and ruby. It offers record and playback features also which does not require the user to understand selenium IDE.

Load testing Tools:

J-Meter

The Apache JMeter application is an open source software which is used to load test functional behaviour and measure the performance.