Test Plan

for

The Pixel Wizard

Prepared by: Aleksandra Rzepka

12/05/2020

Contents

[1.0 INTRODUCTION 2](#_Toc40325783)

[2.0 OBJECTIVES AND TASKS 4](#_Toc40325784)

[2.1 Objectives 4](#_Toc40325785)

[2.2 Tasks 4](#_Toc40325786)

[3.0 SCOPE 5](#_Toc40325787)

[4.0 TESTING STRATEGY 6](#_Toc40325788)

[4.1 Unit Testing 7](#_Toc40325789)

[4.2 System and Integration Testing 7](#_Toc40325790)

[4.3 Performance and Stress Testing 8](#_Toc40325791)

[4.4 User Acceptance Testing 9](#_Toc40325792)

[4.7 Beta Testing Participants: 10](#_Toc40325793)

[5.0 TEST SCHEDULE 10](#_Toc40325794)

[6.0 CONTROL PROCEDURES 11](#_Toc40325795)

[7.0 FEATURES TO BE TESTED 12](#_Toc40325796)

[8.0 FEATURES NOT TO BE TESTED 13](#_Toc40325797)

[9.0 RESOURCES/ROLES & RESPONSIBILITIES 14](#_Toc40325798)

[10.0 SCHEDULES 15](#_Toc40325799)

[11.0 RISKS/ASSUMPTIONS 16](#_Toc40325800)

[12.0 TOOLS 17](#_Toc40325801)

# INTRODUCTION

A 2D platformer game designed for the player to beat. The playable character of a wizard will be able to make their way through the level using the keyboard. The player must face oncoming enemies which shoot at the wizard in order to kill them. The game contains “boss fights”, when the character us faced with a difficult boss to kill in order to advance further in the game. The player has a health bar at the top left of the screen which shows the amount of health remaining, this is represented by red diamonds, while the enemy’s health is represented in the top-right corner of the screen, blue diamonds represent the enemies health bar.

# 2.0 OBJECTIVES AND TASKS

## 2.1 Objectives

The goal of the game is for the player to control the character in such a way as to destroy any oncoming enemies in order to win the game. The player has the responsibility of gathering important artefacts throughout different levels of the game in order to make the final fights in the levels easier. If these artefacts are overlooked, the boss fights become significantly harder, as the basic weapons that are supplied at the start do not deal as much damage as those that can be found on the levels.

As stated in the Master Test Plan Document, the smoothness of the gameplay aka the movement of the player, the enemies the background as well as music and the shooting analytics are all tested in order to provide the player with the best possible experience. All the menus in the game, aka the start menu options (Play Game, Settings, Load Game, Delete Game, and Exit Game) are all tested by the developers as well as beta testers at different stages of development to provide feedback as to what needs to be changed before the full game release.

As well as the main game menu being tested, the game has a pause meu that the player is able to access easily while playing the game, there is a button in the bottom right corner of the screen that the player is able to press that opens a Pause Menu, this menu allows for the restart of the level, whish resets the player’s position and all the enemies on the level as well as all stats and weapons, through this menu the player is also able to save the game, adjust options (Graphic Settings, Music Level, and Sound Level) for the game session. This menu also includes the option for the player to exit the game without saving which takes the player back to the main menu screen.

As mentioned above according to the test plan the in-game functions are to be tested.

## 2.2 Tasks

The player movements, the enemy movements, the boss fights as well as levels and their increasing difficulty are all to be tested on different levels as stated above. Post-testing these areas can be improved and how this can happen is in the role of the software developer. If any issues arise, they are to follow the bug report and report any incidents with accordance to it.

# 3.0 SCOPE

General:

Level navigation, Menu Options and Saving Modules.

Tactics:

Level navigation, Menu Options and Saving Modules will firstly be tested by the software developers and software testers who are developing the levels themselves. The time for this is allocated in the test plan document. If any issues arise with the code at this stage of the making of the game, the leader of the software development team will contact the PM of the company who will then contact Game Development International Ltd. If the problem persists then more time will be allocated for the fixing of bugs and errors at this stage of development.

In turn the User Acceptance Team will run tests in order to determine whether the game is suitable to be released out to the market or not. If not, then the game will go back to software developers for more bug fixes and general fixes all around to make the game more suitable for the players.

# 4.0 TESTING STRATEGY

In terms of the different testing techniques which the game itself has undergone, the main code has gone through component, this tested the characters ability to move around, allowing the player to make the character move and shoot by pressing different keys on the keyboard. This test allowed our developers to check how the character behaves when a variety of keys are pressed whether it be simultaneously or one after the other. The developers found some trouble in this due to the complexity of the code required. The player’s health is not regenerated over time as random enemies drop health orbs for the player to collect in order to restore his/her health.

Each one of the components made individually for the game was tested individually; the enemies movements that are tracked by the path that they are o follow so as to not fall over “cliffs” or edges of the game, the amount of shots that have to be fired upon them in order for them to die and disappear off the screen, the enemies get harder as the levels go on and so these had to be adjusted after every level. The enemy shooting tactic and the way in which they can defend themselves had to also be implemented since the game itself would be easier to beat if the enemies did not become more advanced as the game went on. The enemies are walking back and forth on their platforms and from time to time drop health which the player can “pick up” to restore his own health.

The boss has also been tested through component testing. Each boss at the end of the level has different abilities, such as having more health than the one previous, and having stronger weapons which deal more damage to the player. As well as the boss’s do not move in the same way as the enemies (from one side of the platform to the other), boss’s follow the player around the screen in order to kill him/her, and are able to change directions in which their weapons fire.

The game menus were tested individually by the system developer, this involved coding to allow the player to save their game from any point in the levels, being able to load a saved game without major bugs or just letting the player start at the beginning of the level, the music and game sounds needed to be integrated with the game, so as to allow for sound effects produced by the gun as well as the death of enemies or the player.

Integration testing was done by the another team in order to understand what it was that they wanted from the game and what could have been changed in order to make the game more enjoyable for other players. This involved testing the game as a whole, the levels as well as the menus in order to find out any possible errors or bugs that were not found in the previous stages of testing. This uncovered a significant amount of errors, as each part was previously tested individually, however when put together a major amount of code needed reworking in order to allow for a smooth experience.

System tests were run in order to make sure that the game requirements set by the company were made. Such as having a moveable character, enemies spawning that attempt to kill the player, shooting weapons at the enemies and the player, having platforms throughout the level on which enemies are located and onto which the player could jump onto. A boss at the end of each level that increases in strength after the level in order to challenge the player.

Acceptance testing was done last through beta testers in order to make sure that the game meets their requirements before releasing it into the market. A game that was too easy would not have create that much interest with the public as one that challenges the player would have.

## 4.1 Unit Testing

Definition:

UNIT TESTING is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

Participants:

Zarah Wynn

Lacey-May Wainwright

Myla William

Sania Delarosa

Methodology:

Module Interface test: Test whether the game application is working properly as a single unit aka, whether each small component is doing exactly as it was coded to be doing.

Local data structures: To test whether the game is being saved or not.

Boundary conditions: To make sure that the program is working as it should be at its’ boundary condition.

Independent paths: All independent paths are tested to see that they are properly executing their task and terminating at the end of the program.

Error handling paths: To test if error handling path are handling errors properly or not and logging them into the system.

The Unit Testing Team is going to write the scripts for unit testing to be performed, the team will have a month to run any tests that they see fit in order to eliminate as many errors as they can before the application is fully finished.

## 4.2 System and Integration Testing

Definition:

System integration testing involves the overall testing of a complete system of many subsystem components or elements. The system under test may be composed of hardware, or software, or hardware with embedded software, or hardware/software with human-in-the-loop testing

Participants:

Jane Matthews

Mike Russo

Phil Dunphy

Jay Brogan

Michelle Browne

Nazifa Knox

Jevan Rios

Jercy Morrow

Taine Hussain

Sonny Hollis

Methodology:

Testing the application as a whole. Meaning checking every single part of the game in order to provide the best output possible for it.

The Unit Testing Team will write the tests that are to be carried out.

The testing will be done on a bottom-up testing basis this involves the integration of individual components ( which in this case have already been tested in the component testing stage ) in levels until the whole system is created.

## 4.3 Performance and Stress Testing

Definition:

Performance testing is carried out to check the system's performance under varying loads. Stress testing is carried out to check the behaviour of the system under the sudden increased load. It contains load and stress testing as components.

Participants:

Delia Petty

Roseanne Wise

Tyrique Ratliff

Rachelle Montes

Ihsan Mcneil

Methodology:

Performance testing is going to be done in a closed environment to check how the game holds up when faced with a real life gamer. The Stress Testing as well will be done by the testers team who will put the work done by the developers to a test in which the game will be tested on all aspects, so as to find hidden bugs that can be fixed before the release of the game.

The User Acceptance Team will write the test to be carried out by the testers.

## 4.4 User Acceptance Testing

Definition:

The purpose of acceptance test is to confirm that the system is ready for operational use. During

acceptance test, end-users (customers) of the system compare the system to its initial requirements.

Participants:

Delia Petty

Roseanne Wise

Tyrique Ratliff

Rachelle Montes

Ihsan Mcneil

Methodology:

The User Acceptance Team will perform the test.

This test will test the usability of the game, whether it is user friendly, whether the game controls are easy to operate.

4.5 Batch Testing

Definition:

Batch testing is a comprehensive test on your current trained model to measure its performance in LUIS. The data sets used for batch testing should not include example utterances in the intents or utterances received from the prediction runtime endpoint.

Participants:

Delia Petty

Roseanne Wise

Tyrique Ratliff

Rachelle Montes

Ihsan Mcneil

4.6 Automated Regression Testing

Definition:

Regression testing is the selective retesting of a system or component to verify that modifications

have not caused unintended effects and that the system or component still works as specified in the

requirements.

Participants:

Delia Petty

Roseanne Wise

Tyrique Ratliff

Rachelle Montes

Ihsan Mcneil

Methodology:

Each team member, after the software developers made changes to the code due to bugs and/or other incidents will test the application in order to verify that nothing is missing from it, that all the requirements are met and that the customer is satisfied with the finished product.

## 4.7 Beta Testing Participants:

Methodology:

Beta Testers will be picked at random from a group of participants in order to test run the game in case the developers missed out on any bugs, or plot holes or failures within the game that previously run tests did not detect.

# 5.0 TEST SCHEDULE

Test milestones as identified in the Software Project Schedule:

Research --

- Type of Code and different properties – 1 month

- Game Aim – 2 weeks

- Usage of Programs – 2 weeks

- Possible game difficulty choices – 2 weeks

- Possible procedures to create games – 2 weeks

Planning –

- Game Decision – 2 weeks

- Code Choice – 1 week

- Target Audience – 1 week

- Choice of development Program – 1 week

Implementation

- Creation of sprites – 2 weeks

- Controls creation – 1 month

- Music Finding – 1 month

- Coding – 2 months

Testing

- Self testing – 2 months

* Testing each one of the components of the game – 1 month – Software Developers
* Testing the integration between the components – 1 month – Software Testers

- External Testing – 2 months

* System Testing – 1 month – Software Testers, Acceptance Testing Team
* Acceptance Testing – 1 month – Acceptance Testing team

Documentation/Paperwork – 1 month

# 6.0 CONTROL PROCEDURES

Problem Reporting:

When an incident is encountered data is to be collected about the incident; from what area of the game it came from and what was done in order to get the incident to happen eg. Pressing two buttons at the same time. Highlight the message as in find out what exactly caused the bug to occur. Look through the code to see what is not in the correct place and could’ve caused the error to occur. Find out the mistake. Iterate to fix the error.

The main coordinator Nick Forde signs off at the different changes made to the project. Any type of change even the smallest one is to be reported as it could hinder the execution of the other stages of developing the project.

# 7.0 FEATURES TO BE TESTED

The gameplay; moving the character around the screen and the progression of the level, as in random platforms being generated throughout the level with enemies on them that attack the player if they get too close to them. These enemies would get harder and beat as the levels go on.

The bosses of each level get harder to beat as the levels progress, their health bar increases as well as their weapon strength dealing more damage to the player.

The menu screens such as the main menu screen, where the player can start the game, load a saved game, change the music and sound as well as graphic settings are to be tested in order to establish a balance between them. The pause menu screen will allow the player to save the game and change the settings same as they can be changed in the main menu settings. Both menus will also allow the player to exit the game.

All these features are to be tested altogether as if one was playing the game in order to check if all these main features work.

# 8.0 FEATURES NOT TO BE TESTED

Every feature is to be tested, if time permitted. However if not then the main gameplay feature is to be tested the rest can wait until the release when bugs are fixed.

# 9.0 RESOURCES/ROLES & RESPONSIBILITIES

Staff involved in the making of this project are :

Game Development International Ltd *(Project Sponsor)*

Jennifer Smith and Dylan Greene *(Subject Matter Experts)*

Mike Russo *(Project Owner)*

Haley Murphy *(Project Manager)*

Nick Forde *(Technical Lead)*

Jane Matthews, Mike Russo, Phil Dunphy, Jay Brogan, Michelle Browne *(Software Developers)*

Jane Matthews, Mike Russo, Phil Dunphy, Jay Brogan, Michelle Browne, Nazifa Knox, Jevan Rios, Percy Morrow, Taine Hussain, Sonny Hollis *(Software Testers)*

Zarah Wynn, Lacey-May Wainwright, Myla William, Sania Delarosa *(Unit Testers)*

Delia Petty, Roseanne Wise, Tyrique Ratliff, Rachelle Montes, Ihsan Mcneil *(Testers)*

Calum Blundell, Tyla Bellamy, Adrienne Bevan, Rodney Mclaughlin, Ella-May Hills *(User Acceptance Testers)*

# 10.0 SCHEDULES

Documents which set a schedule:

- Test Plan

- Test Cases

- Test Incident Reports

- Test Summary Reports

# 11.0 RISKS/ASSUMPTIONS

In the case of running behind schedule the developers on the team are asked to stay overtime in order to get the project finished in time.

In the case of running behind on test items the staff is asked to stay on night shifts in order to get the project back on track.

In the case of the team members not collaborating well meeting are to be put in order to resolve any issues that may arise between the team members.

If all tests are not accomplished, due to the complexity of the project only the most important ones are to be performed. (Component Testing and Integration)

Automation is to be implemented as much as possible in order to speed up the way in which the tests are being run.

Provide for the right testing tools and environments for the team to be able to accomplish as much as they can before the deadline runs out.

Have a collection of past tests that were run in order to have a record of past bugs and errors, this may help with any bugs or errors of the same type that arise.

# 12.0 TOOLS

Automation Tools:

TestComplete – SmartBear

LEAPWORK

XRAY

Bug Tracking Tool:

BugHeard