**Test Plan Template: The Pixel Wizard**

**Ryan Higgins** [**G00337350@gmit.ie**](mailto:G00337350@gmit.ie)

**3rd Year Software Dev. Project**

**TABLE OF CONTENTS**

1.0 - INTRODUCTION

2.0 - OBJECTIVES AND TASKS

2.1 - Objectives

2.2 - Tasks

3.0 - SCOPE

4.0 - Testing Strategy

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

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

**1.0 INTRODUCTION**

Game Development International Ltd have requested for me to develop a test plan for the new game that they have just designed. The product being tested is a new game called “The Pixel Wizard”. The game which I will be testing is a 2D side-scrolling platformer game. In this game the player controls a wizard character which must go through levels and each level contains enemies which they must defeat to progress to the next level and at the end of each level there is a boss character.

Main game movements – The game is both playable on pc and phone which will have two different control methods. To move forwards and backwards in the game it is simple with the mobile platform as were are having arrows on the screen which you press and hold while on pc you simply use the (D,A) keys for forwards and backwards or the arrow key in that direction. To crouch use an arrow on mobile and on PC you hold the (C) key. To attack as your character on phone you use the dedicated attack button which will be clearly visible. On PC the user simple left clicks or clicks (R). Inside the game there is also the available function of pausing the game this can be done on mobile using the button in the top right of the screen or done on PC using the spacebar.

Menus – Once the game is launched the main menu will load. There will be several different buttons which can be pressed which are “Play Game”, “Settings”, “Load Game”, “Delete Game”, and “Exit Game”. When you press the “Play Game” button the game will be then loaded and this will always load the game into a new game. The “Settings” button will bring the user to a menu where the player can alter settings such as mute sound and volume level. Load game will allow the user to load a previous game in which they saved while previously playing the game and they used the save button while the delete button allows the user to delete a previously saved game. Exit game will just exit the user from the application.

In-Game menus – The in-game menu is displayed when the game is being played and is accessed when the space bar is pressed during the game. This then pauses the game. The menu contains 3 buttons which are “Save Game”, “Settings” and “Exit Game”. Save game allows the user to save the game and then access it whenever they want in the future. Settings allows them to change the music and sound volume. And the exit button will close the game.

**2.0 OBJECTIVES AND TASKS**

**2.1 Objectives**

In this section we will cover the testing objectives, what needs to be tested and who is responsible for testing. The objectives for our testing are to ensure every aspect of the game works as designed and for the users experience to be as good as possible.

Here is a list of objectives:

* Create a teams and add in all members participating in the project.
* Make test cases and divide the workload for each team. Complete all test cases
* Create guidelines and deadlines and try to stick if possible.
* Conduct daily meetings between all teams to ensure goals are being met.
* Form all test cases into a report and include expected and actual results
* Send report back to Game development International Ltd.

**2.2 Tasks**

The tasks are broken down are divided amongst the three teams which you can see below. Each team will be responsible for certain testing aspects of the game as it’s a team effort to ensure the game goes to plan. In each team there will be a team leader which will be appointed to ensure everything runs smoothly.

List of tasks:

* Test menu functions which includes Loading the game, Saving the game, Reloading the game and finally deleting the game.
* Test Game Interface.
* Test all features in a large batch test
* Perform all the tests which will be depicted later.
* Test all controls function accordingly.

I would like if there were three teams. Team 1 would concentrate on the actual game as in game functionality and just all the general aspects of the game. Team 2 would concentrate on the opening of the game and all the opening menus. Team 3 would then

**3.0 SCOPE**

**General**

In the test plan we are planning to test all aspects of the game. I have made a list of the different aspects which will be tested which is above in the list of tasks.

**Tactics**

To accomplish a very high level of testing for this game we have decided that one of the teams which I assigned above will be testing different parts of the game independently and then they will be telling the team leader on there progress and if they are fulfilling deadlines. The types of testing we will be using are unit testing, performance stress testing and a beta test.

**4.0 TESTING STRATEGY**

Our approach to testing is going to be a group effort which is split between 3 teams each having their own responsibilities and have there own tasks to achieve but then at the end of this we want to incorporate all this together. We will be using many different testing methodologies which I will explain below.

**4.1 Unit Testing**

**Definition:** Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. The main objective of unit testing is to isolate written code to test and determine if it works as intended.

**Participants:** All teams will be using aspects of unit testing as they will be individually testing different small parts of the game to ensure each part is working correctly.

**Methodology:** Were carrying out the tests to make sure the games behaving as it should be and that all features are working as they should be. If any bugs errors found, they will be fixed.

**4.2 - System and Integration Testing**

**Definition:** Is defined as a type of software testing that’s carried out to verify the behavior of the complete system. It also will verify the behavior of the complete system

**Participants:** All teams will be taking part in these tests.

**Methodology:** We are carrying out these tests to ensure the game is behaving as we expected and that all independently tested aspects are now working together. If this is not the case, then the game will be fixed before we move on.

**4.3 - Performance and Stress Testing**

**Definitions:**

**Stress Testing:** Refers to testing software/hardware to determine if its performance will work under any extreme or unfavorable conditions that could be due to heavy amount of people using it or process loading.

**Performance Testing:** Refers to process of determining speed responsiveness and stability of a computer or network under a heavy workload. It can involve quantitative tests.

**Participants:** All teams are involved in the performance testing of this game.

**Methodology:** If the game is not working properly after some of the performance or stress tests have been done we would expect them to contact the team leader. We would want all these same tests to be carried out on a normal member of publics so they could then try and test it from a different point of view.

**4.4 - User Acceptance Testing**

**Definition:** User acceptance testing is basically testing the software by the user or client to determine whether its tested or not. This is the final stage of testing that is performed. It is also knows at (UAT)

**Participants:** Teams must be on standby and there is real people that will be using this game.

**Methodology:** As soon as all teams complete testing objectives any errors are then corrected which will make the game ready for users to play. Instead of instantly releasing it we have made a beta testing which will allow players to play the game and report back any bugs or issues that they encountered.

**4.5 - Batch Testing**

**Definition:** Batch testing is performed by running the entire test set. All automated test scripts are executed one at a time by keeping the other scripts in waiting mode.

**Participants:** All teams are to be involved.

**Methodology:** Any issues found during this time then must be fixed by the team responsible for that areas before the release of the game.

**4.6 - Automated Regression Testing**

**Definition:** is defined as a type of software testing to confirm that a recent program or code change has not adversely affected existing features. Regression Testing is nothing but a full or partial selection of already executed test cases which are re-executed to ensure existing functionalities work fine.

**Participants:** The teams are working through each area of their testing plan but if any issues are found they need to first report it and secondly then fix it.

**Methodology:** As teams work through the parts they are testing if any issues are found and fixed it should be subject to regression testing to ensure the code and game are still working.

**4.7 - Beta Testing**

**Definition:** Beta Testing is performed by real users of the software application in a real environment and can be considered as a form of external User Acceptance Testing. Beta version of the software is released to a limited number of end-users of the product to obtain feedback on the product quality. It is the final test before shipping a product to the customers.

**Participants:** The teams and real people are participants of this.

**Methodology:** This is the last step before large release of this game to the public and is the last place for errors to be fixed before release. All teams must be on standby to try and find any errors reported regarding the areas they have tested.

**5.0 – TEST SCHEDULE**

Below is going to be a breakdown of the weeks and where we hope to be and at what stage. If any issues may arise, we can tweak this plan slightly to allow extra time.

Week 1 – Have a group meeting with each team at 12 every Monday, where they meet for first time and the team leader will assign the work into each of the teams. These people will then start to begin testing on the product and report back to the leaders.

Week 2 – Meet at the same time 12 every Monday, discuss how work has been going and present issues found.

Week 3 - Monday meeting again at 12pm where all issues found during testing are presented and then assigned to be fixed.

Week 4 - Monday meeting again at 12pm where issues and fixes are discussed. If fixes have been completed and all teams are happy beta testing can commence this week.

Week 5 - Monday meeting again at 12pm where current ongoing beta testing and any issues found are discussed among the teams. Beta testing continues for this week.

Week 6 - Monday meeting again at 12pm where beta testing is closed, and all issues found are addressed.

Week 7 - Monday meeting again at 12pm where all teams continue working on issues that were reported in beta testing. If code is running perfectly they can implement all these changes.

Week 8 - Monday meeting again at 12pm where the now finished game is published.

**6.0 - Control Procedures**

**Problem Reporting:** If you happen to encounter an error, please request the team leader of your team to help you with this. Please write down in a word document where the error is in the game along with a brief description of what is going wrong. This document will be sent to the team leader who can show the project manager who will decide what will happen. Each team leader needs to keep all problem reports contained in a document to be presented to the project manager called “Problem Reporting [team number]”.

**Change Requests:** If changes are being made to the software the team leader must be informed before anything is changed. Changes you are making need to be stated on a word document along with why they are necessary. If these changes impact other parts of the game these need to be included because if another team is working on other parts you are impacting, they need to be notified.

**7.0 FEATURES TO BE TESTED**

As discussed before all features of this game need to be tested before the game is released.. Every aspect of the game is going to be undergoing testing. I’ve made a list below of what will be tested.

* Main game movements
* Shooting
* Menus
* In-Game Menus
* In-Game Items

**8.0 FEATURES NOT TO BE TESTED**

There are no functionality aspects of this game that we are not going to test.

**9.0 – RESOURCES / ROLES & RESPONSIBILITIES**

The roles for this project are down to the team managers to assign. The teams 1-3 have been assigned by management according to people’s skills and spreading out people with expertise in every area to be in each group.

The team leaders as follows and they are responsible for assigning jobs to people, and time management of the testing cases.

Team 1 Leader – Johny Martin

Team 2 Leader – Ryan Higgins

Team 3 Leader – Donal Cafferky

**10.0 SCHEDULES**

Test plan and test cases need to be submitted to management by each team at the start of this testing process. There are two key documents leaders need to submit which are “Problem Reporting and Changes made, which will be used to track the changes made to the game.

**11.0 RISKS/ASSUMPTIONS**

There are a few risks with this project. If there are any delays it could lead to a late release which would not look good on the company. But we believe the schedule allows spare time so if we run into any problems that delay us for a week or so we will be fine.

**12.0 TOOLS**

Selenium, Bugzilla, Microsoft Teams.

**SCREENCAST**

Link to screencast - <https://www.youtube.com/watch?v=j-_N-huGu_g&feature=youtu.be&hd=1>