

The Pixel wIZARD

Prepared by Adam Varden



May 14, 2020

Contents

[1.0 Introduction 3](#_Toc40375344)

[2.0 Objectives and Tasks 3](#_Toc40375345)

[2.1 Objectives 3](#_Toc40375346)

[2.2 Tasks 3](#_Toc40375347)

[3.0 Scope 4](#_Toc40375348)

[General 4](#_Toc40375349)

[Tactics 4](#_Toc40375350)

[4.0 Testing Strategy 4](#_Toc40375351)

[4.1 Unit Testing 5](#_Toc40375352)

[Definition: 5](#_Toc40375353)

[Participants: 5](#_Toc40375354)

[Methodology: 5](#_Toc40375355)

[4.2 System and Integration Testing 5](#_Toc40375356)

[Definition: 5](#_Toc40375357)

[Participants: 5](#_Toc40375358)

[Methodology: 5](#_Toc40375359)

[4.3 Performance and Stress Testing 5](#_Toc40375360)

[Definition: 5](#_Toc40375361)

[Participants: 5](#_Toc40375362)

[Methodology: 5](#_Toc40375363)

[4.4 User Acceptance Testing 6](#_Toc40375364)

[Definition: 6](#_Toc40375365)

[Participants: 6](#_Toc40375366)

[Methodology: 6](#_Toc40375367)

[4.5 Batch Testing 6](#_Toc40375368)

[Definition: 6](#_Toc40375369)

[Participants: 6](#_Toc40375370)

[Methodology: 6](#_Toc40375371)

[4.6 Automated Regression Testing 6](#_Toc40375372)

[Definition: 6](#_Toc40375373)

[Participants: 6](#_Toc40375374)

[Methodology: 6](#_Toc40375375)

[4.7 Beta Testing: 7](#_Toc40375376)

[Participants: 7](#_Toc40375377)

[Methodology: 7](#_Toc40375378)

[5.0 Test Schedule 7](#_Toc40375379)

[6.0 Control Procedures 7](#_Toc40375380)

[Problem Reporting 7](#_Toc40375381)

[Change Requests 7](#_Toc40375382)

[7.0 Features to Be Tested 7](#_Toc40375383)

[8.0 Features Not to Be Tested 7](#_Toc40375384)

[9.0 Resources/Roles & Responsibilities 7](#_Toc40375385)

[10.0 Schedules 7](#_Toc40375386)

[11.0 Risks/Assumptions 7](#_Toc40375387)

[12.0 Tools 7](#_Toc40375388)

# Introduction

The game is a 2D side scrolling platformer. The player has control over specific character. The character will battle and journey through levels. The game will begin with a splash screen for the main menu with various options to choose from such as play, setting, load game, delete game and exit game. When playing the game, the character uses magic to defeat the enemies encountered. When the game is paused a popup, screen comes up allowing the user to save the game, go to settings or exit the game. The user can move forward, move backwards ,jump , crouch, attack, and pause/resume.

# 2.0 Objectives and Tasks

## 2.1 Objectives

The objective of the game is for the player is to be able to move and shoot at ongoing enemies and transverse through the different level. The game must get significantly harder as the player progresses to challenge the players skills. They should be able to gain powerups and health toe ensure gameplay can run if possible, without major difficulty. Functionality like this are important for a successful game

Functionality is not the only thing that needs to be focused the overall aesthetic and how the game looks plays a key factor. The game is a side scrolling platformer. We must be insured that the background will move on progression with the game, in the sample level it shows the platforms where the user may jump up onto to get away from enemies tests need to be done to ensure these work how they should.

The splash screens need to work accurately as they are the first things seen by the player e.g. the main menu. It is all good if the game works but if the user cannot play the game it is a major fault and hit to everyone. Our form of communication with one another will be Microsoft teams’ meetings will be held once a week to ask about test progression and bugs discovered so far in testing. Our records of discovered bugs will be recorded on an excel spreadsheet with the area tested instructions done to test it and what we received or did not receive.

## 2.2 Tasks

The product will be tested bit by bit to ensure complete functionality and harmony among all components. The discovery of bugs and errors affecting other components will be recorded in our excel spreadsheet where the developer will be able to see the area tested, the error and what we discovered. Post testing all this information gathered will be given to the developing team to make these changes to the product.

# 3.0 Scope

## General

We will be testing all functionality of the product in small increments. Functionality such as player movement, player health, player powerups, enemy movement, enemy spawning, boss movement, boss spawning, background and platform interactions, the splash screen functionality which allow the player to play, save and delete the game.

## Tactics

I will first divide the features evenly among the team members for a fair and even workload. Issues that would arise will be noted and delivered to the appropriate people to resolve the issues. The issues will hopefully be resolved as soon as possible so that testing time is not set back. With the meeting that will happen once a week to give a progress update will enlighten us to the schedule changes needed for a more efficient testing period for those who may finish there allocated list will reassigned temporarily to another group where progress is slow.

# Testing Strategy

By breaking down the features evenly we allow for greater specification of testing testers will be able to focus on their part and give a more accurate test and discover the hidden bugs and errors that may be hiding in the code.

Featured that would be grouped together would be a grouping of player features where a group of testers will test all the player functionality in a variety of ways like that when the player jumps the drop down is a drop and not a float down other examples would be that the player remains on the ground and doesn’t fall through this would be testing the colliders that would be on the player so it can interact with other came components. We would test the health function again making sure the colliders work when the player is hit it will trigger the necessary code and output that the player has lost a life. These sort of player features will be grouped together and tested

Another group will take all functionality related to the enemies and bosses as they would have similar attributes to harm the player they would be grouped together. We would test how they interact with the platform and that their movement I works accurately. We would also test that they spawn at correct moments.

We would then use integration testing when all units have been tested to see how they all interact one another because the player may not be able to kill the enemy and boss but the boss and enemy may be able to kill the player.

Tests will be run on the overall game to test its behaviour under different system environment. Seeing it in a different environment would give a better insight of its requirements for most enjoyable gameplay.

We would test to make sure they spawn correctly and not all at once and that there isn’t an unbeatable amount for the player this would be tested using beta testing where it would be tested with people with no knowledge of the game and see how they find the difficulty and is it a reasonable challenge and not something they cannot beat.

## 4.1 Unit Testing

### Definition:

Testing individual units of the product to get a accurate test of the products specific component

### Participants:

May Mcgee

Ellisha Osborne

Yusra Cross

### Methodology:

May Mcgee will be responsible for testing all the players functionality and the main menu splash screen. Ellisha Osborne will be responsible for testing the boss and enemy’s functionality and the pause menu splash screen. Yursa Cross will test the background and platform interactions and collisions and will test that the game can be deleted and saved.

## 4.2 System and Integration Testing

### Definition:

Once unit testing is completed it will move on to the integration testing where all the components tested in unit testing will be tested as a whole and see how each component works with each other to see if there is an conflicting functionality.

### Participants:

May Mcgee

Ellisha Osborne

Yusra Cross

### Methodology:

With the knowledge from unit testing the participants will test the game with their previous knowledge they will notice any loss or errors in functionality. We will see how the enemies and players interact with the background and platforms. We will test how the player and enemy/boss interact with one another and vice versa. We test that both the save and delete game work with one another.

## 4.3 Performance and Stress Testing

### Definition:

Stress Testing is putting the program under extreme circumstances in order to test how it can handle a high-level situation and see where its breaking point is. Performance testing checks the behaviour during varies load instances

### Participants:

Alexia Paterson

Duncan Kidd

Bonnie Marsh

### Methodology:

Stress testing will be carried out by having the game played on higher levels of difficulty to see if the game can handle multiple enemies and bosses on the screen shooting at the player and see is the reaction time of the player changed by the level on moving assets in the game. For performance testing we will use varying levels with various levels of difficulty and see how the game holds up

## 4.4 User Acceptance Testing

### Definition:

The customers will test the product to see does it match all the business requirements they wanted to be in it.

### Participants:

Rahma Ryder

Paul Friedman

Dolores Kaye

### Methodology:

The customer will be invited in and left with the product and they will be left to review the game and note anything they wish to be changed or something that hasn’t been implemented and discuss it over with the participants of this test.

## 4.5 Batch Testing

### Definition:

A group of tests running one by one, one failed test results in the whole batch test failing.

### Participants:

Alexia Paterson

Duncan Kidd

Bonnie Marsh

### Methodology:

A series of tests will be run and one failure will result in an entire relook into the product.

## 4.6 Automated Regression Testing

### Definition:

When code that has been changed that doesn’t adversely affect other functionality will run through previously used tests that are re-executed.

### Participants:

Rahma Ryder

Paul Friedman

Dolores Kaye

### Methodology:

When the tester has discovered a bug in the code and noted and reported it to the appropriate developers. We will rerun the previous test to see is the bug resolved.

## 4.7 Beta Testing:

### Participants:

Halima Gibbons

Siobhan Fenton

Mathew Morgan

### Methodology:

We will put the participants in separate rooms and monitor them using a camera and note how the participants interact with the product.

# Test Schedule

|  |  |
| --- | --- |
| Test | Time Period |
| Unit Testing | 2 weeks |
| System and Integration Testing | 1 week |
| Performance and Stress Testing | 2 weeks |
| User Acceptance Testing | 1 day |
| Batch Testing | 2 weeks |
| Automated Regression Testing | 3 days |
| Beta Testing | 2 days |

# Control Procedures

## Problem Reporting

When encountering an error, it is to be noted exactly where in the code it occurred, what it did, what it should have done and level of severity. A screenshot of the code and what it outputted should be added for added clarity.

## Change Requests

When the tester comes across functionality that doesn’t cooperate with other functionality like what could be found in integration testing it should be noted in a report with the type of test, the error, the conflicts and the severity.

# Features to Be Tested

Features that will be tested will be all the gameplay functionality. The main menu functionality for playing the game and saving and deleting. The player and all its functions will be tested because it’s an important component along with the bosses, enemies and the platform where they all interact. Along with all of them being tested separately they will be tested on their harmony together.

# Features Not to Be Tested

Features that will not be tested will be the volume settings and music settings as in the hierarchy on priority they are low on the hierarchy.

# Resources/Roles & Responsibilities

**Project Sponsor:** Game Development International Ltd (Project Sponsor)

**Software Developer Team:**

Rahim Medrano

Arlo Finnegan

Safah Brewer

**Unit Testing System and Integration Testing Team:**

May Mcgee

Ellisha Osborne

Yusra Cross

**Performance and Stress Testing Batch Testing Team:**

Alexia Paterson

Duncan Kidd

Bonnie Marsh

**User Acceptance Testing Team:**

Rahma Ryder

Paul Friedman

Dolores Kaye

**Beta Testers:**

Halima Gibbons

Siobhan Fenton

Mathew Morgan

# 10.0 Schedules

# 11.0 Risks/Assumptions

# 12.0 Tools