

**SIMPLY RUGBY TEST STRATEGY DOCUMENT**

**Project Name:** SimplyRugby

**Project Number:** 1

**Prepared For:** Simply Rugby

**Prepared By:** Chris Dworczyk

**Creation:** 22/1/2018

**Last Updated:** 25/4/2018

**Date Submitted:** 02/5/2018

**Class:** HND Software Development

**Version Number:** 01

**Security Classification:** Low

# Document Control

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision No** | **Revision Date** | **Description of Change** | **Author** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Document Detail

|  |  |
| --- | --- |
| **Title** | SIMPLY RUGBY TEST STRATEGY DOCUMENT |
| **Version** | V1 |
| **Date** | 25/4/2018 |
| **Electronic File Name** | SimplyRugby TestPlan(Chris Dworczyk) |
| **Electronic File Location** | G:\HND\Graded Unit\Testing |
| **Author** | Chris Dworczyk |
| **Contributors** | N/A |

# Referenced Documentation

|  |  |  |
| --- | --- | --- |
| **Ref** | **Document Name** | **Electronic File location** |
| Functional validation | SimplyRugby - Functional\_Acceptance\_TestLogs | G:\HND\Graded Unit\Testing\Functional |
| Event validation | SimplyRugby - Event\_Validation\_TestLogs | G:\HND\Graded Unit\Testing\Functional |
| Navigation validation | SimplyRugby - Navigation\_TestLog | G:\HND\Graded Unit\Testing\Functional |
| Data Validation | SimplyRugby - Data\_Validation\_TestLogs | G:\HND\Graded Unit\Testing\Functional |

# Team Members and Roles

|  |  |
| --- | --- |
| **Resource Name** | **Role** |
| N/A | Project Manager |
| N/A | Programmer |
| Chris Dworczyk | Tester |

# Diary/Log of Errors

|  |  |  |
| --- | --- | --- |
| **Date** | **Error Description** | **Action** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Table of Contents

[Document Control 2](#_Toc512866133)

[Revision History 2](#_Toc512866134)

[Document Detail 2](#_Toc512866135)

[Referenced Documentation 2](#_Toc512866136)

[Team Members and Roles 2](#_Toc512866137)

[Diary/Log of Errors 2](#_Toc512866138)

[1.0 Introduction 4](#_Toc512866139)

[2.0 Objectives 4](#_Toc512866140)

[3.0 Scope 4](#_Toc512866141)

[4.0 Test Items 4](#_Toc512866142)

[5.0 Testing Methods 4](#_Toc512866143)

[6.0 Testing Strategy 5](#_Toc512866144)

[7.0 Testing Levels 5](#_Toc512866145)

[7.1.4 Acceptance Testing 6](#_Toc512866146)

[9.0 Exit Criteria 6](#_Toc512866147)

[10.0 Environmental Needs 6](#_Toc512866148)

[11.0 Staffing and Training 6](#_Toc512866149)

[12.0 Test Schedule 6](#_Toc512866150)

[13.0 Testing Deliverables 7](#_Toc512866151)

[15.0 Features not to be Tested 7](#_Toc512866152)

[16.0 Risks and Contingencies 7](#_Toc512866153)

[17.0 Approvals 7](#_Toc512866154)

# Introduction

This test plan will be written on the current project in the works for the SimplyRugby club in which I have been tasked in creating a database system to keep track of all their player information and player skills. In the program there are some constraints that need to be considered both for the users using the system and the data being inputted and stored. These will have to be rigorously tested to ensure a reliable and robust system.

# Objectives

I will be using separate test logs to document features and bugs of the system. These will be helpful in making the system as robust and reliable as possible. All major bugs and crashes will be address and a fully functional program will be provided. Moreover, most if not all minor bugs and glitches will also be taken care of to deliver a good software experience.

# Scope

I will test the functionality to ensure it matches the specified functionality in the requirement specification. Data validation tests will make sure the program only accepts the correct data and doesn’t throw any errors when forms are submitted. As well all the events will be tested so make sure all the UI controls respond correctly and in an extend manner. Overall navigation will be also tested to provide a smooth experience when using the program.

# **Test Items**

* Requirement specification
* User Guide
* Data Validation Test Logs
* Event Validation Test Logs
* Functional Acceptance Test Logs
* Navigational Test Logs

# Testing Methods

In this case I will be using the black box testing approach. For this the software will be placed in a metaphorical box where the inside code is not looked at but rather we will be feeding it test data and document carefully the outputs.

This approach helps the customer see that all the functional requirements are met and perform to their standards, the code and structure is not necessarily shown as they may not fully understand all the inner workings but rather they are only interested in the software working as intended.

Also, with this approach we will be able to create some of the test data with the already provided functional requirements. For example, to test the login feature (functional requirement: FR001) we would use 3 separate test data’s.

1. Test the admin can successfully login
2. Test the coach can successfully login
3. Test the software not allowing any other logins and shows error message

After this the test results will be logged. (Expected result: admin and coach login while wrong password show error message to user.)

# Testing Strategy

The test strategy we will be using is the bottom-up testing. With this strategy we will be looking at the smaller less significant parts of the software and then work out way through the software testing more of the individual functions until we can start combining them all together into one single program.

This strategy goes hand in hand with object-oriented programming as out program is already made up of different multiple different objects that can be tested by themselves. One they all behave and perform as intended they can be combined.

# 7.0 Testing Levels

## Unit Testing

Each of the programs functions will be tested individually. This will prevent any of the other components interacting with each other and possibly creating a combined bug. Instead we will test each of the components on their own, and ensure they perform as expected. For example:

* Login system
* Creating member skills and details XML file if not already existing
* Displaying players on screen
* Disabling editing controls depending on user type
* Creating new member forms
* Updating existing player forms
* Removing players

## Integration Testing Once the individual units have been tested the software will being integration testing where the function will be tested together to ensure none of them interfere with each other and operate smoothly inside the software system.

## System Testing

The system will be tested as whole. No individual unit or component will be looked at directly but rather the entire system will be tested on performing its intended tasks. This will ensure that the software meets all the specified requirements

7.1.4 Acceptance TestingAt this stage of testing the software will be given to actual software users. This will check if the system can operate in its intended real-world environment. This will help find any bugs that the software developers may have missed.

The testers don’t know how the software operates on the inside, so they may attempt to execute tasks in such a way that it breaks the software. This is very useful as developer may not be able to predict such behaviours on their own before realising the final build.

This will also give the customer a look on how the final product will operate and look allowing to give any necessary feedback to possibly make final changes.

## Installation Testing

The final software version will be installed on the customers hardware and final checks will be performed to assure the customer it’s all working. Check the software installs, starts up and performs all the basic features like logging in, adding/removing and editing member details and skills as well as the navigation tests.

## Entry Criteria

The test will be beginning on the 1st of April and will run throughout the month.

# 9.0 Exit Criteria

*The tests will stop a week before software has to be delivered to give time for any changes. This will be the 25th of April.*

# 10.0 Environmental Needs

|  |  |
| --- | --- |
| **Hardware** | **Software** |
| PC | Visual Studio |
|  | Microsoft Word |
|  |  |

# 11.0 Staffing and Training

As written in the Software Requirement Specification there will be 1 session a week for 2 weeks for the coach and secretary. The test will run on the customers hardware as it’s where the software will run by default. Both users will have their separate time to ask questions and be shown how to perform their basic tasks of viewing, editing and removing member records of skills and personal details.

A detailed user guide will be also provided with pictures which will explain any common questions that may arise during software use.

# 12.0 Test Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| **Task Name** | **Start** | **Milestone/Finish** | **Comments** |
| System Analysis | 01/04/2018 | 02/04/2018 | Approved |
| Getting Test Scenarios | 02/04/2018 | 04/04/2018 | Approved |
| Unit Testing | 04/04/2018 | 14/04/2018 | Approved |
| Meeting | 15/04/2018 | 15/04/2018 | Approved |
| Test Case | 16/04/2018 | 20/04/2018 | Approved |
| Acceptance Testing | 20/04/2018 | 25/04/2018 | Approved |

# 13.0 Testing Deliverables

|  |  |  |
| --- | --- | --- |
| **Deliverable** | **For** | **Date/Milestone** |
| Test Plan | Project Manager | 02/04/2018 |
| Data Validation Logs | Project Manager | 06/04/2018 |
| Event Validation Logs | Project Manager | 09/04/2018 |
| Functional Acceptance Logs | Project Manager | 12/04/2018 |
| Navigation Logs | Project manager | 14/04/2018 |
| User Guide | Project Manger | 20/04/2018 |

## 14.0 Features to be Tested

Unit testing will be performed on each of the programs functions to ensure they operate correctly this test things like:

* The program correctly calculates the age of new members
* The program correctly detects if there are any duplicate SRUs being entered
* The program validates emails
* The program checks for any empty fields
* The program correctly reads from the XML file

Usability testing will be performed by testers who have not worked on the program so that unexpected behaviours can be observed and considered. Testers will be asked to perform tasks and their efficient will be graded on a scale. They can also provide any feedback that may be helpful in smoothing out the user experience.

Software security will be tested to make sure only the correct user type and password combinations work. This will prevent any unwanted access to the sensitive member details containing names, emails and birthdays.

## Functional testing will be performed last to check if all the specified requirements have been met. The main ones are:

* User can log into their account using different passwords and user types
* Age groups can be cycled through or changed using drop down showing only players in that age group
* Members can be removed from roster
* User can log out
* Player Profile button can be pressed and display the selected members skill sheet
* Player Form button can be pressed with a selected player to show their details
* New player forms can be created in player form window by pressing New Member button
* You can save player forms
* You can save player profiles

15.0 Features not to be Tested  
The software will not be tested on its performance when dealing with large numbers of members which may slow down the loading and displaying of member details.

# 17.0 Approvals

|  |  |  |
| --- | --- | --- |
| **Prepared By** | **Signature** | **Date** |
| Author  Job Title  Organisation | Krzysztof Dworczyk  Software Tester  Simply Rugby | 01/04/18 |

|  |  |  |
| --- | --- | --- |
| **Accepted By** | **Signature** | **Date** |
| Project Manager  Job Title  Organisation | Krzysztof Dworczyk | 01/0418 |