WattFarm

Functional Testing Plan and Test Case Descriptions

Version 1.0

Sean Mann

April 9, 2018

Table Of Contents

1 INTRODUCTION

1.1 Purpose of the Test Plan Document

The Test Plan document outlines the approach that will be taken in the testing of this project's product. It will keep track of the necessary information needed to to define the testing approach. This document is created during the development phase of the project, and will be updated throughout the development and delivery phases. The information in this document is pertinent to the developer, project manager, and tester. This document will be presented in the second demo to illustrate the testing process, and feedback from that demo will be taken into account in future versions.

2 COMPATIBILITY TESTING

Compatibility Testing will be covered in later iterations of this document. This is because the product needs to be more full developed in order to warrant and require compatibility testing.

3 CONFORMANCE TESTING

Conformance Testing will be covered in later iterations of this document. However, throughout development the product will be checked based on the specified standards laid out in the course materials. It will also be developed with its specified standards from the design document.

4 FUNCTIONAL TESTING

4.1 Test Risks / Issues

Due to the graphical/visual nature of some of the features of the product, certain testing may be difficult to automate, quantify, or check. To mitigate this potential issue, more extensive system and field testing may need to be done to supplement functional testing.

4.2 Items to be Test / Not Tested

Item to Test	Test Description	Test Date
Graphic.paintComponent	Test the graphics output of a plot of data	Week 13
Graphic.export	Tests to make sure the Graphics can output the graph in multiple formats	Week 14
GUI.loginSys	Test the main login system launched by the GUI. Check to make sure the Session is accurately updated.	Week 14
GUI.createLoginSys	Tests creating a new login user for both Rower and Coach	Week 14
GUI.rowerMainMenu	Tests functionality of the rower Main Menu	Week 14
GUI.coachMainMenu	Tests functionality of the coach Main Menu	Week 14
GUI.graphicsMenu	Tests functionality of the graphics menu. Users should be able to create and view graphics.	Week 14
GUI.profileMenu	Tests ability of users to change information and data in their profile.	Week 14
GUI.teamMenu	Tests a Coach user's ability to manipulate and view data based off a team.	Week 14
GUI.workoutMenu	Functionality test for manually entering data for a workout and changing workout data.	Week 14

4.3 Test Approach

The approach to testing will be mainly based off two primary methods: JUnit testing and Field Testing. Field Testing will supplement for the graphic and visual-based functionality. Field testing through the use of helper classes/methods will be best suited due to a large portion of the project being based off visual representation.

4.4 Test Regulatory / Mandate Criteria

There are no regulator / mandate criteria for these tests.

4.5 Test Pass / Fail Criteria

The graphical and visual testing will have slightly more organic/subjective passing and failing criteria. It will be based on the “look and feel” of the feature, as well as the functionality requirements of that given feature. This means in order to pass, a feature must successfully complete its requirements while being visually usable and appealing. For example, if a feature would technically work, but the user isn’t able to access or understand it based on display errors, that feature would fail its test criteria.

4.6 Test Entry / Exit Criteria

For the visual and graphical features, mock objects may be used to allow for a simpler environment rather than needing excessive setup in the testing class/methods.

4.7 Test Deliverables

The only test deliverables will be the whatever displays are generated by the visual and graphical features.

4.8 Test Suspension / Resumption Criteria

Testing will be continuous for all features, unless a bug or other issue is present that causes testing to be (a) unproductive or (b) not doable.

4.9 Test Environmental / Staffing / Training Needs

All testing will be done by the head tester throughout the development stage. Training/ learning will be on a strict need-to-know policy.

5 UNIT TESTING

5.1 Test Risks / Issues

Due to the graphical/visual nature of some of the features of the product, certain testing may be difficult to automate, quantify, or check. To mitigate this potential issue, more extensive system and field testing may need to be done to supplement functional testing.

5.2 Items to be Test / Not Tested

Item to Test	Test Description	Test Date
Coach Class	Constructor tested using JUnit	Week 13
Team Class	Constructor tested using JUnit	Week 13
Workout Class	Constructor tested using JUnit	Week 13
PM data aquisition	Tests the ability to obtain data from a PM and log workout data.	TBD
Database	Functions tested includes setting up a database connection, getting values, entering values, and changing values.	Week 15
Profiles.Rower/CoachLogin	Tests to see if logins are validated correctly via database.	Week 15

5.3 Test Approach

The approach to testing will be mainly based off two primary methods: JUnit testing and Field Testing. The project is written in Java, which means automated and parametrized testing can be done using the JUnit framework. JUnit testing will cover the data-centered functionality.

5.4 Test Regulatory / Mandate Criteria

There are no regulator / mandate criteria for these tests.

5.5 Test Pass / Fail Criteria

For JUnit tests to pass, they must satisfy the JUnit framework's system for passing and failing. The JUnit tests will be designed to document the reason for failure, and will output a successful result upon passing.

5.6 Test Entry / Exit Criteria

Using JUnit's built in entry and exit criteria, all criteria will be completed before the test methods are run. JUnit methods can use execution ordering so that all conditions and/or on-going activities that must be present before a process can begin are in their appropriate state.

5.7 Test Deliverables

The only test deliverables will be the results from the JUnit Testing.

5.8 Test Suspension / Resumption Criteria

Testing will be continuous for all features, unless a bug or other issue is present that causes testing to be (a) unproductive or (b) not doable.

5.9 Test Environmental / Staffing / Training Needs

All testing will be done by the head tester throughout the development stage. Training/learning will be on a strict need-to-know policy.

6 TESTING METHODS NOT USED

The testing methods outlined in this document will be sufficient for the requirements of the project. Because of this, other testing methods will not be used, such as Load Testing, Performance Testing, Regression Testing, Stress Testing, System Testing, or User Acceptance Testing.

7 TEST CASE DESCRIPTIONS

Item to Test	Test Case Description
Graphic.paintComponent	Confirm plot created Confirm plot visible to user Confirm correct axis formatting Verify data plotted correctly/accurately Verify overall formatting meets quality requirements

Graphic.export	Confirm graphic export created Confirm export filetype is correct Confirm export is reachable by user Verify appearance/quality of export
GUI.loginSys	Confirm user type window is visible Confirm user type selection is working Confirm login window is visible Verify correct login and user data exchange Confirm user logged in Verify Session update Verify correct subsequent procedures
GUI.createLoginSys	Confirm create Login possible Confirm data entry received Verify information correctly entered to database Verify user login works
GUI.rowerMainMenu	Confirm menu window visible Confirm buttons/options working Verify correct subsequent procedures
GUI.coachMainMenu	Confirm menu window visible Confirm buttons/options working Verify correct subsequent procedures
GUI.graphicsMenu	Confirm menu window visible Confirm buttons/options working Verify correct subsequent procedures
GUI.profileMenu	Confirm menu window visible Confirm buttons/options working Verify correct subsequent procedures
GUI.teamMenu	Confirm menu window visible Confirm buttons/options working Verify correct subsequent procedures
GUI.workoutMenu	Confirm menu window visible Confirm buttons/options working Verify correct subsequent procedures
Coach Class	Confirm constructor functionality Confirm getter/setter functionality
Team Class	Confirm constructor functionality Confirm getter/setter functionality
Workout Class	Confirm constructor functionality Confirm getter/setter functionality

PM data aquisition	TBD
Database	Confirm database functionality Verify related procedures implemented correctly
Profiles.Rower/CoachLogin	Confirm connection opened Confirm query functionality Confirm query changes Verify change in data Verify data transferred correctly

8 KEY TERMS

PM = Performance Monitor, indoor rowing machine workout data collector.