**CS673S16 Software Engineering** 

**Team 3 - The Wishlist**

**Tests Report**

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Duncan Morrissey | Project Leader,  Configuration Leader | *Duncan Morrissey* | 4/25/2018 |
| Edward Ryan | Design Leader,  QA Leader | *Edward Ryan* | *4/26/2018* |
| Yiannis Karavas | Requirements Leader, Implementation Leader | *Yiannis Karavas* | 4/26/2018 |
| Zach Lister | Backup Project Leader,  Management Plan Leader | *Zach Lister* | 4/26/2018 |
| Ben Mitchell | Security Leader | *Benjamin Mitchell* | 4/25/18 |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Change** |
| **3.0** | **Edward** | **4/24/18** | **Updated results for Iteration 3; corrected metric errors from Iterations 1 and 2** |
| **2.0** | **Edward** | **3/29/18** | **Updated results for Iteration 2** |
| **1.0** | **Duncan, Edward, Yiannis, Zach, Ben** | **3/1/18** | **As issued** |

[Introduction](#_87t9hln2vjz0)

[Test Summary](#_sm5odwyvuk3j)

[Tests Reports](#_2iy9xpvb9o9s)

[Testing Metrics](#_mtfbusfb0eq3)

[References](#_15tmymhipvdv)

[Glossary](#_8n34lvocupub)

# Introduction

This document describes the testing activities performed during development of The Wishlist application and the results obtained from them. More detailed information about the results of specific tests may be found in the referenced documents.

# Test Summary

## 2.1 Definition and Scope of Test Types

### Unit Testing

Unit tests should be implemented by a developer either (a) in conjunction with the implementation of a feature or (b) in conjunction with implementing corrective action for a defect. Unit tests can be run at any time by any team member, but their pass/failure state will only be formally reported at the end of each iteration by the QA leader.

### Integration Testing

Integration testing occurs when features are implemented to ensure that features implemented previously or concurrently do not introduce defects. Failures found during integration testing are reported as defects and may be reported by any team member.

### System Testing

System testing occurs at the end of each iteration, ensuring the product functions in the target production environment. Failures during system testing are reported as defects and may be reported by any team member.

### Functional Testing

Functional testing takes two forms: ad-hoc and formal functional testing. Ad-hoc functional testing can be performed at any time by any team member, and failures found are reported as defects.

Formal functional testing is the execution of test cases confirming the product functions as specified. Formal functional testing is performed at the end of each iteration. Failures during formal functional testing are reported as defects and as test cases being marked as failed. Formal functional testing is lead by the QA leader, but may be performed by any team member.

### Performance Testing

Performance testing ensures the product functions within the specified performance requirements. Performance tests can be run informally at any time by any team member, but their pass/failure state will only be formally reported at the end of each iteration by the QA leader.

## 2.2 Test Coverage by Iteration

The following sections describe what testing was performed in which areas for each iteration. Please note that not testing was performed for Iteration 0, so it is intentionally omitted.

### Iteration 1

Unit testing focused on the basic Create-Retrieve-Update-Destroy (CRUD) operations for the core application models, as well as the implementations for user creation/authentication, list creation/deletion and list item creation/deletion. Informal integration testing was performed as these features were added, with more formal system testing performed once all of the features had completed work for the iteration. Acceptance tests were run as a part of feature completion.

### Iteration 2

Unit testing framework transitioned to RSpec framework. This resulted in some unit test restructuring in addition to additions for new functionality. Expanded functional testing to include newly implemented functionality for the iteration. Acceptance tests were run as a part of feature completion.

### Iteration 3

Unit tests updated for new/refactored functionality. Functional tests expanded to include both new functionality and to cover performance/usability requirements. Acceptance tests were run as a part of feature completion.

# Tests Reports

## 3.1 Unit Testing Reports

Unit tests are separated out into major areas based on the portion of the architecture under test (be it model, controller, helper, etc.).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Iteration** | **Area** | **Total Tests** | **New Tests Created** | **Test Pass Rate** |
| **Iteration 0** | No unit tests were included as a part of Iteration 0 | | | |
| **Iteration 1** | Controllers | 7 | 7 | 7/7 (100%) |
| Helpers | 1 | 1 | 1/1 (100%) |
| Models | 24 | 24 | 24/24 (100%) |
| **Iteration 2** | Controllers | 7 | 0 | 7/7 (100%) |
| Helpers | 1 | 0 | 1/1 (100%) |
| Models | 40 | 16 | 40/40 (100%) |
| Other | 4 | 4 | 4/4 (100%) |
| **Iteration 3** | Controllers | 8 | 1 | 8/8 (100%) |
| Helpers | 1 | 1 | 1/1 (100%) |
| Models | 42 | 2 | 42/42 (100%) |
| Other | 5 | 1 | 5/5 (100%) |

## 3.2 System Testing Reports

Please refer to the Test Case Report for the appropriate iteration for details regarding the test cases executed and their results.

|  |  |
| --- | --- |
| **Iteration** | **Test Case Report Link** |
| 0 | No test cases were executed this iteration |
| 1 | <https://drive.google.com/open?id=106c8tmhS8jay8uByrmHbv3_4egqHV3ML> |
| 2 | <https://drive.google.com/open?id=1v02lPZjk2eY6oRUzkmdj8P-KX2C8CZfX> |
| 3 | <https://drive.google.com/open?id=1EZ1loXxIy9jZKmzDRRbfwAnTaNHZef_G> |

## 3.3 Acceptance Testing Reports

Please refer to the Test Case Report for the appropriate iteration for details regarding the test cases executed and their results.

|  |  |
| --- | --- |
| **Iteration** | **Test Case Report Link** |
| 0 | Acceptance tests tracked as a part of feature completion. Please refer to PivotalTracker (<https://www.pivotaltracker.com/n/projects/2148000>) for additional information. |
| 1 | Acceptance tests tracked as a part of feature completion. Please refer to PivotalTracker (<https://www.pivotaltracker.com/n/projects/2148000>) for additional information. |
| 2 | Acceptance tests tracked as a part of feature completion. Please refer to PivotalTracker (<https://www.pivotaltracker.com/n/projects/2148000>) for additional information. |
| 3 | Acceptance tests tracked as a part of feature completion. Please refer to PivotalTracker (<https://www.pivotaltracker.com/n/projects/2148000>) for additional information. |

## 

# Testing Metrics

The following tables provide a brief description of the various test metrics and how to interpret changes in their values. They also include the results of those metrics at the end of each iteration. Please note that a value of ‘n/a’ means that there was not enough information available at the end of the iteration to report that metric. Iteration 0 is intentionally omitted as there was not enough information to report any testing metric.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Product - Complexity** | | | | |
| **Metric** | **Description** | **Iteration 1** | **Iteration 2** | **Iteration 3** |
| **Lines of Code (LOC)** | The total lines of code in all application files. An increase in value correlates to an increase in complexity. | 983 | 1474 | 1638 |
| **Number of Files** | The total number of files in the application. An increase in value correlates to an increase in complexity. | 131 | 170 | 169 |
| **Number of Classes** | The total number of classes defined in all application files. An increase in value correlates to an increase in complexity. | 23 | 18 | 18 |
| **Number of Methods** | The total number of methods/functions defined in all application files. An increase in value correlates to an increase in complexity. | 68 | 49 | 58 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Product - Stability** | | | | |
| **Metric** | **Description** | **Iteration 1** | **Iteration 2** | **Iteration 3** |
| **Defects found (iteration)** | The number of defects found in the iteration. An increase in value correlates to a decrease in stability. | 5 | 2 | 2 |
| **Defects found (total)** | The number of defects found across all iterations. An increase in value correlates to a decrease in stability. | 5 | 7 | 9 |
| **Defects fixed (iteration)** | The number of defects fixed in the iteration. An increase in value correlates to an increase in stability. | 0 | 5 | 4 |
| **Defect backlog** | The number of defects that are waiting to be fixed in the current release. An increase in value correlates to a decrease in stability. | 5 | 2 | 0 |
| **Residual defects** | The number of defects that have been elected to either not be fixed or to be fixed in a future release. An increase in value correlates to an decrease in stability. | 0 | 0 | 0 |
| **Defect fix rate** | The ratio of defects fixed to defects found in the iteration. An increase in value correlates to an increase in stability. | 0 | 5:2 | 4:2 |
| **Distribution of defect severities\*** | The percentage of defects associated with each severity category. A distribution favoring lower severities correlates to an increase in stability. | 4 - 0/5 (0%)  3 - 4/5 (80%)  2 - 1/5 (20%)  1 - 0/5 (0%) | 4 - 0/7 (0%)  3 - 4/7 (57%)  2 - 3/7 (43%)  1 - 0/7 (0%) | 4 - 0/9 (0%)  3 - 6/9 (67%)  2 - 3/9 (33%)  1 - 0/7 (0%) |
| **Unit test pass rate** | The percentage of unit tests that pass on the build at the end of the iteration. An increase in value correlates to an increase in stability. | 100% | 100% | 100% |

\*\* Please refer to the Glossary Section 6.1 for definitions of severities.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Product - Cost** | | | | |
| **Metric** | **Description** | **Iteration 1** | **Iteration 2** | **Iteration 3** |
| **Total hours spent on project** | The total hours spent on any aspect of the project. An increase in value correlates to an increase in cost. | 203.5 | 342 | 479 |
| **Distribution of hours by product activity\*\*** | The percentage of hours spent on each project activity. This metric shows what activities are incurring cost, rather than affecting the cost by its value alone. | 0 - 59 (28.99%)  1 - 7.5 (7.37%)  2 - 15 (30.96%)  3 - 63 (30.96%)  4 - 17 (8.35%)  5 - 33 (16.21%)  6 - 9 (4.42%) | 0 - 74 (21.64%)  1 - 13.5 (3.95%)  2 - 22.5 (6.58%)  3 - 146 (42.69%)  4 - 29 (8.48%)  5 - 47 (13.74%)  6 - 10 (2.92%) | 0 - 82 (17.12%)  1 - 17.5 (3.65%)  2 - 27.5 (5.74%)  3 - 220(45.93%)  4 - 53 (11.06%)  5 - 66 (13.78%)  6 - 13 (2.71%) |

\*\* Please refer to the Glossary Section 6.2 for definitions of activities.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Process - Testing** | | | | |
| **Metric** | **Description** | **Iteration 1** | **Iteration 2** | **Iteration 3** |
| **Unit test code coverage** | The percentage of the application code tested by at least one unit test. A higher value is better. | n/a  (simplecov not yet set up) | 88.75% | 85.49% |
| **Ratio of Lines of Code to Lines of Test Code** | The ratio lines of “production” code to lines of code in testing. This value should be somewhere in the vicinity of 1:1.0-2.0 | 1:0.4 | 1:0.5 | 1:0.5 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Process - Defect Tracking** | | | | |
| **Metric** | **Description** | **Iteration 1** | **Iteration 2** | **Iteration 3** |
| **Defects awaiting disposition** | The total number of defects awaiting disposition at the end of the iteration. At the end of each iteration, this value should be zero. | 0 | 0 | 0 |
| **Defects missing severity/probability** | The total number of defects missing either the severity or probability categorization at the end of the iteration. At the end of each iteration, this value should be zero. | 0 | 0 | 0 |

# References

A general overview of the Quality Assurance plan for this project may be found in the Software Project Proposal & Planning document.

# Glossary

## 6.1 Definitions of Defect Severities

All defects must be assigned a severity as a part of the justification for how a defect is dispositioned. A defect may be assigned one of the following severities based on the associated criteria:

|  |  |
| --- | --- |
| **Severity** | **Problem Type** |
| **4 - Critical** | Problem will cause the product to become unresponsive and unable to recover. Unintended data loss is likely or guaranteed. |
| **3 - Serious** | Problem severely restricts use of the product for frequently or for a prolonged duration. Work around is nonexistent or difficult.  OR  Problem could result in the product being unresponsive.  OR  Problem could result in unintended data loss. |
| **2 - Medium** | There is a simple work around for the problem and product use is acceptable.  AND  Problem does not result in the product being unresponsive. |
| **1 - Low** | Problem is a minor inconvenience to user, and product use is acceptable.  AND  Problem does not result in the product being unresponsive. |

## 6.2 Definitions of Task Breakdown Activities

Time spent working on the application is divided into 7 categories (numbered 0 to 6):

|  |  |
| --- | --- |
| **ID** | **Description** |
| 0 | Learning |
| 1 | Requirement Analysis |
| 2 | Design |
| 3 | Implementation |
| 4 | Test |
| 5 | Communication/Management |
| 6 | Unclassified |