**CS673S16 Software Engineering**

**Team 4 - Project Name**

**Tests Report**

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Oindrilla Chatterjee | Junit testing | *OC* | 11/23/2017 |
| Asli Terzioglu | Selenium Testing | *AT* | 11/26/2017 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Change** |
| **1.0** | **Asli Terzioglu** | **11/26/2017** |  |
|  |  |  |  |

[Introduction](#_87t9hln2vjz0)

[Test Summary](#_sm5odwyvuk3j)

[Tests Reports](#_2iy9xpvb9o9s)

[Testing Metrics](#_mtfbusfb0eq3)

[References](#_15tmymhipvdv)

[Glossary](#_8n34lvocupub)

# 1.0 Introduction

## **1.1** Purpose

This document covers test cases for all the features mentioned in the SDD document. The goal of this document is to answer what/why/when/how questions surrounding the verification test details of the project:

* What is the high-level test strategy for this project, from initial planning all the way testing?
* What functions/requirements need to be covered?
* What is needed in terms of software, tools, and equipment to cover it?
* When will it be covered (i.e., new or old)
* Why does it need to be covered? (i.e., requirement mapping).
* ·How will it be covered?

## **1.2** Document Organization

This document is organized as follows:

**Section 1.0: Introduction**. This section introduces the goal and the flow of the document.

**Section 2.0: Test Summary.** This section is considered the high-level unified plan covering unit tests, system tests and acceptance tests. Each subsection in this section have different testing stages and purposes for the project.

**Section 3.0: Test Reports.** This section presents a detailed description of each test scenario, the execution process and the outcome.

**Section 4.0: Test Metrics.** This section identifies the test metrics which are generally used to measure the overall process, monitor and improve process quality, e.g. coverage, defects rate, etc.

**Section 5.0: References & Glossary.** This section identifies the reference documents.

# 2.0 Test Summary

## 2.1 Test Stages

The test scenarios set up for all the features mentioned in the SDD document with giving priority to the essential functional features. The following table further breaks down the overall various stages of testing that occur throughout a project.

**Table 1 - Stages of Testing**

|  |  |  |
| --- | --- | --- |
| **Project Test Activity** | **Description** | **Requirements** |
| Unit Tests | Tests an individual component within a subsystem such as classes and major methods. | Unit testing of the project will be performed on the software development environment. We write Django unit tests using the *unittest* module in to the Python standard library. Test code has been uploaded to Github. |
| System Tests | Provides a complete verification of feature operations in a working system for both success path and failure path against the requirements in the SDD. | We use Selenium tool integrated with our Django test suite to perform the system test and the code has been uploaded to Github. |
| Acceptance Tests | Verify how well the system meets major functional requirements and selected nonfunctional requirements. |  |

# 3.0 Tests Reports

## 3.1 Test Cases

We prepared the test scenarios in another document with the following details:

● Test case ID, name

● New or old:

● Test items: (what do you test )

● Test priority (high/medium/low)

● Dependencies (to other test case/requirement if any):

● Preconditions: (if any)

● input data:

● Test steps:

● Postconditions:

● Expected output:

● Actual output:

● Pass or Fail:

● Bug id/link:

● Additional notes:

Please check Test Cases sheet on Team4\_defect\_tracking\_log file on Google Drive for test scenarios.

## **3.2** Issue Tracking

Any issues identified during the execution of test cases will be tracked using use cases. All issues should be reported to the Pivotal Tracker and also Team4\_defect\_tracking\_log document. Each case will be entered in to these documents with a priority agreed with the group members.

The case priorities are as follows:

**Table 2 – Priorities of issues**

|  |  |
| --- | --- |
| **Priority** | **Definition** |
| Critical | An issue which is blocking the system’s operation, maintenance, or administration. Issues threatening the core functions of the project at the critical level will be assigned a priority of Critical which means there are no alternatives and has to be fixed immediately. |
| High | An issue which has a potential risk and serious impact on the system’s core functionality, maintenance, or administration, however the rest of the system is still functional. |
| Medium | An issue which impacts minor functional and does not have a serious impact to the system’s operation, maintenance, or administration but have undesirable behaviors. |
| Low | An issue usually according to improvement to product features due to bad usability or based on feedback. For example, issues which do not cause a test case failure but which are noticed during system integration or testing may be assigned a priority of Low. |

Cases will be tracked by team members during the testing period. They will either provide the fix or route the issue to the relevant other case.

# 4.0 Testing Metrics

**Test Case Preparation Productivity:**

Test Case Preparation Productivity = (No of Test Case)/ (Effort spent for Test Case Preparation)

No. of Test cases = 28

Effort spent for Test case preparation (in hours) = 36

Test Case preparation productivity = 28/36 = 0.77 test cases/hour

**Test Design Coverage:**

Test Design Coverage = ((Total number of requirements mapped to test cases) / (Total number of requirements)\*100

Total number of requirements: 35

Total number of requirements mapped to test cases: 28

Test Design Coverage = (28/35)\*100 = 80%

**Test Execution Productivity:**

Test Execution Productivity = (No of Test cases executed)/ (Effort spent for execution of test cases)

No of Test cases executed = 28

Effort spent for execution of test cases = 52

Test Execution Productivity = 28/52 = 0.53 test cases/hour

**Test Execution Coverage:**

Test Execution Coverage = (Total no. of test cases executed / Total no. of test cases planned to execute)\*100

Total no. of test cases planned to execute = 32

Total no. of test cases executed = 28

Test Execution Coverage = (28/32)\*100 = 87.5%

**Test Cases Passed:**

Test Cases Pass = (Total no. of test cases passed) / (Total no. of test cases executed) \* 100

Total no. of test cases passed = 25

Total no. of test cases executed = 28

Test Cases Pass = (25/28)\*100 = 89.28 = 89.28%

**Test Cases Failed:**

Test Cases Failed = (Total no. of test cases failed) / (Total no. of test cases executed) \* 100

Total no. of test cases failed = 3

Total no. of test cases executed = 28

Test Cases Failed= (3/28)\*100 = 10.7 = 10.7%

**Error Discovery Rate:**

Error Discovery Rate = (Total number of defects found /Total no. of test cases executed)\*100

Total no. of test cases executed = 28

Total number of defects found = 8

Error Discovery Rate = (8/28)\*100 = 28.57%

**Defect Fix Rate:**

Defect Fix Rate = (Total no of Defects reported as fixed - Total no. of defects reopened) / (Total no of Defects reported as fixed + Total no. of new Bugs due to fix)\*100

Total no of defects reported as fixed = 8

Total no. of defects reopened = 2

Total no. of new Bugs due to fix =1

Defect Fix Rate = ((8–2 )/(8 + 1))\*100 = (6/9)100 = 66.6 = 66.66%

**Defect Density:**

Defect Density = Total no. of defects identified / Actual Size (requirements)

Total no. of defects identified = 8

Actual Size= 26

Defect Density =8 /26 =0.30

# 5.0 References

# 6.0 Glossary

SDD - Software Design Document