Test Document

Team 1

8 April 2012

Table 1: Team

Name	ID Number
Jonathon Bergeron	9764453
Marc-Andre Faucher	9614729
Jeffrey How	9430954
Dmitry Kuznetsov	5679311
Willian Ling	9193480
Thomas Paulin	9333630
Alain Sakha	9770836
Kai Wang	5652723

1 Introduction

This testing plan document outlines the testing to be done on the system that ensures that it meets specified requirements in a proper, organized way, and provides a reasonable user experience within those requirements and that feature set.

Specifically, the goals of this document are to formulate a scheduled plan to thoroughly validate that system requirements are met at the system, subsystem and unit levels, along with the inclusion of integration and regression testing, and to list the results of these tests in a manner in which the requirements they meet are known. Additionally, this document aims to verify that the way in which the requirements are implemented provides a certain feature set that has the ability to surprise and delight users.

The information contained in this document includes a brief overview of the forms of testing done on the system, the schedule and resources required and used for testing, brief notes on the design of the test cases, as well as detailed test results at each aforementioned level, along with their results and the system requirements which they are intended to satisfy.

2 Test Plan

The testing planned for the system spans the various levels of production. As the system was tested starting at the unit level, the method of testing was bottom up. Regardless, the tests will be outlined in a top-down fashion, based on the complete design and implementation of the system.

2.1 System Level Test Cases

These system test cases define overarching test cases for the software that are designed to test both the design and behavior of the system and its main functional requirements. Specific, detailed scenarios will be covered in the sections describing subsystem and unit tests.

Test Case 1

Purpose
State the purpose of the test. Indicate which requirement and which aspect of that

requirement is being tested.

Input Specification
State the context for the test in terms of system state. State the input test data. You may need to mention operations invoked as well as data for the operation. You can cross-reference to actual file data specified in an appendix.

Expected Output
State the expected system response and output. You can cross-reference to actual file data specified in an appendix.

Traces to Use Cases
State which requirements (at the level of use case and scenario) are tested by this test case.

2.2 Subsystem Level Test Cases

The subsystem tests for the system are designed to test specific collections of units of functionality within the system. As the system separates requirements by the Model, View, Controller architecture, those will serve as the major subsystems of the system.

One subsection per subsystem

Subsystem X

2.3 Unit Test cases

All test cases for testing at the unit level.

One subsection per unit

Unit X

2.4 Regression Testing

2.5 Integration Testing

Implementation testing was performed as components were added to the system. While there are no specific test cases, the regression test were run as components were added.

As stated earlier, the system was implemented in a bottom-up manner, meaning that modules were slowly coupled together, and implemented beginning at the unit. Below is a diagram of the way in which the modules of the system were integrated:

[INSERT BOTTOM-UP DIAGRAM HERE]

2.6 Schedule and Resources for Testing

This section defines the schedule and resources used in the testing of the system.

Testing Schedule

[TESTING SCHEDULE]

Resources Used

[RESOURCES USED]

2.7 Notes on Test Case Design

The test cases were designed to be operated within the confines of the MVC system, meaning that each case was written and performed keeping the notion of a separate Model, View and Controller in mind. This means that test cases need to ensure that each separate portion of the system maintained integrity both as a whole, and separately.

2.8 Requirements Tested and Omitted

Requirements Tested

The requirements tested in the development of the system are listed in the requirements document and in Appendix C of this document. The specific requirements are labeled functional requirements F2-F10, F13, as well as operational requirement O1, with the omission of functional requirement F1, F11, F12 covered in detail below.

Requirements Omitted

The requirements listed in the requirements document (included in Appendix C) omitted from the testing and eventual development of the system are requirements F1 and N1.

Functional requirement F1 states Only one person should be logged in at a time. This requirement was later dropped from development during the development of Phase 2, as it was deemed beyond the scope of the application, as the application functioned better in a more general form. This functionality was thus not tested as part of the requirements.

Functional requirement F11, F12 were discarded, as the notion of administrator users was omitted as part of F1. As this is the case,

Non-functional requirement N1, specifically the system must know who the user is in order to order the appropriate options, was thus rendered invalid, as the entire notion of different users using the system was discarded.

3 Test Results

List the tests, indicating which passed and which did not pass. List requirements indicating the percentage of tests that passed for that requirement.

4 References

A Description of Input Files

A.1 People.xml

¡people¿ : Root node used to specify the beginning of the listing of people

¡person¿ : Node used to specify a new person

¡identifier¿ : A unique, alphanumberic identifier for each person

¡fname¿ : Node used to store the first name of the person

¡lname¿ : Node used to store the last name of the person

¡jobtitle¿ : Node used to track the job title of the person. Used to differentiate between-roles

¡jobdescription¿ : Node used for the person's job description.

¡clearance¿ : Node used to specify the clearance a person within the organization. Restricts permission based on value. 0 - Employee, 1 - Manager, 2 - Administrator

A.2 A.2 Tasks.xml

¡tasks¿ : Root node used to specify the beginning of the listing of tasks

¡task¿ : Node used to specify a new task

¡identifier¿ : A unique, alphanumberic identifier for each task

¡title¿ : Node to store a short title for the task

¡description¿ : Node to store a full description for the task

¡startdate¿ : Node to store the appropriate start date for the task

¡duration¿ : The duration, in hours of the task.

¡deliverable¿ : The expected physical deliverable upon task completion

<deadline> : The date by which the task must be completed

<peopleassigned> : A listing of the individuals assigned to the task. The duration will be split evenly among assignees.

<id> : The corresponding identifier for each person assigned

<completion> : The percentage of completion for the task

A.3 Sample Invalid XML Files

B Description of Output Files

B.1 People.txt

```
{
The list of people with a summary of their names, the total amount of time

Name                          | Total Hours | Project List
----------------------------------------------------------------------------
LName , FName                 |         X.X | A, B, C
}
```

C List of Requirements

C.1 Functional Requirements

F1

Only one person should be logged in at a time.

F2

A view of tasks in the company is to be shown to the user in order for him to manage them.

F3

The user is to be able to enter a new task.

F4

The user is to be able to remove a previously created task.

F5

The user is to be able to modify a previously created task.

F6

The user is to be able to view a dependency graph of tasks.

F7

The user is to be able to view a Gantt chart of tasks.

F8

The user must be able to manage employees on a task.

F9

The user must be able to assign employees to a task.

F10

The user must be able to view a list of employees in the company in order for him to manage them.

F11

The user is to be able to add a new employee.

F12

The user is to be able to remove an employee.

F13

The user must be able to withdraw employees from a task.

C.2 C.2 Non-Functional Requirements

N1

The system must know who the user is in order to order the appropriate options.

O1

It shall be possible to run the system on any computer system that runs the JVM.