Testing Documentation

Task Manager - Team #1 - Increment 2

Objective: Review high-level design of system to obtain a reasonable understanding of the system inputs, processes and outputs. Draw up test cases and test the functionality of the Task Manager.

Understanding the System: A Tester's Perspective – Phase 2 Testing

The system takes input XML files (people.xml and tasks.xml) and produces a graphical user interface that allows a user to perform several operations through the use of views. Users may view tables of tasks, allowing them to edit, add and remove tasks. Users may then view a table of people, where general information is available for each person known by the system. Additionally, a user may choose to output a text file describing the people (people.txt). The XML files contain specific data about the company's tasks and employees, which is represented by the tables and output file.

As tested and validated in Phase 1, the basic functional requirements involving the XML files, including structure and data, worked according to plan, and preventative safeguards were put into place. For the testing of Phase 2, the onus was on ensuring that data entering the system was managed and displayed properly, and ensuring that the changes made to the data model through various operations (additions and removals of tasks, changing of assignees, etc.) was handled properly, and represented in each possible output scenario.

Our goal is to give reasonable assurance that the system is performing up to standards. In order to do that, we divided the testing into two major aspects:

1. Regression Testing: Validate that the inclusion of new code does not affect the functionality of the new system.

> **Scope:** Proper XML Structure, Data, Basic functionality tests.
>
> **Background**: With the inclusion of newly developed code, the basic functionality of the system still needs to work, and not be violated in any way by any new code or data changes.
>
> **Expectations**: As many of the methods from the first Increment were re-factored to include different functionality, the expectation was that minor issues would occur, and those would need to be reconciled before moving forward.

2. New Functionality Testing: Validate the functionality of new code included in the project.

> **Scope:** Data usage, output scenarios, overall functionality
>
> **Background:** The integration of the MVC architecture should be done properly and in such a way that no data integrity or functionality is lost or affected.

Expectations: As the program runs, a variety of output scenarios affect the XML and output files in various ways. The goal is to ensure that the system exhibits expected behavior, and does not violate any data or functional requirements.

Understanding the System: A Tester's Perspective – Phase 2 Testing

In accordance with proper Software Engineering principals, regression testing was necessary as the code for Increment 2 evolved. Many of the tests performed were modified versions of those from the first increment, which ensured that our data and format integrity. Those methods were built upon and modified to accurately test the functionality added by the developers for the second increment.

Minimal model-changing code was included on the part of the developers, as it has been properly thought out with the future in mind for the first Increment. This meant that regression testing was quick, and relatively painless as the project evolved. The brunt of the testing came from minor methods in new classes which wrote the data structures, as the user interface had little testability to it.

Conclusion: Many of the contingencies were thought out properly for the first increment, and as such, testing was simply and straightforward. The system maintained proper data integrity throughout, and the changes made, after being refined, did not affect the functionality presented by the first Increment.