**Recipe Box Requirements Document (version 1.0)**

Project: Recipe Box

Date(s): March 1st, 2015

Prepared by: David Jansing

Document status: \_\_Draft \_\_\_ Proposed \_\_ Validated X Approved

1. Introduction

This document contains the system requirements for Recipe Box. These requirements have been derived mainly from discussions with my wife about the project. It is based on a template provided by the Center for Distributed Learning, California State University. Most companies, including my own, have a standard format for the requirements document. This is the one that I will be using for this project.

1.1 Purpose of This Document

This document is intended to guide development of Recipe Box. It would ordinarily go through several stages during the course of the project:

1. **Draft:** The first version, or draft version, is compiled after requirements have been discovered, recorded, classified, and prioritized.
2. **Proposed:** The draft document is then proposed as a potential requirements specification for the project. The proposed document should be reviewed by several parties, who may comment on any requirements and any priorities, either to agree, to disagree, or to identify missing requirements. Readers include end-users, developers, project managers, and any other stakeholders. The document may be amended and reproposed several times before moving to the next stage.
3. **Validated:** Once the various stakeholders have agreed to the requirements in the document, it is considered validated.
4. **Approved:** The validated document is accepted by representatives of each party of stakeholders as an appropriate statement of requirements for the project. The developers then use the requirements document as a guide to implementation and to check the progress of the project as it develops.

For purposes of this demonstration project, let us assume that the requirements document has been approved by all concerned. Approval allows me to start work on Recipe Box.

1.2 How to Use This Document

I expect that this document will be used by people with different skill sets. This section explains which parts of this document should be reviewed by various types of readers.

**Types of Reader**

This document is aimed at any participants who might have a stake in this project. These include Java programmers, graphic designers, technical writers, testers, end users, and project managers. End users can probably skip to Section 3.

**Technical Background Required**

This document is not intended to be overly technical, that is, a deep knowledge of Java should not be required to understand it. However, a fundamental understanding of computer science would be helpful.

1.3 Scope of the Product

Recipe Box is a platform-independent software project for cataloging, displaying, and manipulating recipes, like those kept on 3x5 cards in someone’s kitchen. It provides a convenient means of keeping them handy and available.

1.4 Business Case for the Product

This product’s purpose is to provide a means of demonstrating various automated testing techniques, thus it is not an actual commerical product. It will, however, be useful enough to actually use as a recipe storage program, if desired.

Recipe Box will be simple enough for even a beginning computer user to use effectively. Instead of having a big pile of clippings, 3x5 cards, or recipes printed from the Internet, the user will have a single location from which to store, browse, recall, manipulate, and create recipes.

1.5 Overview of the Requirements Document

This document is comprised of the following parts:

Section 2: A general description of the project.

Section 3: A list of user, system, and interface requirements.

Section 4: A glossary of terms used in the document.

Section 5: A reference citing the source material for the format and content of this document.

2. General Description

This section will give the reader an overview of the project, including why it was conceived, what it will do when complete, and the types of people we expect will use it. We also list constraints that were faced during development and assumptions we made about how we would proceed.

2.1 Product Perspective

This is a somewhat less than trivial, but small software project developed in order to demonstrate test automation techinques. It is nevertheless functional in accordance with most of the requirements listed below. Those requirements that

2.2 Product Functions

The essential parts of this system are:

* A data area where the recipe data are kept.
* A GUI for recalling, editing, and deleting the data. This consists of:
  + A pull-down list of the recipe titles.
  + A collection of buttons for manipulating data.
  + A text box to display the ingredients list.
  + A text box to display the directions for preparation.
  + A text box and button to increase or decrease the number of intended servings, thus scaling the recipe as needed.

2.3 User Characteristics

We would expect this product to be enjoyed by anyone with a love of cooking and the need for a system by which one might save, maintain, and otherwise manipulate a “book” of recipes. We do not expect that using Recipe Box would require any more technical expertise than that required to use a text editor.

2.4 General Constraints

None noted.

2.5 Assumptions and Dependencies

None noted.

3. Specific Requirements

This section of the document lists specific requirements for Recipe Box. Requirements are divided into the following sections:

1. User requirements. These are requirements written from the point of view of end users, usually expressed in narrative form.
2. System requirements. These are detailed specifications describing the functions the system must be capable of doing.
3. Interface requirements. These are requirements about the user interface, which may be expressed as a list, as a narrative, or as images of screen mock-ups.

3.1 User Requirements

1. The recipe data are to be kept in a data directory, in a format that can be interpreted by the system.
2. Recalling, editing, and deleting the data shall be controlled by a graphic interface which shall consist of:
   1. A pull-down list of the recipe titles.
   2. A collection of buttons for manipulating data.
   3. A text box to display the ingredients list.
   4. A text box to display the directions for preparation.
   5. A text box and button to increase or decrease the number of intended servings.
3. It shall be possible for the end user to enter new recipes into the system.
4. It shall be possible for the end user to edit an existing recipe.
5. It shall be possible for the end user to adjust ingredient quantities for varying number of servings.
6. It shall be possible to delete a recipe.
7. Quantities consisting of non-whole numbers shall be expressed in fractions in the user interface.
8. Quantities consisting of non-whole numbers shall nevertheless be stored in the system as decimal quantities, regardless of the measurement system used.
9. During the servings adjustment, large numbers of smaller units shall be automatically converted to larger units. For example, there are 16 tablespoons in ¼ cup.
10. Fractional quantities should be congruent to those found in a typical set of kitchen measuring cups (1/4, 1/3, 1/2, 1)
11. It should be made possible to convert measurements to and from U.S. and metric units.
12. If a graphic of the finished dish is available, it should be displayable by the system.

3.2 System Requirements:

Recipe Box should be usable on any machine on which the Java Runtime Environment, version 1.7.0\_11 or higher, has been installed.

**3.3 Interface Requirements**

A preliminary screen shot of the GUI design is provided below:



There is a pull-down box that holds the recipe titles, along with buttons for getting, editing, and deleting recipes. There is a box for the ingredient list, a pull-down box for suggested ingredient substitutions, a box and button for adjusting the recipe for a desired number of servings, and a box for the cooking directions.

4. Glossary

Pull-down box – a JComboBox

Text box – a JTextField or JTextArea, as applicable

Button – a JButton or jToggleButon, as applicable

Ingredient – this normally consists of a quantity, a unit of measure (optional), and an item.

Serving – this is an arbitrary measure, determined by the recipe author, of how many people a recipe is intended to serve. It also acts as a means to scale a recipe up or down.

Directions – cooking directions, which is a free-form string.

5. References

Rachel S. Smith, *Writing a Requirements Document: Workshop Materials*, Center for Distributed Learning, California State University